Multi-hop Attention Graph Neural Networks

Guangtao Wang¹,*, Rex Ying², Jing Huang¹ and Jure Leskovec²
¹JD AI Research
²Computer Science, Stanford University

guangtao.wang@jd.com, rexying@stanford.edu, jing.huang@jd.com, jure@cs.stanford.edu

Abstract

Self-attention mechanism in graph neural networks (GNNs) led to state-of-the-art performance on many graph representation learning tasks. Currently, at every layer, attention is computed between connected pairs of nodes and depends solely on the representation of the two nodes. However, such attention mechanism does not account for nodes that are not directly connected but provide important network context. Here we propose Multi-hop Attention Graph Neural Network (MAGNA), a principled way to incorporate multi-hop context information into every layer of attention computation. MAGNA diffuses the attention scores across the network, which increases the “receptive field” for every layer of the GNN. Unlike previous approaches, MAGNA uses a diffusion prior on attention values, to efficiently account for all paths between the pair of disconnected nodes. We demonstrate in theory and experiments that MAGNA captures large-scale structural information in every layer, and has a low-pass effect that eliminates noisy high-frequency information from graph data. Experimental results on node classification as well as the knowledge graph completion benchmarks show that MAGNA achieves state-of-the-art results: MAGNA achieves up to 5.7% relative error reduction over the previous state-of-the-art on Cora, Citeeseer, and Pubmed. MAGNA also obtains the best performance on a large-scale Open Graph Benchmark dataset. On knowledge graph completion MAGNA advances state-of-the-art on WN18RR and FB15k-237 across four different performance metrics.

1 Introduction

Self-attention [Bahdanau et al., 2015; Vaswani et al., 2017] has pushed the state-of-the-art in many domains including graph presentation learning [Devlin et al., 2019]. Graph Attention Network (GAT) [Veličković et al., 2018] and related models [Li et al., 2018; Wang et al., 2019a; Liu et al., 2019; Oono and Suzuki, 2020] developed attention mechanism for Graph Neural Networks (GNNs), which compute attention scores between nodes connected by an edge, allowing the model to attend to messages of node’s neighbors.

However, such attention computation on pairs of nodes connected by edges implies that a node can only attend to its immediate neighbors to compute its (next layer) representation. This implies that receptive field of a single GNN layer is restricted to one-hop network neighborhoods. Although stacking multiple GAT layers could in principle enlarge the receptive field and learn non-neighboring interactions, such deep GAT architectures suffer from the over-smoothing problem [Wang et al., 2019a; Liu et al., 2019; Oono and Suzuki, 2020] and do not perform well. Furthermore, edge attention in a single GAT layer is based solely on representations of the two nodes at the edge endpoints, and does not depend on their graph neighborhood context. In other words, the one-hop attention mechanism in GAT’s limits their ability to explore the relationship between the broader graph structure. While previous works [Xu et al., 2018; Klicpera et al., 2019b] have shown advantages in performing multi-hop message-passing in a single layer, these ap-
proaches are not graph-attention based. Therefore, incorporating multi-hop neighboring context into the attention computation in graph neural networks remains to be explored.

Here we present Multi-hop Attention Graph Neural Network (MAGNA), an effective multi-hop self-attention mechanism for graph structured data. MAGNA uses a novel graph attention diffusion layer (Figure 1), where we first compute attention weights on edges (represented by solid arrows), and then compute self-attention weights (dotted arrows) between disconnected pairs of nodes through an attention diffusion process using the attention weights on the edges.

Our model has two main advantages: (1) MAGNA captures long-range interactions between nodes that are not directly connected but may be multiple hops away. Thus the model enables effective long-range message passing, from important nodes multiple hops away. (2) The attention computation in MAGNA is context-dependent. The attention value in GATs [Veličković et al., 2018] only depends on node representations of the previous layer, and is zero between non-connected pairs of nodes. In contrast, for any pair of nodes within a chosen multi-hop neighborhood, MAGNA computes attention by aggregating the attention scores over all the possible paths (length \( \geq 1 \)) connecting the two nodes.

Mathematically we show that MAGNA places a Personalized Page Rank (PPR) prior on the attention values. We further apply spectral graph analysis to show that MAGNA emphasizes on large-scale graph structure and lowering high-frequency noise in graphs. Specifically, MAGNA enlarges the lower Laplacian eigen-values, which correspond to the large-scale structure in the graph, and suppresses the higher Laplacian eigen-values which correspond to more noisy and fine-grained information in the graph.

