Link prediction in Knowledge Graph with Multi-Attention Graph Neural Networks

Presenter: Nhut-Nam Le[1]

[1]Computer Science Department, Faculty of Information Technology, Univeristy of Science, VNU, HCM City, Vietnam

October 25, 2021

Table of contents

Multi-relational Graphs & Self-Attention Mechanism

i) A knowledge graph is a directed labeled graph in which the labels have well-defined meanings. One important characteristic of KG is **incompleteness**
ii) Self-Attention in GNNs let to **SOTA performance** on many Graph Representation Learning tasks

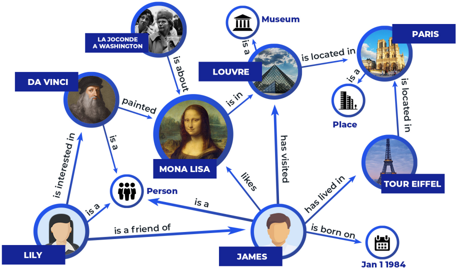

Figure: Visualize Knowledge Graph

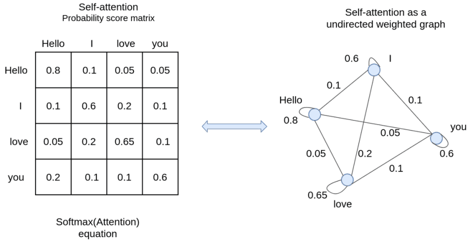

Figure: Self-attention

Attention mechanism limitation

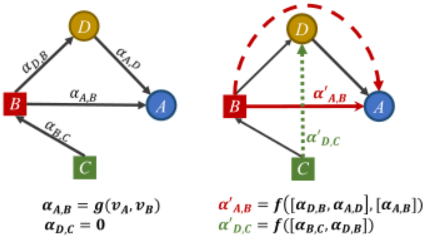

Figure: Multi-hop attention diffusion

- Previous model using attention mechanism only consider nodes that directly connected by an edge.
- The nodes in multi-hop neighbors of a node can provide important network context information

Preliminaries and problem statement

i) Knowledge graph (KG) is a heterogeneous graph. KG is defined by a set of entities (nodes) $v_i \in \mathcal{V}$, a set of relations (edges) $e = (v_i, r_k, v_j)$
ii) KG completion refers to the task of predicting an entity that has a specific relation with another given entity [Bordes et al., 2013]

- Input: Given $(?, r, t)$ or $(h, r, ?)$ or $(h, ?, t)$
- Output: Give a list ranked contain entity/relation which can replace "?"

iii) A general Graph Neural Network (GNN) approach learns an embedding that maps nodes and/or edge types into a continuous vector space.

Multi-hop Attention Diffusion

- Input: A set of triples (v_i, r_k, v_j), where v_i, v_j are nodes and r_k is the edge type

i) Edge Attention Computation: Compute the attention scores on all edges

Attention score s for an edge (v_i, r_k, v_j)

$$s_{i,k,j}^{(l)} = \delta(\mathbf{v}_a^{(l)} \tanh(\mathbf{W}_h^{(l)} \mathbf{h}_i^{(l)} || \mathbf{W}_t^{(l)} \mathbf{h}_j^{(l)} || \mathbf{W}_r^{(l)} \mathbf{r}_k^{(l)})) \tag{1}$$

For each edge of the graph $\mathcal{G}$, applying Eq.1, obtain an attention score matrix $\mathbf{S}^{(l)}$

$$\mathbf{S}_{i,j}^{(l)} = \begin{cases} s_{i,j,k}^{(l)}, & \text{if } (v_i, r_k, v_j) \text{ appears in } \mathcal{G} \\ -\infty, & \text{otherwise} \end{cases} \tag{2}$$

