DIRECTION AWARE POSITIONAL AND STRUCTURAL ENCODING FOR DIRECTED GRAPH NEURAL NETWORKS

Yonas Sium [1] *Georgios Kollias* [2] *Tsuyoshi Idé* [2] *Payel Das* [2] *Naoki Abe* [2] *Aurélie Lozano* [2] *Qi Li*[1]

[1]Iowa State University
[2] IBM Research, T. J. Watson Research Center

ABSTRACT

We propose a novel method for computing joint 2-node structural representations for link prediction in directed graphs. Existing approaches can be grouped into two families. The first group of methods learn structural embeddings of individual nodes in the entire graph through a directed Graph Neural Network (*GNNs*), and then combine pairs of the encodings to get a representation for the respective node pairs. Methods in the second group compute a representation of the subgraph enclosing the two nodes by employing *GNNs* initialized with positional encodings and consider these as their potential edge embeddings. Both families of link prediction techniques suffer from considerable shortcomings: The former fail to differentiate two distant nodes with similar neighborhoods; The latter, although provably appropriate for learning edge representations, adopt *undirected GNNs*, positional encodings, and subgraphs, so the edge direction signal is inevitably lost. Our proposal is also based on the idea of enclosing subgraphs, but the subgraphs are assumed directed, and directed Graph Neural Networks (*GNNs*) are used to learn their node encodings and initial positional embeddings are direction-aware. Our emphasis on capturing the direction of edges is reflected in superior performance in the link prediction task against baselines with undirected *GNNs* on symmetrized enclosing subgraphs and existing directed *GNNs* over a collection of benchmark graph datasets.
[1]

1. INTRODUCTION

In recent years graph neural networks (*GNNs*) have attracted considerable attention as an effective deep learning model for analyzing graph data([1]). Link prediction (predicting the existence of a link between two nodes) is one of the graph learning tasks that are broadly studied in different domains: e.g. in social networks [2], biological networks [3], recommender systems [4] and knowledge graphs [5]. In many applications the direction of the predicted edge is critical: In knowledge graphs, subject nodes should point to object nodes; In citation graphs a derivative work cannot be pointed by earlier research; In causal graphs a node representing an effect should not prepend its cause. Failure to capture edge directionality could also have disruptive effects on important downstream application tasks: e.g. conflicts in induced time orderings, blocked paths to root causes, distorted flow computations between node sets, etc.

The majority of existing *GNN* models learn the representation of a link by combining the representation learned of the two linked nodes. The linked nodes get their representations through message passing schemes [6] through aggregation of information from the neighboring nodes combined with their own features [7, 8, 9]. With this approach, nodes that have the same local neighborhood get the same representation, and hence, node based link prediction may fail for similar nodes with similar topological structure. Earlier works that proposed directed *GNNs* [10, 11, 12] adopted the above approach so they inhrerit this conceptual flaw.

In [13] the authors identified and formalized two types of encodings for graph nodes: *structural* and *positional*. *GNN* produce *structural* embeddings, which are highly effective for node labeling tasks: we want nodes with similar neighborhoods to be close in the representation space. As mentioned earlier, however, they fail to induce proper *link* representations: two nodes with identical local structure may be undistinguishable and hence lead to a wrong link prediction. They then show that *joint 2-node structural representations* are required for link prediction. On the other hand, *positional encodings* are commonly produced by lower-dimensional projections of the graph adjacency information. Such encodings for nodes that are close neighbors in the graph are close to each other, implying that links should most probably exist between nodes with minimal distances in the corresponding representation space. This is desirable for the link prediction task.

Our task is *directed link prediction*, so according to [13] we would need a scheme of generating either *positional* or *joint 2-node structural* encodings for graph nodes. Our approach blends these two perspectives in an innovative manner, while emphatically preserving the asymmetry arising from directed edges in the original graph. Notably: (i) we use *GNN* to learn structural encodings but do not use them directly in link prediction; (ii) we compute positional encodings but these are only used for *GNN* initialization.