We experiment on standard datasets for semi-supervised node classification as well as knowledge graph completion. Experiments show that MAGNA achieves state-of-the-art results: MAGNA achieves up to 5.7% relative error reduction over previous state-of-the-art on Cora, Citeseer, and Pubmed. MAGNA also obtains better performance on a large-scale Open Graph Benchmark dataset. On knowledge graph completion, MAGNA advances state-of-the-art on WN18RR and FB15k-237 across four metrics, with the largest gain of 7.1% in the metric of Hit at 1.

Furthermore, we show that MAGNA with just 3 layers and 6 hop wide attention per layer significantly out-performs GAT with 18 layers, even though both architectures have the same receptive field. Moreover, our ablation study reveals the synergistic effect of the essential components of MAGNA, including layer normalization and multi-hop diffused attention. We further observe that compared to GAT, the attention values learned by MAGNA have higher diversity, indicating the ability to better pay attention to important nodes.

## 2 Multi-hop Attention Graph Neural Network (MAGNA)

We first discuss the background and explain the novel multi-hop attention diffusion module and the MAGNA architecture.

### 2.1 Preliminaries

Let \( G = (V, E) \) be a given graph, where \( V \) is the set of \( N_v \) nodes, \( E \subseteq V \times V \) is the set of \( N_e \) edges connecting \( M \) pairs of nodes in \( V \). Each node \( v \in V \) and each edge \( e \in E \) are associated with their type mapping functions: \( \phi : V \rightarrow T \) and \( \psi : E \rightarrow R \). Here \( T \) and \( R \) denote the sets of node types (labels) and edge/relationship types. Our framework supports learning on heterogeneous graphs with multiple elements in \( R \).

A general Graph Neural Network (GNN) approach learns an embedding that maps nodes and/or edges types into a continuous vector space. Let \( X \in \mathbb{R}^{N_v \times d_v} \) and \( R \in \mathbb{R}^{N_r \times d_r} \) be the node embedding and edge/relationship type embedding, where \( N_v = |V| \), \( N_r = |R| \). \( d_v \) and \( d_r \) represent the embedding dimension of node and edge/relationship types, each row \( x_i = X[i,:) \) represents the embedding of node \( v_i \) \( (1 \leq i \leq N_v) \), and \( r_j = R[j,:) \) represents the embedding of relation \( r_j \) \( (1 \leq j \leq N_r) \). MAGNA builds on GNNs, while bringing together the benefits of Graph Attention and Diffusion techniques.

### 2.2 Multi-hop Attention Diffusion

We first introduce attention diffusion to compute the multi-hop attention directly, which operates on the MAGNA’s attention scores at each layer. The input to the attention diffusion operator is a set of triples \((v_i, r_k, v_j)\), where \( v_i, v_j \) are nodes and \( r_k \) is the edge type. MAGNA first computes the attention scores on all edges. The attention diffusion module then computes the attention values between pairs of nodes that are not directly connected by an edge, based on the edge attention scores, via a diffusion process. The attention diffusion module can then be used as a component in MAGNA architecture, which we will further elaborate in Section 2.3.

#### Edge Attention Computation

At each layer \( l \), a vector message is computed for each triple \((v_i, r_k, v_j)\). To compute the representation of \( v_j \) at layer \( l+1 \), all messages from triples incident to \( v_j \) are aggregated into a single message, which is then used to update \( v_j^{l+1} \).

In the first stage, the attention score \( s \) of an edge \((v_i, r_k, v_j)\) is computed by the following:

\[
\mathbf{s}_{i,k,j}^{(l)} = \delta(v_i^{(l)} \tanh(W_h^{(l)} h_i^{(l)} || W_t^{(l)} h_j^{(l)} || W_r^{(l)} r_k^{(l)}))
\]

where \( \delta = \text{LeakyReLU} \), \( W_h^{(l)}, W_t^{(l)}, W_r^{(l)} \in \mathbb{R}^{d_v \times d_v} \), \( W_t^{(l)} \in \mathbb{R}^{d_v \times d_r} \), and \( v_i^{(l)}, r_i^{(l)} \in \mathbb{R}^{d_v \times d_r} \) are the trainable weights shared by \( l \)-th layer. \( h_i^{(l)} \in \mathbb{R}^{d_v} \) represents the embedding of node \( i \) at \( l \)-th layer, and \( h_j^{(l)} = a_i \), \( r_k^{(l)} \) is the trainable relation embedding of the \( k \)-th relation type \( r_k \) \((1 \leq k \leq N_r)\), and \( a || b \) denotes concatenation of embedding vectors \( a \) and \( b \). For graphs with no relation type, we treat as a degenerate categorical distribution with 1 category.

