Homework-2 solutions

CS224W (Fall 2021)

Karan Bania*
Machine Learning Department
Carnegie Mellon University
kbania@andrew.cmu.edu

1 GCN

Question 1.1

Yes, these graphs **are** isomorphic, mapping -

Node number, x	$\phi(x)$
1	A
2	D
3	E
4	H
5	C
6	B
7	G
8	F

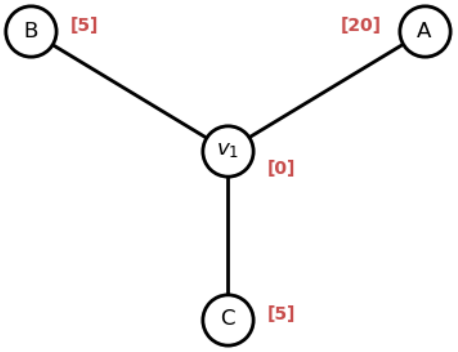
Table 1: Mapping ϕ

However, there are several such mappings possible.

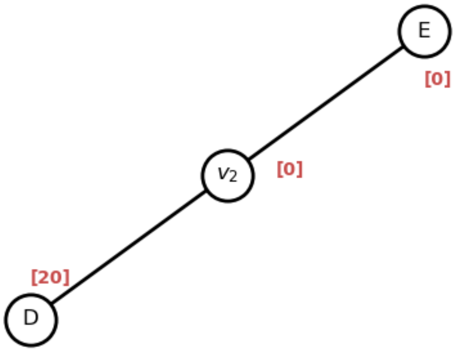
*I solved them while I was an undergraduate at BITS Pilani, Goa Campus.

Question 1.2

The two graphs are -



(a) Graph 1.



(b) Graph 2.

Now, clearly, if we perform mean aggregation, $h^{(1)}_{v1} = h^{(2)}_{v1} = 10$ and for max aggregation, $h^{(1)}_{v1} = h^{(2)}_{v1} = 20$, but using sum aggregation yields $h^{(1)}_{v1} = 30, h^{(2)}_{v1} = 20$, which are different (I have set $h^{(0)}_{v1} = h^{(0)}_{v2} = [0]$, so that the solution works out regardless of the choice of taking self-loops, but otherwise the information that these two nodes have the same initial embedding is of no use for the `aggregate` step).

Question 1.3

Proving by contradiction, we are given,

$$\texttt{readout}(h^{(K)}_v, \forall v \in V1) \neq \texttt{readout}(h^{(K)}_v, \forall v \in V2) \tag{1}$$

and the WL test cannot decide the graphs are not isomorphic after the Kth iteration, which means [1] [2],

$$\forall v_1 \in V_1 \, \exists v_2 \in V_2 \,|\, l^{(K)}_{v_1} = l^{(K)}_{v_2} \tag{2}$$

We also set the `combine`, `aggregate` and `readout` functions to `HASH`, $\{\{\cdot\}\}$ (multiset) and set-equality respectively to align with the WL algorithm. Given (1), we

[1] Importantly, there is another version of the WL test, which *does not* input the number of iterations K, i.e., it continues till the hashes have converged, however, this is not the case for Algorithm 3 in the assignment. In fact, proving up to K iterations ($\forall K$) is enough to conclude that the two algorithms are equal, as then we just set $K = \textit{convergence}$ and be done.

[2] Also, 2 should have a reverse condition, i.e., $\forall v_2 \in V_2 \exists v_1 \in V_1$, but because isomorphism only matters for graphs where $|V_1| = |V_2|$, one side is enough to imply the other.

can conclude that,

$$\exists v_1 \in V_1 \nexists v_2 \in V_2 \,|\, h_{v_1}^{(K)} = h_{v_2}^{(K)} \tag{3}$$

$$\implies h_{v_1}^{(K)} \neq h_{v_2}^{(K)} \tag{4}$$

Because we use the same functions,

$$\exists v_1 \in V_1 \nexists v_2 \in V_2 \,|\, l_{v1}^{(K)} \neq l_{v2}^{(K)} \tag{5}$$

Clearly, 2 & 5 lead to a contradiction, thus our initial assumption that the *WL test cannot decide* two graphs are not isomorphic is wrong.