Structural encodings: We extract a k-hop directed subgraph $G_k(u, v)$ around potentially linked nodes u, v, hence satisfying the *joint 2-node structural representation* requirement for link prediction. Directed *GNN* on the subgraph produce structural encodings for individual nodes in the subgraph, so using them pairwise for link prediction would be suboptimal. Hence, after computing the encodings of *all* nodes, we pool them into a single vector summarizing the subgraph as a whole and we assign it to the node pair.

Positional encodings: We initialize *GNN* based on spectral vectors which are positional encodings. They could be used directly for predicting links, but we only use them as input to the directed *GNNs* to produce structural encodings for the nodes. Note that spectral representation for a directed subgraph is different from its symmetrized version and thus our input encodings are direction-aware.

We provide theoretical justification that our directional-aware positional encodings are valid for learning structural encodings of pairs of nodes (edges) in directed graphs and empirically validate its effectiveness by testing it on the link prediction task on directed graphs. Experiments show our proposed approach improves over benchmark baselines including undirected models with distance-based encoding, existing methods for node representation in directed graphs and ex-

[1]This work was done during an internship at IBM Research

isting models enhanced with our proposed structural and positional encodings.

2. RELATED WORK

The idea of representing an edge by the encoding of its enclosing subgraph appears in Weisfeiler-Lehman Neural Machine (WLNM) [14] and the follow up work in SEAL framework [1]. In WLNM, special emphasis is placed on vertex ordering for the adjacency matrix of the input subgraph, using the Weisfeiler-Lehman (WL) coloring procedure for its computation. In SEAL, they introduce a distance-based encoding of nodes in the subgraph and they critically improve on WLNM by leveraging *GNN* for node embedding learning. However, WLNM and SEAL do not address directed graphs, which is our focus here. We use spectral encodings for input node labeling which are inherently *direction-aware* and we process enclosing *directed* subgraphs utilizing *directed GNN*. A further generalization to multi-node representations, rather than two-node ones which is effectively our case with graph edges, is studied in [15].

Our focus is on capturing the directionality of predicted links, so we train *directed GNN* for computing the node encodings of the directed subgraphs. In particular, we use digraph convolution (DiGCN) and its multiscale generalization (DiGCNIB) from [11] and Magnet from [10]. The directed *GNN* we use are based on specific definitions of the Laplacian operator for directed graphs, followed by the standard graph convolution approach in [16]. In [11] they define a symmetric Laplacian for digraphs in terms of the transition matrix for personalized PageRank, which is then extended to higher-order proximity between graph nodes. In [10] a complex-valued, Hermitian Laplacian matrix is introduced. Another stream of research considers improving the power of *GNN* towards getting better representation for a link by fusing positional and structural encoding [15, 17, 18, 13], however not including link direction in their analysis or reported results.

3. MODEL

3.1. Description

Given a pair of nodes (u, v) we (i) extract the directed subgraph enclosing of them; (ii) compute spectral, positional encodings for the directed subgraph nodes by levaraging truncated SVD; (iii) do the same with HITS algorithms; and (iv) concatenate these encodings and use them as initial inputs in training a directed *GNN* to compute subgraph node encodings, which are pooled at the end to represent the (u, v) pair for the link prediction task. Figure 1 illustrates the overall approach.

(i) Directed Subgraph Extraction: We extract a k-hop directed subgraph $G_k(u, v)$ around the endpoints (u, v) of the potential directed edge $u \mapsto v$. Let $\mathcal{N}^+(i) = \{j|(i, j) \in E\}$ be the set of nodes that i points to, $\mathcal{N}^-(i)$ the set of source nodes targetting i and $\mathcal{N}(i) = \mathcal{N}^+(i) \cup \mathcal{N}^-(i)$ the set of all incident nodes. We recursively define $G_k(u, v) = (V_k(u, v), E_k(u, v))$, by setting $G_0 = (V_0, E_0) = (\{u, v\}, \emptyset)$ and $G_l = (V_l, E_l)$ where $V_l = \bigcup_{j \in V_{l-1}} \mathcal{N}(j)$ and $E_l = \{(i, j) \in E | i, j \in V_l)\}$, for $l = 1, 2, \ldots, k$; (u, v) dropped for notational convenience.