Applying Eq. 1 on each edge of the graph \( G \), we obtain an attention score matrix \( \mathbf{S}^{(l)} \):

\[
\mathbf{S}^{(l)}_{i,j} = \begin{cases} 
\mathbf{s}_{i,k,j}^{(l)}, & \text{if } (v_i, r_k, v_j) \text{ appears in } G \\
-\infty, & \text{otherwise}
\end{cases}
\]

\[1\] In this case, we can view that there is only one “pseudo” relation type (category), i.e., \( N_r = 1 \).
Subsequently we obtain the attention matrix $A^{(l)}$ by performing row-wised softmax over the score matrix $S^{(l)}$: $A^{(l)} = \text{softmax}(S^{(l)})$. $A_{ij}^{(l)}$ denotes the attention value at layer $l$ when aggregating message from node $j$ to node $i$.

**Attention Diffusion for Multi-hop Neighbors** In the second stage, we further enable attention between nodes that are not directly connected in the network. We achieve this via the following attention diffusion procedure. The procedure computes the attention scores of multi-hop neighbors via graph diffusion based on the powers of the 1-hop attention matrix $A$:

$$A = \sum_{i=0}^{\infty} \theta_i A^i \quad \text{where} \quad \sum_{i=0}^{\infty} \theta_i = 1 \quad \text{and} \quad \theta_i > 0 \quad (3)$$

where $\theta_i$ is the attention decay factor and $\theta_i > \theta_{i+1}$. The powers of attention matrix, $A^i$, give us the number of relation paths from node $h$ to node $t$ of length up to $i$, increasing the receptive field of the attention (Figure 1). Importantly, the mechanism allows the attention between two nodes to not only depend on their previous layer representations, but also taking into account of the paths between the nodes, effectively creating attention shortcuts between nodes that are not directly connected (Figure 1). Attention through each path is also weighted differently, depending on $\theta$ and the path length.

In our implementation we utilize the geometric distribution $\theta_i = \alpha (1 - \alpha)^i$, where $\alpha \in (0, 1]$. The choice is based on the inductive bias that nodes further away should be weighted less in message aggregation, and nodes with different relation path lengths to the target node are sequentially weighted in an independent manner. In addition, notice that if we define $\theta_0 = \alpha \in (0, 1]$, $A^0 = I$, then Eq. 3 gives the Personalized Page Rank (PPR) procedure on the graph with the attention matrix $A$ and teleport probability $\alpha$. Hence the diffused attention weights, $A_{ij}$, can be seen as the influence of node $j$ to node $i$. We further elaborate the significance of this observation in Section 4.3.

We can also view $A_{ij}$ as the attention value of node $j$ to $i$ since $\sum_{j=1}^{N_n} A_{ij} = 1$.\(^2\) We then define the graph attention diffusion based feature aggregation as

$$\text{AttDiff}(G, H^{(l)}, \Theta) = A H^{(l)}, \quad (4)$$

where $\Theta$ is the set of parameters for computing attention. Thanks to the diffusion process defined in Eq. 3, MAGNA uses the same number of parameters as if we were only computing attention between nodes connected via edges. This ensures runtime efficiency (refer to Appendix\(^3\) A for complexity analysis) and model generalization.

**Approximate Computation for Attention Diffusion** For large graphs computing the exact attention diffusion matrix $A$ using Eq. 3 may be prohibitively expensive, due to computing the powers of the attention matrix [Klicpera et al., 2019a]. To resolve this bottleneck, we proceed as follows: Let $H^{(l)}$ be the input entity embedding of the $l$-th layer ($H^{(0)} = X$)

\(^2\)Obtained by the definition $A^{(l)} = \text{softmax}(S^{(l)})$ and Eq. 3.

\(^3\)Appendix can be found via https://arxiv.org/abs/2009.14332
where $\|$ denotes concatenation and $\Theta_i$ are the parameters in Eq. 1 for the $i$-th head ($1 \leq i \leq M$). $W_o$ represents a parameter matrix, and $LN = \text{LayerNorm}$. Since we calculate the attention diffusion in a recursive way in Eq. 5, we add layer normalization which helpful to stabilize the recurrent computation procedure [Ba et al., 2016].

