Supplementary Material

0.1 Embedding Process

The relation embedding process is shown in Algorithm 1, and the entity embedding and link prediction process is shown in Algorithm 2. The foundation model captures interactions by creating a graph of relations $G_r = (R, R_{fund}, E_r)$ to map out how different types of relations interact. In G_r, all $r \in R$ are unique relation types in the original graph G, and two nodes are linked if the corresponding types of relations in G share a common entity. $R_{fund} = \{$tail-to-tail edges, head-to-head edges, head-to-tail edges, tail-to-head edges$\}$ which are the four types of links in G_r that describe the basic topological structure of a directed graph.

Specifically, in our graph of relations, G_r, each node represents a unique relation type from the original graph, G. Two nodes in G_r are connected if their corresponding relation types in G share a common node. Depending on the directionality of these connections in G, the connections in G_r are classified as either tail-to-tail, head-to-head, head-to-tail, or tail-to-head.

Embeddings are obtained using conditional representations [1], which are demonstrably more expressive and practical than unconditional GNN encoders [2, 3]. By doing so, node representations are conditioned on query relations.

In Algorithm 1 and 2, **AGG** is a summation aggregation function and **MSG** is a multiplication messaging function.

Algorithm 1 Relation Graph Embedding

Input: Original graph G, Query $(h, q, ?)$, Dimension d
Output: d-dimensional node representations N_q of G_r

function ComputeRepr(q, G_r, d)
    for each node u in G_r **do**
        $N_q[u] \leftarrow (u == q)?\mathbf{1}^d : \mathbf{0}^d$
    return N_q

function MessagePassing(G_r, N_q)
    $t \leftarrow 0$
    repeat
        for each node v in G_r **do**
            $h_v[t+1] \leftarrow \text{UPDATE}(h_v[t], \text{AGG}(\{\text{MSG}(h_w[t], r) \mid w \in neighbors(v), r \in R_{\text{fund}}\}))$
        $t \leftarrow t + 1$
    until convergence
    return $h_v[t]$

$G_r \leftarrow \text{ConstructGr}(G)$
$N_q \leftarrow \text{ComputeRepr}(q, G_r, d)$
$N_q \leftarrow \text{MessagePassing}(G_r, N_q)$

0.2 Preserving Equivariance

0.2.1 Invariance and Equivariance

In inductive inference, neural networks are trained to discern and generalize invariant features within a graph to accommodate novel data. [4, 5] has revealed a notable parallel between the Weisfeiler-Lehman (WL) test and the operations of graph message passing neural networks, which facilitate convolution-like functionalities on graphs. A standard GNN updates the hidden state

Algorithm 2 Entity Embedding and Link Prediction

Input: Graph G, Query $(h, q, ?)$, Conditional relation representations R_q, Dimension d
Output: Logits $p(h, q, v)$ for each node v in G

function InitGraph(G, h, q, R_q, d)
    $h_0 \leftarrow \text{Vector}(|G|, d, 0)$ ▷ Zero vector for each node
    $h_0[h] \leftarrow R_q[q]$ ▷ Set query vector for head node
    return h_0

function MessagePassing(G, h_0, R_q, d)
    $t \leftarrow 0$
    $R_t \leftarrow \text{TransRelations}(R_q)$ ▷ Apply 2-layer MLP
    repeat
        for each node v in G **do**
            $h_v[t+1] \leftarrow \text{UPDATE}(h_v[t], \text{AGG}(\{\text{MSG}(h_w[t], R_t[r]) \mid w \in neighbors(v)\}))$
        $t \leftarrow t + 1$
    until convergence
    return $h_v[t]$

$h_0 \leftarrow \text{InitGraph}(G, h, q, R_q, d)$
$h_v \leftarrow \text{MessagePassing}(G, h_0, R_q, d)$
$logits \leftarrow \text{ComputeLogits}(h_v)$