(ii) Direction Aware Positional Encoding Truncated SVD: We compute a variant of truncated SVD of the adjacency matrix $\mathbf{A}_k(u, v) \in \mathbb{R}^{n \times n}$ of subgraph $G_k(u, v)$. More specifically, we compute an approximation $\mathbf{U}_d \in \mathbb{R}^{n \times d}$ of its left singular subspace corresponding to its d top singular values in $\mathbf{\Sigma}_d = diag([\sigma_1, \sigma_2, \ldots, \sigma_d])$, where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_d$. Then the SVD-based encoding for node i in the subgraph will be the corresponding row of $\mathbf{U}_d\mathbf{\Sigma}_d$. This is also the projection of the matrix row of $\mathbf{A}_k(u, v)$ for i on the top d right singular vectors, which are columns of $\mathbf{V}_d$. Let us call this encoding $\texttt{svd[i]} \in \mathbb{R}^d$.

(iii) Direction Aware Positional Encoding: HITS Our second, rank based positional encoding is the combination of the authority value a and the hub value h. We compute the authority and hub scores for all nodes in $G_k(u, v)$ using HITS algorithm [19], which is a popular method for ranking nodes in a directed network. These are scalar centrality values respectively capturing the importance of nodes as targets or as sources of directed links and ranking them accordingly; we organize them in vectors $\mathbf{a}, \mathbf{h} \in \mathbb{R}^n$. $\mathbf{a}$ is computed by iteratively applying $\mathbf{x} \leftarrow \mathbf{A}_k^\top \mathbf{A}_k \mathbf{x}$, initializing with any vector $\mathbf{x}$ and normalizing at each step, where we have set $\mathbf{A}_k(u, v) = \mathbf{A}_k$. Iterating $\mathbf{x} \leftarrow \mathbf{A}_k \mathbf{A}_k^\top \mathbf{x}$ we converge to $\mathbf{h}$ for similar initialization and normalization strategies. Recalling that truncated SVD is $\mathbf{A}_k = \mathbf{U}_d \mathbf{\Sigma}_d \mathbf{V}_d^\top$ it follows that $\mathbf{A}_k \mathbf{A}_k^\top = \mathbf{U}_d \mathbf{\Sigma}_d^2 \mathbf{U}_d^\top$, so the iterative procedure for computing hub scores $\mathbf{h}$ is essentially the power method [20] used for finding the dominant eigenvector of matrix $\mathbf{A}_k \mathbf{A}_k^\top$, which is clearly the first column in left singular vectors $\mathbf{U}_d$. With analogous arguments, authority scores $\mathbf{a}$ is the first column in right singular vectors $\mathbf{V}_d$. Let us concatenate authority and hub scores for a node $i \in V_k$ as $(\mathbf{a}[i], \mathbf{h}[i])$ and call this encoding $\texttt{hits[i]} \in \mathbb{R}^2$.

(iv) Directed GNN-based Link Prediction: For all nodes $i \in V_k(u, v)$, we concatenate HITS and truncated SVD-based positional encodings, $\texttt{pe[i]} = (\texttt{hits[i]} || \texttt{svd[i]}) \in \mathbb{R}^{d+2}$, use them as initial inputs for training a graph neural network (*GNN*), compute new node embeddings and finally pool them to yield a single vector representation for the whole subgraph $G_k(u, v)$.

3.2. Theoretical Analysis

In [15] they prove that, with a valid *labeling trick*, a node-most-expressive *GNN*, defined as giving different representations to non-isomorphic nodes, can learn structural representations of a target node set S. They detail two conditions for a node labeling scheme to actually be a *labeling trick* and a sufficient condition for each of them to hold. In addition, for directed graphs, where the order in the target set S is important they comment that it suffices to define a labeling trick respecting the order of S. The two sufficient conditions read as (i) "the target nodes S have distinct labels from the rest of the nodes, so that S is distinguishable from others" and (ii) "the labeling function is permutation equivariant". In our case $S = (u, v)$ is the ordered set of two nodes, which we want to represent. We recall that the indicator function for S, $\chi_S(i)$, evaluates to 1 if node $i \in S$ and is zero otherwise. $\chi_S(i)$ is the simplest valid labelling trick [15], referred to as *zero-one* labelling trick.