**Deep Aggregation** Moreover our MAGNA block contains a fully connected feed-forward sub-layer, which consists of a two-layer feed-forward network. We also add the layer normalization and residual connection in both sub-layers, allowing for a more expressive aggregation step for each block [Xiong et al., 2020]:

$$H^{(t+1)} = \tilde{H}^{(t)} + H^{(t)}$$

$$H^{(t+1)} = W_2^{(t)} \text{ReLU} \left( W_1^{(t)} \text{LN} \left( \tilde{H}^{(t+1)} \right) \right) + \tilde{H}^{(t+1)}$$ (7)

**MAGNA generalizes GAT** MAGNA extends GAT via the diffusion process. The feature aggregation in GAT is $H^{(t+1)} = \sigma \left( A H^{(t)} W^{(t)} \right)$, where $\sigma$ represents the activation function. We can divide GAT layer into two components as follows:

$$H^{(t+1)} = \sigma \left( A H^{(t)} W^{(t)} \right)_{(2)}$$ (8)

In component (1), MAGNA removes the restriction of attending to direct neighbors, without requiring additional parameters as $A$ is induced from $A$. For component (2) MAGNA uses layer normalization and deep aggregation to achieve higher expressive power compared to elu nonlinearity in GAT.

3 Analysis of Graph Attention Diffusion

In this section, we investigate the benefits of MAGNA from the viewpoint of discrete signal processing on graphs [Sandryhaila and Moura, 2013]. Our first result demonstrates that MAGNA can better capture large-scale structural information. Our second result explores the relation between MAGNA and Personalized PageRank (PRR).

3.1 Spectral Properties of Graph Attention Diffusion

We view the attention matrix $A$ of GAT, and $A$ of MAGNA as weighted adjacency matrices, and apply Graph Fourier transform and spectral analysis (details in Appendix) to show the effect of MAGNA as a graph low-pass filter, being able to more effectively capture large-scale structure in graphs. By Eq. 3, the sum of each row of either $A$ or $A$ is 1. Hence the normalized graph Laplacians are $L_{sym} = I - A$ and $L_{sym} = I - A$ for $A$ and $A$ respectively. We can get the following proposition:

**Proposition 2.** Let $\lambda_i^g$ and $\lambda_i^g$ be the $i$-th eigenevalues of $L_{sym}$ and $L_{sym}$.

$$\frac{\hat{\lambda}_i^g}{\lambda_i^g} = \frac{1 - \frac{\alpha}{\lambda_i^g}}{\lambda_i^g} = \frac{1 - \frac{\alpha}{\lambda_i^g}}{\lambda_i^g} \frac{\lambda_i^g}{1 - \frac{\alpha}{\lambda_i^g}}$$ (9)

Refer to Appendix for the proof. We additionally have $\lambda_i^g \in [0, 2]$ (proved by [Ng et al., 2002]). Eq. 9 shows that when $\lambda_i^g$ is small such that $\frac{1}{\lambda_i^g} + \lambda_i^g < 1$, then $\hat{\lambda}_i^g > \lambda_i^g$, whereas for large $\lambda_i^g$, $\hat{\lambda}_i^g < \lambda_i^g$. This relation indicates that the use of $A$ increases smaller eigenvalues and decreases larger eigenvalues. See Section 4.3 for its empirical evidence. The low-pass effect increases with smaller $\alpha$.

The eigenvalues of the low-frequency signals describe the large-scale structure in the graph [Ng et al., 2002] and have been shown to be crucial in graph tasks [Klicpera et al., 2019b]. As $\lambda_i^g \in [0, 2] \text{[Ng et al., 2002]}$ and $\frac{1}{\lambda_i^g} > 0$, the reciprocal format in Eq. 9 will amplify the ratio of lower eigenvalues to the sum of all eigenvalues. In contrast, high eigenvalues corresponding to noise are suppressed.