2 Node Embeddings with TransE

Question 2.1

We want to minimize,

$$\mathcal{L}_{simple} = \sum_{(h,l,t) \in S} d(\mathbf{h} + l, \mathbf{t}) \tag{6}$$

Consider the graph -

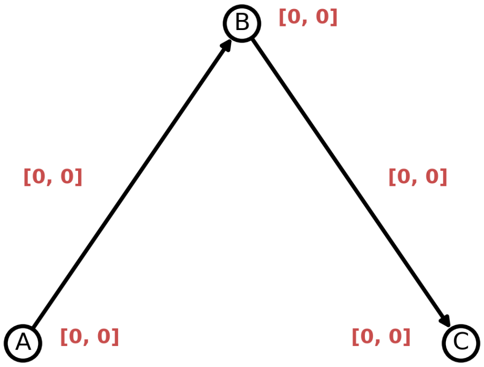


Figure 1: Example graph which tricks L_{simple}.

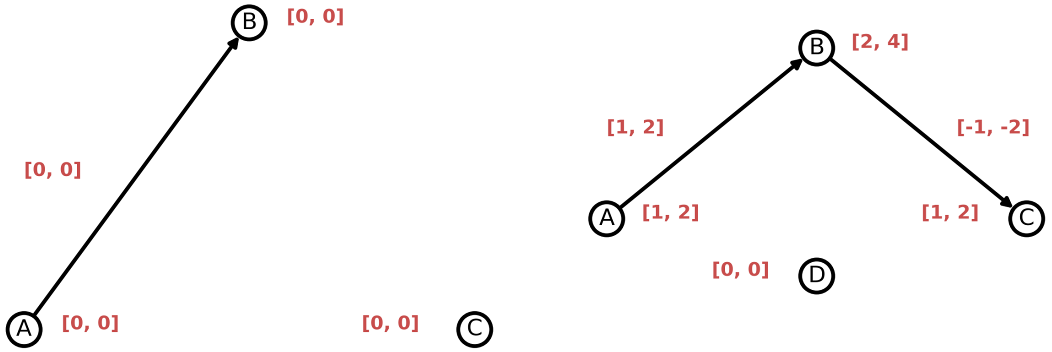
These embeddings drive L_{simple} to 0, but are completely useless. The graph has 3 distinct types of nodes, one only has one outgoing edge, one only has one incoming edge, and one has both an incoming and an outgoing edge; which are all mapped to the same embedding.

Question 2.2

Now, the loss function is,

$$\mathcal{L}_{\text{no margin}} = \sum_{(h,l,t)\in S} \sum_{(h',l,t')\in S'_{(h,l,t)}} [d(\mathbf{h}+l,\mathbf{t}) - d(\mathbf{h}'+l,\mathbf{t}')]_+ \tag{7}$$

The graph in 1 can be modified for this case as,



(a) Example graph which tricks $L_{\text{no margin}}$

(b) Slightly non-trivial example graph which tricks $L_{\text{no margin}}$

Figure 2: Examples of graphs which trick the $L_{\text{no margin}}$ loss function.

Given all embeddings,

$$\mathbf{e}_A = \begin{bmatrix}0\\0\end{bmatrix}, \quad \mathbf{e}_B = \begin{bmatrix}0\\0\end{bmatrix}, \quad \mathbf{e}_C = \begin{bmatrix}0\\0\end{bmatrix}, \quad \mathbf{e}_{(A,B)} = \begin{bmatrix}0\\0\end{bmatrix}$$

We can't tell where the edge is. A slightly complex example is in 2b.

2.1 Question 2.3

The algorithm could then just scale $\mathbf{h}$ & $\mathbf{t}$ such that $\mathbf{h} + l \approx \mathbf{t}$, in this case all embeddings would be quite close (& large), separated by small ls[3].

[3] While not directly related to the answer, scaling would also help with SGD, as otherwise the gradients could be quite large.

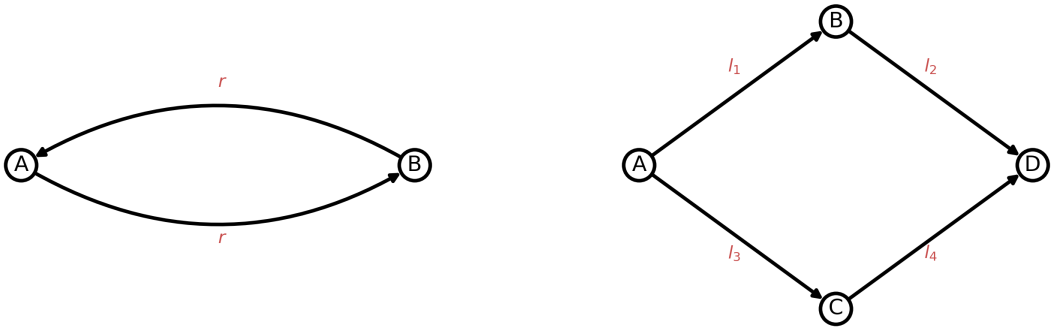
2.2 Question 2.4

An impossible simple graph is described in 3a. Suppose a perfect embedding exists, described by $e_{\{A-B\}}$ and r. These embeddings describe the linear system,

$$e_A + r = e_B \tag{8}$$

$$e_B + r = e_A \tag{9}$$

This clearly leads to $e_A = e_B$ and $r = 0$, which also means $e_A + r = e_A$ which doesn't exist in out graph. So TransE is cannot model *symmetric* relationships.