Theorem 3.1 $(\texttt{pe[i]} || \chi_S(i))$ *is a labelling trick.*

Proof: The zero-one component $\chi_S(i)$ in the encoding clearly distinguishes nodes in $S = (u, v)$ from those in $V_k(u, v) - S$, so the first sufficient condition holds. The second condition is also satisfied: the positional encoding component $\texttt{pe[i]}$ is based on SVD and SVD is equivariant; zero-one component $\chi_S(i)$ is also equivariant from [15]. More specifically, for any permutation π in the set Π_n of permutations over n symbols, $\pi \in \Pi_n$ we can define the respective permutation matrix $\mathbf{P}$ with $\mathbf{P}_{(\pi(j),j)} = 1$ and vanishing elsewhere. Then the permuted adjacency matrix corresponding to a permutation of nodes will read: $\mathbf{P}\mathbf{A}_k\mathbf{P}^\top = (\mathbf{P}\mathbf{U}_d)\mathbf{\Sigma}_d(\mathbf{P}\mathbf{V}_d)^\top$ and this means that the node encodings are also permuted under the same π. **Q.E.D.**

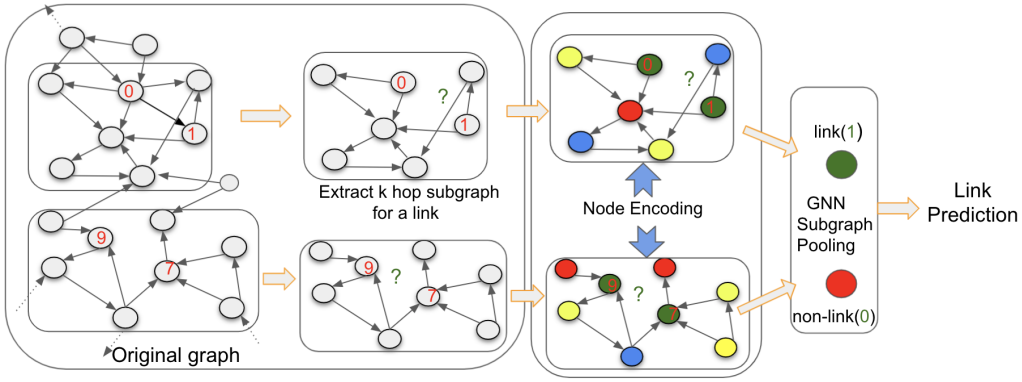

Fig. 1: **Proposed Framework:** First we extract k hop directed subgraph for each directed link in the original graph. Then, we compute truncated SVD and node ranking for each subgraph and encode the nodes as their positional encoding. Finally, the encoded subgraph is provided as input to a *GNN* model to get structural encoding for directed link prediction.

4. EXPERIMENTS

We conducted extensive experiments to evaluate the effectiveness of our positional encoding for directed link prediction on three *GNN* models designed for directed graphs and three other *GNN* models designed for undirected graphs. We also compare the performance with node based variants of the directed *GNN* models. Additionally, we compare our direction aware positional encoding with other structural encodings mainly designed for undirected graphs. We test on WebKB and citation network datasets, which are directed graphs.

4.1. Datasets and Baselines

We use directed WebKB (Cornell, Texas, Wisconsin) datasets [21]; nodes represent web pages, and edges represent hyperlinks between them. We also use the directed citation network (Cora-ML, CiteSeer datasets[22]); nodes represent articles while edges represent citations between them. We used the indegree and outdegree as the node features. The statistics of the datasets are summarized in Table 1.

Baselines: The baseline can be grouped into node based link prediction methods on *GNNs* designed for directed graphs, which are Magnet [10], DGCN [12], DiGraphIB [23] and subgraph based link prediction methods on *GNNs* designed for undirected graphs, which are GCN [16], *GIN* [24], *GraphSage* [25]. *GIN* and *GraphSage* are implemented by adapting the orignal paper [15]. In all baselines, ReLU nonlinearities are used in the output of each hidden layer. To aggregate the neighborhood sum-pooling aggregation is used.