3.2 Personalized PageRank Meets Graph Attention Diffusion

We can also view the attention matrix $A$ as a random walk matrix on graph $G$ since $\sum_{k=1}^{N} A_{i,j} = 1$ and $A_{i,j} > 0$. If we perform Personalized PageRank (PPR) with parameter $\alpha \in (0, 1]$ on $G$ with transition matrix $A$, the fully Personalized PageRank [Lofgren, 2015] is defined as:

$$A_{ppr} = \alpha \left( I - (1 - \alpha)A \right)^{-1}$$ (10)

Using the power series expansion for the matrix inverse, we obtain

$$A_{ppr} = \alpha \sum_{i=0}^{\infty} (1 - \alpha)^i A^i = \sum_{i=0}^{\infty} \alpha (1 - \alpha)^i A^i$$ (11)

Comparing to the diffusion Equation 3 with $\theta_i = \alpha (1 - \alpha)^i$, we have the following proposition.

**Proposition 3.** Graph attention diffusion defines a personalized page rank with parameter $\alpha \in (0, 1]$ on $G$ with transition matrix $A$, i.e., $A = A_{ppr}$.

The parameter $\alpha$ in MAGNA is equivalent to the teleport probability of PPR. PPR provides a good relevance score between nodes in a weighted graph (the weights from the attention matrix $A$). In summary, MAGNA places a PPR prior over node pairwise attention scores: the diffused attention between node $i$ and $j$ depends on the attention scores on the edges of all paths between $i$ and $j$.

4 Experiments

We evaluate MAGNA on two classical tasks: (1) on node classification we achieve an average of 5.7% relative error reduction; (2) on knowledge graph completion we achieve 7.1% relative improvement in the Hit@1 metric.

4.1 Task 1: Node Classification

**Datasets** We employ four benchmark datasets for node classification: (1) standard citation network benchmarks Cora, Citeseer and Pubmed [Sen et al., 2008; Kipf and Welling, 2016]; and (2) a benchmark dataset ogbn-arxiv on
170k nodes and 1.2m edges from the Open Graph Benchmark [Weihuia Hu, 2020]. We follow the standard data splits for all datasets. Further information about these datasets is summarized in the Appendix.

**Baselines** We compare against a comprehensive suite of state-of-the-art GNN methods including: GCNs [Kipf and Welling, 2016], Chebyshev filter based GCNs [Defferrard et al., 2016], DualGCN [Zhuang and Ma, 2018], JKNet [Xu et al., 2018], LGCN [Gao et al., 2016], Diff-GCN [Klicpera et al., 2019b], APPNP [Klicpera et al., 2019a], g-U-Nets [Gao and Ji, 2019], DualGCN [Zhuang and Ma, 2018], JKNet [Xu et al., 2018] and GAT [Veličković et al., 2018].

**Ablation Study** We report (Table 1) the model performance after removing each component of MAGNA (layer normalization, attention diffusion and feed forward layers) from every MAGNA layer. Note that the model is equivalent to GAT without these three components. We observe that diffusion and layer normalization play a crucial role in improving the node classification performance for all datasets. Since MAGNA computes attention diffusion in a recursive manner, layer normalization is crucial in ensuring training stability [Ba et al., 2016]. Meanwhile, comparing to GAT (see the next-to-last row of Table 1), attention diffusion allows multi-hop attention in every layer to benefit node classification.

<table>
<thead>
<tr>
<th>Models</th>
<th>Cora</th>
<th>Citeseer</th>
<th>Pubmed</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN [Kipf and Welling, 2016]</td>
<td>81.5</td>
<td>70.5</td>
<td>79.0</td>
</tr>
<tr>
<td>Cheby [Defferrard et al., 2016]</td>
<td>81.2</td>
<td>69.8</td>
<td>74.4</td>
</tr>
<tr>
<td>DualGCN [Zhuang and Ma, 2018]</td>
<td>83.5</td>
<td>72.6</td>
<td>80.0</td>
</tr>
<tr>
<td>JKNet [Xu et al., 2018]</td>
<td>81.1</td>
<td>69.8</td>
<td>78.1</td>
</tr>
<tr>
<td>LGCN [Gao et al., 2018]</td>
<td>83.3 ± 0.5</td>
<td>73.0 ± 0.6</td>
<td>79.5 ± 0.2</td>
</tr>
<tr>
<td>Diff-GCN [Klicpera et al., 2019b]</td>
<td>83.6 ± 0.2</td>
<td>73.4 ± 0.3</td>
<td>79.6 ± 0.4</td>
</tr>
<tr>
<td>APPNP [Klicpera et al., 2019a]</td>
<td>84.3 ± 0.2</td>
<td>71.1 ± 0.4</td>
<td>79.7 ± 0.3</td>
</tr>
<tr>
<td>g-U-Nets [Gao and Ji, 2019]</td>
<td>84.4 ± 0.6</td>
<td>73.2 ± 0.5</td>
<td>79.6 ± 0.2</td>
</tr>
<tr>
<td>GAT [Veličković et al., 2018]</td>
<td>83.0 ± 0.7</td>
<td>72.5 ± 0.7</td>
<td>79.0 ± 0.3</td>
</tr>
<tr>
<td>MAGNA</td>
<td>85.4 ± 0.6</td>
<td>75.7 ± 0.5</td>
<td>81.4 ± 0.2</td>
</tr>
</tbody>
</table>