$h_v^{(k)}$ of a node v at layer k as: $h_v^{(k)} = \phi\left(h_v^{(k-1)}, f\left(\left\{h_u^{(k-1)} : u \in N(v)\right\}\right)\right)$ [6], where ϕ denotes a transformation operation, f denotes an aggregation operation, $N(v)$ denote the neighboring nodes from previous layer $k-1$. Whereas, a 1-WL test iteration represents the label $l_v^{(k)}$ of node v as: $l_v^{(k)} = g\left(l_v^{(k-1)}, \text{sort}\left\{l_u^{(k-1)} : u \in N(v)\right\}\right)$ [4], where g is an injective function. [4] show by induction that (Theorem 3 in their paper) there always exists an injective function φ that $h_v^{(k)} = \varphi\left(l_v^{(k)}\right)$, which show that WL's 1-dimensional form 1-WL is analogous to GNN's neighbor aggregation. Showing the existence of the injective function φ is stating that for any given unique label $l_v^{(k)}$, there is a unique embedding $h_v^{(k)}$ in the GNN. It implies that these embeddings are independent of node IDs, similar to the WL test's ability to create unique labels for nodes based on their structure, rather than arbitrary properties like node IDs. Hence, these studies have underscored the benefits for inductive graph tasks from assuming node IDs are arbitrary, suggesting that models should maintain consistency despite permutations of node IDs. This concept, known as *permutation equivariance*, implies that the network does not rely on node IDs for embedding but rather employs the graph's structural topology for representation [7].

Extending this to multi-relational graphs such as KGs, the notion of learnable invariances extends to the joint permutations of node IDs and relation IDs, a concept referred to as *double permutation equivariance* or *double equivariance* defined by [3]. If you consider a graph that is processed by an RGCN (Relational Graph Convolutional Network) [8], the goal is for the RGCN to learn representations of nodes and relations from their structure rather than their labels or order. Given we have a permutation operation π, which shuffles the order of nodes and the types of relations between them. Permutation invariance in multi-relational graphs like RGCNs can be captured by demonstrating that the aggregation function f we defined previously is permutation-invariant concerning the representations of neighboring nodes of v through relation type r as $(N^r(v))$, then permutation invariance is established which can be defined as the aggregation function f_r:

$$f_r(\{h_u^{(k-1)}, w_r : u \in N^r(v)\}) =$$
$$f_r(\{h_{\pi(u)}^{(k-1)}, w_r : \pi(u) \in \pi(N^r(v))\}) \quad (1)$$

Theorem 1. *Graphs with counterfactual edge augmentation preserve double equivariance.*

$$f_r(\{h_u^{(k-1)}, w_r : u \in N_{cf}^r(v)\}) =$$
$$f_r(\{h_{\pi(u)}^{(k-1)}, w_r : \pi(u) \in \pi(N_{cf}^r(v))\}) \quad (2)$$

Proof. Let $G = (V, L, F)$ be the original KG, and $G' = (V, L \cup L_{cf}, F \cup F_{cf})$ represent the graph augmented with counterfactual edges. N_{cf}^r is the neighborhood of node v in G'. The aggregation function for relation r at layer k is denoted as f_r, which aggregates features from the neighborhood $N_{cf}^r(v)$ of a node v through relation r. Initially defined in Equation 1, the relation-based aggregation function f_r is permutation-invariant.

Since the addition of counterfactual edges L_{cf}, F_{cf} to the graph G to form G' simply extends the neighborhood $N_{cf}^r(v)$ without altering the inherent permutation-invariant property of f_r, the aggregation function will process $N_{cf}^r(v)$ and $\pi(N_{cf}^r(v))$ identically. Therefore, f_r preserves double equivariance in the presence of counterfactual edges, as the permutation of node IDs does not affect the outcome of the aggregation function. □

References

[1] Zhu Z, Zhang Z, Xhonneux LP, Tang J. Neural bellman-ford networks: A general graph neural network framework for link prediction. Advances in Neural Information Processing Systems. 2021;34:29476-90.

[2] Huang X, Romero M, Ceylan I, Barceló P. A Theory of Link Prediction via Relational Weisfeiler-Leman on Knowledge Graphs. Advances in Neural Information Processing Systems. 2024;36.

[3] Gao J, Zhou Y, Zhou J, Ribeiro B. Double equivariance for inductive link prediction for both new nodes and new relation types. arXiv preprint arXiv:230201313. 2023.

[4] Xu K, Hu W, Leskovec J, Jegelka S. How powerful are graph neural networks? arXiv preprint arXiv:181000826. 2018.

[5] Morris C, Ritzert M, Fey M, Hamilton WL, Lenssen JE, Rattan G, et al. Weisfeiler and leman go neural: Higher-order graph neural networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33; 2019. p. 4602-9.

[6] Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:160902907. 2016.

[7] Srinivasan B, Ribeiro B. On the equivalence between positional node embeddings and structural graph representations. arXiv preprint arXiv:191000452. 2019.

[8] Schlichtkrull M, Kipf TN, Bloem P, Van Den Berg R, Titov I, Welling M. Modeling relational data with graph convolutional networks. In: The semantic web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, proceedings 15. Springer; 2018. p. 593-607.