4.2. Experimental Setup

We train three two-layer subgraph based *GNNs* designed for directed graphs for directed link prediction, which are Magnet, DGCN and DiGraphIB. We also train three two-layer subgraph based *GNNs* designed for undirected graphs for directed link prediction. In both cases, we used our proposed truncated SVD and Rank positional encodings.

For all the experiments, we use AUC as our evalution metric. We repeat the experiments 4 times and report the average AUC results. We use 32 as the batch size and train for 500 epochs. We tune learning rate $\eta \in \{10^{-j}, j = 1, 2, 3\}$; $\eta = 10^{-3}$ gives the best performance. We randomly split the edges of each graph into train/validation/test sets at 85/05/10 percent. They are balanced sets of k-hop subgraphs around pairs of nodes u, v which are either connected (positive samples) or not (negative samples). For the Magnet[10] model, we take the optimal hyperparameter value of q, which is provided in the original paper for each dataset.

Table 1: Dataset statistics

Datasets	Nodes	Edges	Features
Cornell	183	295	2
Texas	183	309	2
Wisconsin	251	499	2
CiteSeer	3,312	4,715	2
Cora-ML	2,995	8,416	2

4.3. Results and Discussion

The results of experiments using our truncated SVD and Rank positional encodings are shown in Table 2. The results when truncated SVD and rank positional encoding are used independently are shown in Table 3 and Table 4. In all cases the bold font is used to indicate the best performing models.

Directed v.s. undirected *GNNs* with truncated SVD and Rank positional encodings. In both cases the performance improvesd significantly when truncated SVD and Rank positional encodings are used as input features. The *GNNs* designed for directed graphs, however, are exhibiting better improvement than the *GNNs* designed for undirected graphs. For example, for CoraML and Cornel datasets, Magnet and DGCN directed *GNNs* are the top and second best performing, with 3.13% and 1.28% improvement on Cornel and 5.57% and 2.07% improvement on CoraML over GIN model designed for undirected graphs. Except for Texas dataset, Magnet directed *GNN* is performing the best for all datasets. This indicats that truncated SVD and Rank positional encodings are successfully capturing the directional structural information.

Directed *GNNs* with both truncated SVD and Rank positional encodings v.s. without. We compare the performance of Magnet, DiGraphIB, DGCN directed *GNNs* with v.s. without positional encodings. Table 2 shows clearly that using our encodings improves the performance by a larg margin. This indicates that, though the models are designed to capture the directional structural information, the global directed structural information captured by our positional

Table 2: AUC performance for Directed Link Predition, when both truncated SVD and Rank positional encodings are used.

Model	Cornel	Texas	Wisconsin	Citeseer	CoraML
GCN(SVD + Rank)	86.16 ± 1.52	87.27 ± 2.77	82.13 ± 2.26	87.97 ± 0.57	88.15 ± 0.73
GIN(SVD + Rank)	88.01 ± 2.75	**90.72 ±2.24**	90.72 ± 1.68	89.12 ± 0.57	88.28 ± 0.25
SAGE(SVD + Rank)	88.24 ± 3.2	88.88 ± 2.72	89.13 ± 2.27	87.47 ± 1.97	87.92 ± 0.23
DGCN	82.24 ± 3.47	84.01 ± 1.67	82.89 ± 1.74	82.02 ± 0.8	82.92 ± 0.37
DiGraphIB	81.93 ± 1.65	82.72 ± 1.58	81.67 ± 1.74	84.89 ± 0.76	85.27 ± 0.62
Magnet	83.32 ± 2.71	83.01 ± 1.72	84.7 ± 1.92	86.72 ± 1.42	85.77 ± 0.42
DGCN(SVD + Rank)	89.24 ± 2.47	87.04 ± 1.92	87.21 ± 1.74	88.75 ± 0.66	90.21 ± 1.37
DiGraphIB(SVD + Rank)	87.58 ± 2.17	87.01 ± 2.87	88.11 ± 2.74	89.82 ± 0.68	89.2 ± 0.58
Magnet(SVD + Rank)	**91.98 ± 1.62**	89.98 ± 2.91	**90.82 ± 1.08**	**91.66 ± 0.81**	**93.85 ± 1.27**

Table 3: AUC performance for Directed Link Predition, when only truncated SVD positional encoding is used.