Table 1: Node classification accuracy on Cora, Citeseer, Pubmed. MAGNA achieves state-of-the-art.

### 4.4 Task 2: Knowledge Graph Completion

**Datasets** We evaluate MAGNA on standard benchmark knowledge graphs: WN18RR [Dettmers et al., 2018] and FB15K-237 [Toutanova and Chen, 2015]. See the statistics of these KGs in Appendix.

**Baselines** We compare MAGNA with state-of-the-art baselines, including (1) translational distance based models: TransE [Bordes et al., 2013] and its latest extension RotatE [Sun et al., 2019], OTE [Tang et al., 2020] and ROTH [Chami et al., 2020]; (2) semantic matching based models: ComplEx [Trouillon et al., 2016], QuatE [Zhang et al., 2019], CoKE [Wang et al., 2019b], ConvE [Dettmers et al., 2018], DistMult [Yang et al., 2015], TuckER [Balazevic et al., 2019] and AutoSF [Zhang et al., 2020]; (3) GNN-based models: R-GCN [Schlichtkrull et al., 2018], SACN [Shang et al., 2019] and A2N [Bansal et al., 2019].

**Experimental Setup** We use the multi-layer MAGNA as encoder for both FB15k-237 and WN18RR. We randomly initialize the entity embedding and relation embedding as the input of the encoders, and set the dimensionality of the initialized entity RELATION vector as 100 used in DistMult [Yang et al., 2015]. We select other MAGNA model hyper-parameters, including number of layers, hidden dimension, hop number, top-k, learning rate, hop number, teleport probability $\alpha$ and dropout ratios (see the settings of these parameter in Appendix), by a random search during the training.

**Training procedure** We use the standard training procedure used in previous KG embedding models [Balazevic et al., 2019; Dettmers et al., 2018] (Appendix for details). We follow an encoder-decoder framework: The encoder applies the proposed MAGNA model to compute the entity embeddings. The decoder makes link prediction given the embeddings. To show the power of MAGNA, we employ a simple decoder DistMult [Yang et al., 2015].

**Evaluation** We use the standard split for the benchmarks, and the standard testing procedure of predicting tail (head) entity given the head (tail) entity and relation type. We exactly follow the evaluation used by all previous works, namely the Mean Reciprocal Rank (MRR), Mean Rank (MR), and hit at $K$ ($H@K$). See Appendix for a detailed description of this standard setup.

**Results** MAGNA achieves new state-of-the-art in knowledge graph completion on all four metrics (Table 3), MAGNA compares favourably to both the most recent shallow embedding methods (QuatE), and deep embedding methods (SACN). Note that with the same decoder (DistMult), MAGNA using its own embeddings achieves drastic improvements over using the corresponding DistMult embeddings.

### 4.3 MAGNA Model Analysis

Here we present (1) spectral analysis results, (2) robustness to hyper-parameter changes, and (3) attention distribution analysis to show the strengths of MAGNA.

**Spectral Analysis: Why MAGNA works for node classification?** We compute the eigenvalues of the graph Laplacian

---
of the attention matrix $A$, $\lambda_i^g$, and compare to that of the diffused matrix $A$, $\lambda_i^s$. Figure 3 (a) shows the ratio $\lambda_i^g/\lambda_i^s$ on the Cora dataset. Low eigenvalues corresponding to large-scale structure in the graph are amplified (up to a factor of 8), while high eigenvalues corresponding to eigenvectors with noisy information are suppressed [Klicpera et al., 2019b].