Attention matrix

$$\mathbf{A}^{(l)} = \text{softmax}(\mathbf{S}^{(l)}) \tag{3}$$

Multi-hop Attention Diffusion

ii) Attention Diffusion for Multi-hop Neighbors: Enable attention between nodes that are not directly connected in the graph by using Attention diffusion procedure
Procedure processing based the powers of the 1-hop attention matrix **A**

$$\mathcal{A} = \sum_{i=0}^{\infty} \theta_i \mathbf{A}^i \quad (4)$$

Where $\sum_{i=0}^{\infty} \theta_i = 1$ and $\theta_i > 0$
Implementation: Using geometric distribution $\theta_i = \alpha(1-\alpha)^i$, where $\alpha \in (0, 1]$
If $\theta_0 = \alpha \in (0, 1]$, $\mathbf{A}^0 = \mathbf{I}$ => Personalized Page Rank (PPR)
Graph attention diffusion based feature aggregation:

$$\text{AttDiff}(\mathcal{G}, \mathbf{H}^{(l)}, \Theta) = \mathcal{A}\mathbf{H}^{(l)} \quad (5)$$

Approximate $\mathcal{A}\mathbf{H}^{(l)}$ by defining a sequence which converges to the true value of $\mathcal{A}\mathbf{H}^{(l)}$ is $Z^{(K)}$ when $K \to \infty$: $Z^{(0)} = \mathbf{H}^{(l)}$, $Z^{(k+1)} = (1-\alpha)\mathcal{A}Z^{(k)} + \alpha Z^{(0)}$

Multi-head Graph Attention Diffusion Layer

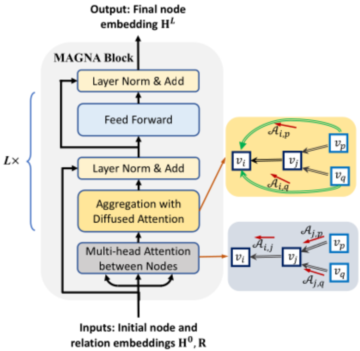

Figure: MAGNA Architecture

Multi-head Graph Attention Diffusion Layer

$$\hat{\mathbf{H}}^{(l)} = \text{MultiHead}(\mathcal{G}, \widetilde{\mathbf{H}}^{(l)}) = \left(||_{i=1}^{M} \text{head}_i \right) \mathbf{W}_0$$

$$\text{head}_i = \text{AttDiff}(\mathcal{G}, \widetilde{\mathbf{H}}^{(l)}, \Theta), \widetilde{\mathbf{H}}^{(l)} = LN(\mathbf{H}^{(l)})$$

Deep Aggregation

$$\hat{\mathbf{H}}^{(l+1)} = \hat{\mathbf{H}}^{(l)} + \mathbf{H}^{(l)}$$

$$\mathbf{H}^{(l+1)} = \mathbf{W}_2^{(l)} \text{ReLU}(\mathbf{W}_1^{(l)} LN(\mathbf{H}^{(l+1)})) + \hat{\mathbf{H}}^{(l+1)}$$

MAGNA generalizes GAT:

- Removing the restriction of attending to direct neighbors
- Using layer normalization and deep aggregation to achieve higher expressive

Knowledge Graph Completion

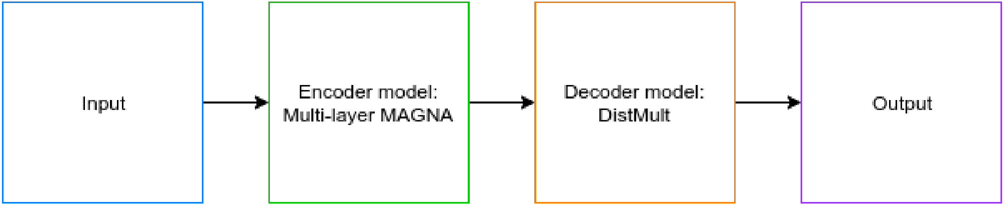

Figure: Encoder-Decoder framework for LP Problem

- The encoder applies the proposed MAGNA model to compute the entity embeddings
- The decoder makes link prediction given the embeddings

Conclusion & Improvement ideas

Conclusion Multi-hop Attention Graph Neural Network, MAGNA, has two main advantages:

- Captures long-range interactions between nodes that are not directly connected but may be multiple hops away.
- The attention computation is context-dependent.

Improvement ideas

- Can we compose combine local features around a node with multi-hop attention using Graph Diffusion, then obtained entity embeddings and relation embeddings???
- Graph Diffusion is still complexity too much, can we improve this issue?

References

Guangtao Wang, Rex Ying, Jing Huang, and Jure Leskovec.
Multi-hop attention graph neural network, 2021.