Model	Cornel	Texas	Wisconsin	Citeseer	CoraML
GCN(SVD)	82.03 ± 1.52	82.57 ± 1.7	77.33 ± 1.8	83.87 ± 1.47	82.87 ± 1.7
GIN(SVD)	84.25 ± 2.1	**91.35 ± 3..87**	81.32 ± 2.76	85.58 ± 0.74	87.92 ± 0.42
SAGE(SVD)	83.3 ± 2.77	82.21 ± 2.5	81.2 9 ± 1.92	82 ± 0.52	84.57 ± 0.75
DGCN(SVD)	84.86 ± 1.6	80.27 ± 1.6	85.65 ± 1.78	81.29 ± 0.87	83.47 ± 1.21
DiGraphIB(SVD)	85.86 ± 2.12	84.27 ± 2.63	87.65 ± 1.78	85.29 ± 0.87	88.87 ± 0.77
Magnet(SVD)	**90.14 ± 2.7**	89.69 ± 1.9	**91.42 ± 2.1**	**88.23 ± 1.24**	**89.08 ± 1.72**

Table 4: AUC performance for Directed Link Predition, when only Rank positional encodings is used.

Model	Cornel	Texas	Wisconsin	Citeseer	CoraML
GCN(Rank)	79.03±1.52	82.57 ± 1.7	77.33 ± 1.8	83.87 ± 1.47	82.87 ± 1.7
GIN(Rank)	**86.15 ± 1.31**	83.78 ± 2.1	82.56 ± 1.76	87.52 ± 0.42	86.43 ± 0.78
SAGE(Rank)	82.28 ± 1.97	83.79 ± 2.07	81.72 ± 1.34	87.18 ± 0.48	85.51 ± 0.28
DGCN(Rank)	82.72 ± 2.87	83.22 ± 1.73	83.72 ± 2.34	87.18 ± 0.48	83.42 ± 0.52
DiGraphIB(Rank)	83.04 ± 1.07	81.22 ± 3.34	**83.92 ± 1.34**	88.18 ± 0.28	87.23 ± 0.72
Magnet(Rank)	85.12 ± 1.89	**87.22 ± 1.47**	83.08 ± 0.87	**89.18 ± 0.48**	**90.08 ± 0.65**

encodings is still important.

Comparison truncated SVD and Rank with DRNL positional encodings. When we use DRNL, which is designed for undirected graphs, on directed graphs, the results are the worst. This can be seen by comparing the results in Table 5 and Table 2. This implies that the existing positional encodings for undirected graph are not suitable for capturing the global directed structural information.

4.4. Ablation Studies

We conduct ablation studies to examine to what extent truncated SVD and Rank can independently capture the directed structural information.

Truncated SVD as positional encoding. As we can see in Table 3, the performace with trancated SVD alone is still suprisingly good. The directed *GNN* models are still performing better on three out of five directed datasets than the undirected *GNN* models. This demostrates that trancated SVD alone as positional encoding is quite effective for directed graphs.

Rank as positional encoding. Table 4 shows the effect of Rank when used as positional encoding alone. The results are similar to the case with truncated SVD alone shown in Table 3. Also we observe that the *GNNs* designed for directed graphs are exhibiting better improvement than the *GNNs* designed for undirected graphs. This shows that Rank positional encoding also is capable of capturing global directed structural information of a directed graph. Comparison between Tables 2, 3 and 4 shows that combining truncated SVD and Rank, which are both variants of SVD, imporves the performance further.

Table 5: AUC performance for Directed Link Predition, when Distance Radius Node Label(DRNL) encoding is used.