**MAGNA Model Depth** Here we conduct experiments by varying the number of GCN, Diffusion-GCN (PPR based) GAT and our MAGNA layers to be 3, 6, 12, 18 and 24 for node classification on Cora. Results in Fig. 3 (b) show that deep GCN, Diffusion-GCN and GAT (even with residual connection) suffer from degrading performance, due to the over-smoothing problem [Li et al., 2018; Wang et al., 2019a]. In contrast, the MAGNA model achieves consistent best results even with 18 layers, making deep MAGNA model robust and expressive. Notice that GAT with 18 layers cannot out-perform MAGNA with 3 layers and $K=6$ hops, although they have the same receptive field.

**Effect of $K$ and $\alpha$** Figs. 3 (c) and (d) report the effect of hop number $K$ and teleport probability $\alpha$ on model performance. We observe significant increase in performance when considering multi-hop neighbors information ($K > 1$). However, increasing the hop number $K$ has a diminishing returns, for $K \geq 6$. Moreover, we find that the optimal $K$ is correlated with the largest node average shortest path distance (e.g., 5.27 for Cora). This provides a guideline for choosing the best $K$.

We also observe that the accuracy drops significantly for larger $\alpha > 0.25$. This is because small $\alpha$ increases the low-pass effect (Fig. 3 (a)). However, $\alpha$ being too small causes the model to only focus on the most large-scale graph structure and have lower performance.

**Attention Distribution** Last we also analyze the learned attention scores of GAT and MAGNA. We first define a discrepancy metric over the attention matrix $A$ for node $v_i$ as $\Delta_i = \|A_{v_i}-U\|_{\text{degree}(v_i)}$ [Shanthamallu et al., 2020], where $U_i$ is the uniform distribution score for the node $v_i$. $\Delta_i$ gives a measure of how much the learnt attention deviates from an uninformative uniform distribution. Large $\Delta_i$ indicates more meaningful attention scores. Fig. 4 shows the distribution of the discrepancy metric of the attention matrix of the 1st head w.r.t. the first layer of MAGNA and GAT. Observe that at-
tention scores learned in MAGNA have much larger discrepancy. This shows that MAGNA is more powerful than GAT in distinguishing important nodes and assigns attention scores accordingly.

5 Related Work

Our proposed MAGNA belongs to the family of Graph Neural Network (GNN) models [Battaglia et al., 2018; Wu et al., 2020; Kipf and Welling, 2016; Hamilton et al., 2017], while taking advantage of graph attention and diffusion techniques.

Graph Attention Neural Networks (GATs) generalize attention operation to graph data. GATs allow for assigning different importance to nodes of the same neighborhood at the feature aggregation step [Veličković et al., 2018]. Based on such framework, different attention-based GNNs have been proposed, including GaAN [Zhang et al., 2018], AGNN [Thekumparampil et al., 2018], GeniePath [Liu et al., 2019]. However, these models only consider direct neighbors for each layer of feature aggregation, and suffer from over-smoothing when they go deep [Wang et al., 2019a].

Diffusion based Graph Neural Network Recently Graph Diffusion Convolution [Klicpera et al., 2019b; Klicpera et al., 2019a] proposes to aggregate information from a larger (multi-hop) neighborhood at each layer, by sparsifying a generalized form of graph diffusion. This idea was also explored in [Liao et al., 2019; Luan et al., 2019; Xhonneux et al., 2020; Klicpera et al., 2019a] for multi-scale Graph Convolutional Networks. However, these methods do not incorporate attention mechanism which is crucial to model performance, and do not make use of edge embeddings (e.g., Knowledge graph) [Klicpera et al., 2019b]. Our approach defines a novel multi-hop context-dependent self-attention GNN which resolves the over-smoothing issue of GAT architectures [Wang et al., 2019a]. [Isufi et al., 2020; Cucurull et al., 2018; Feng et al., 2019] also extends attention mechanism for multi-hop information aggregation, but they require different set of parameters to compute the attention to neighbors of different hops, making these approaches much more expensive compared to MAGNA, and were not extended to the knowledge graph settings.

6 Conclusion

We proposed Multi-hop Attention Graph Neural Network (MAGNA), which brings together benefits of graph attention and diffusion techniques in a single layer through attention diffusion, layer normalization and deep aggregation. MAGNA enables context-dependent attention between any pair of nodes in the graph in a single layer, enhances large-scale structural information, and learns more informative attention distribution. MAGNA improves over all state-of-the-art methods on the standard tasks of node classification and knowledge graph completion.

References