(a)

(b)

Figure 3: Impossible graphs with respect to TransE.

Another impossible graph is in 3b. A similar analysis to before gives us the following relations,

$$e_A = e_D \tag{10}$$

$$l_1 = -l_3 \tag{11}$$

$$l_2 = -l_4 \tag{12}$$

This means $e_B + l_2 = e_A$, which clearly doesn't exist in out graph.

3 Expressive Power of Knowledge Graph Embeddings

Question 3.1

Except for Symmetry, TransE can model the other two (explained in lectures).

Question 3.2

I would recommend reading [1] before answering this question.

Symmetry, modeling this requires

$$\mathbf{h} \circ \mathbf{r} = \mathbf{t},$$
$$\mathbf{t} \circ \mathbf{r} = \mathbf{h}$$

without requiring $\mathbf{r} = 0$.

The equations to $(\mathbf{h} \circ \mathbf{r}) \circ \mathbf{r} = \mathbf{h} \implies \mathbf{h} \circ (\mathbf{r} \circ \mathbf{r}) = \mathbf{h}$, also

$$\forall \mathbf{h} \in \mathbb{C}^k \; \exists \mathbf{h}^{-1} \in \mathbb{C}^k \,|\, \mathbf{h} \circ \mathbf{h}^{-1} = \mathbf{1}$$

Thus, we have $\mathbf{r} \circ \mathbf{r} = \mathbf{1}$, i.e. $\mathbf{r} \neq \mathbf{0}$. RotatE *can* model Symmetric relations.

Inverse, modeling this requires

$$\mathbf{h} \circ \mathbf{r}_1 = \mathbf{t}$$
$$\mathbf{t} \circ \mathbf{r}_2 = \mathbf{h}$$

without requiring $\mathbf{r} = 0$.

The model can set[4] $\mathbf{r}_1 = \mathbf{r}_2^{-1}$.

Composition, a typical relation would be

$$\mathbf{h}_1 \circ \mathbf{r}_1 = \mathbf{h}_2$$
$$\mathbf{h}_2 \circ \mathbf{r}_2 = \mathbf{h}_3$$
$$\mathbf{h}_1 \circ \mathbf{r}_3 = \mathbf{h}_3$$

This can be satisfied with[5] $\mathbf{r}_3 = \mathbf{r}_1 \circ \mathbf{r}_2$.

Question 3.3

An impossible graph[6] is 4.

All arrows have the same relation $\mathbf{r}$. Clearly to satisfy the 3-cycle (A-B-C), we require $\mathbf{r}$ to be a rotation of 120°, but to satisfy the 2-cycle (A-D), we require $\mathbf{r}$ to be a rotation of 180°, which is impossible.

TransE cannot model this graph as it has a symmetric relationship (A-D).

[4] for any $\mathbf{r} \in \mathbb{C}^k$, to rotate a vector in the opposite direction, we would multiply by $\mathbf{r}^{-1}$ and not $-\mathbf{r}$, this also follows because each $|r_i| = 1 \implies r_i = e^{i\theta_i}$.

[5] $(\mathbf{h}_1 \circ \mathbf{r}_1) \circ \mathbf{r}_2 = \mathbf{h}_3 \implies \mathbf{h}_1 \circ (\mathbf{r}_1 \circ \mathbf{r}_2) = \mathbf{h}_3$

[6] Perplexity helped me quite a lot with this one!

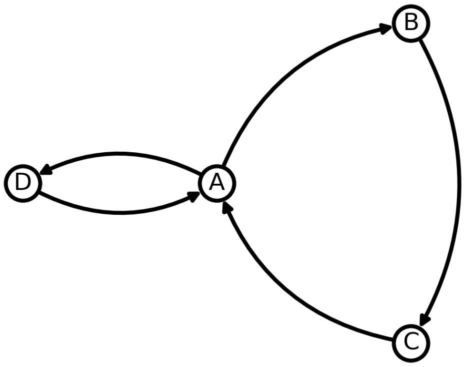


Figure 4: An impossible example for RotatE.

References

[1] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*, 2019.