Model	Cornel	Texas	Wisconsin	Citeseer	CoraML
GCN(DRNL)	79.03	**81.27**	80.72	80.57	81.05
GIN(DRNL)	**81.07**	79.50	81.62	81.67	**82.02**
SAGE(DRNL)	79.92	75.76	**82.02**	**82.12**	81.17

5. CONCLUSIONS AND FUTURE WORK

We combine specially designed, direction-aware, positional encodings of enclosing subgraph nodes with directed *GNN* learning in order to generate provably high-performance encodings for the link prediction problem in directed graphs. Experiments demonstrate the sustained efficacy of our approach. Predicting edges *and* their direction can be critical in graph applications and is a special case of the more general problem of *co-encoding ordered node sets*, which we plan to research within the context of higher-order representations (graph-products, tensorial adjacencies).

6. REFERENCES

[1] Muhan Zhang and Yixin Chen, "Link prediction based on graph neural networks," *Advances in neural information processing systems*, vol. 31, 2018.

[2] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin, "Graph neural networks for social recommendation," in *The world wide web conference*, 2019, pp. 417–426.

[3] Xiao-Meng Zhang, Li Liang, Lin Liu, and Ming-Jing Tang, "Graph neural networks and their current applications in bioinformatics," *Frontiers in genetics*, vol. 12, 2021.

[4] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui, "Graph neural networks in recommender systems: a survey," *ACM Computing Surveys (CSUR)*, 2020.

[5] David Jaime Tena Cucala, Bernardo Cuenca Grau, Egor V Kostylev, and Boris Motik, "Explainable gnn-based models over knowledge graphs," in *International Conference on Learning Representations*, 2021.

[6] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl, "Neural message passing for quantum chemistry," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1263–1272.

[7] Thomas N Kipf and Max Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[8] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen, "An end-to-end deep learning architecture for graph classification," in *Proceedings of the AAAI conference on artificial intelligence*, 2018, vol. 32.

[9] Bing Yu, Haoteng Yin, and Zhanxing Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," *arXiv preprint arXiv:1709.04875*, 2017.

[10] Xitong Zhang, Yixuan He, Nathan Brugnone, Michael Perlmutter, and Matthew Hirn, "Magnet: A neural network for directed graphs," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27003–27015, 2021.

[11] Zekun Tong, Yuxuan Liang, Changsheng Sun, Xinke Li, David Rosenblum, and Andrew Lim, "Digraph inception convolutional networks," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[12] Zekun Tong, Yuxuan Liang, Changsheng Sun, David S Rosenblum, and Andrew Lim, "Directed graph convolutional network," *arXiv preprint arXiv:2004.13970*, 2020.

[13] Balasubramaniam Srinivasan and Bruno Ribeiro, "On the equivalence between positional node embeddings and structural graph representations," *arXiv preprint arXiv:1910.00452*, 2019.

[14] Muhan Zhang and Yixin Chen, "Weisfeiler-lehman neural machine for link prediction," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 575–583.

[15] Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin, "Labeling trick: A theory of using graph neural networks for multi-node representation learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9061–9073, 2021.

[16] Thomas N Kipf and Max Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[17] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson, "Graph neural networks with learnable structural and positional representations," *arXiv preprint arXiv:2110.07875*, 2021.

[18] Jiaxuan You, Rex Ying, and Jure Leskovec, "Position-aware graph neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 7134–7143.

[19] Jon M Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM (JACM)*, vol. 46, no. 5, pp. 604–632, 1999.

[20] Gene H Golub and Charles F Van Loan, *Matrix Computations*, vol. 3, JHU Press, 2013.

[21] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang, "Geom-gcn: Geometric graph convolutional networks," *arXiv preprint arXiv:2002.05287*, 2020.

[22] Aleksandar Bojchevski and Stephan Günnemann, "Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking," *arXiv preprint arXiv:1707.03815*, 2017.

[23] William R Palmer and Tian Zheng, "Spectral clustering for directed networks," in *International Conference on Complex Networks and Their Applications*. Springer, 2020, pp. 87–99.

[24] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka, "How powerful are graph neural networks?," in *International Conference on Learning Representations*, 2018.

[25] Will Hamilton, Zhitao Ying, and Jure Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.