11-741/11-441 – Machine Learning with Graphs **Due: April 25, 11:59PM**

Instructor: Yiming Yang

Homework 5: Knowledge Graph Embedding

In this assignment, you answer questions about several Knowledge Graph Embedding (KGE) algorithms.[1]

1 Honor Code [0 points]

(X) I have read and understood Honor Code before I submitted my work.

Collaboration: Write down the names & Andrew ID of students you collaborated with on Homework 2 (None if you didn't).

Note: Read our website on our policy about collaboration!

2 Node Embeddings with TransE [20 points]

While many real world systems are effectively modeled as graphs, graphs can be a cumbersome format for certain downstream applications, such as machine learning models. It is often useful to represent each node of a graph as a vector in a continuous low dimensional space. The goal is to preserve information about the structure of the graph in the vectors assigned to each node. For instance, the spectral embedding preserved structure in the sense that nodes connected by an edge were usually close together in the (one-dimensional) embedding x.

Multi-relational graphs are graphs with multiple types of edges. They are incredibly useful for representing structured information, as in knowledge graphs. There may be one node representing "Washington, DC" and another representing "United States", and an edge between them with the type "Is capital of". In order to create an embedding for this type of graph, we need to capture information about not just which edges exist, but what the types of those edges are. In this problem, we will explore a particular algorithm designed to learn node embeddings for multi-relational graphs.

The algorithm we will look at is TransE.[2] We will first introduce some notation used in the paper describing this algorithm. We'll let a multi-relational graph $G = (E, S, L)$ consist of the set of *entities* E (i.e., nodes), a set of edges S, and a set of possible relationships L. The set S consists of triples (h, l, t), where $h \in E$ is the *head* or source-node, $l \in L$ is the relationship, and $t \in E$ is the *tail* or destination-node. As a node embedding, TransE tries to learn embeddings of each entity $e \in E$ into $\mathbb{R}^k$ (k-dimensional vectors), which we will notate by $\mathbf{e}$. The main innovation of TransE is that each relationship ℓ is also embedded as a vector $\ell \in \mathbb{R}^k$, such that the difference between the embeddings of entities linked via the relationship ℓ is approximately ℓ. That is, if $(h, \ell, t) \in S$, TransE tries to ensure that $\mathbf{h} + \boldsymbol{\ell} \approx \mathbf{t}$. Simultanesouly, TransE tries to make sure that $\mathbf{h} + \boldsymbol{\ell} \not\approx \mathbf{t}$ if the edge (h, ℓ, t) does not exist.

Note on notation: we will use unbolded letters e, ℓ, etc. to denote the entities and relationships in the graph, and bold letters $\mathbf{e}, \boldsymbol{\ell}$, etc., to denote their corresponding embeddings. TransE accomplishes this

[1]This homework is adopted from **Stanford CS224W**.

[2]See the 2013 NIPS paper by Bordes et al.

by minimizing the following loss:

$$\mathcal{L} = \sum_{(h,\ell,t)\in S} \left(\sum_{(h',\ell,t')\in S'_{(h,\ell,t)}} \left[\gamma + d(\mathbf{h}+\boldsymbol{\ell},\mathbf{t}) - d\left(\mathbf{h}'+\boldsymbol{\ell},\mathbf{t}'\right)\right]_+ \right) \tag{1}$$

Here (h', ℓ, t') are "corrupted" triplets, chosen from the set $S'_{(h,\ell,t)}$ of corruptions of (h, ℓ, t), which are all triples where either h or t (but not both) is replaced by a random entity, and ℓ remains the same as the one in the original triplets.

$$S'_{(h,\ell,t)} = \{(h',\ell,t) \mid h' \in E\} \cup \{(h,\ell,t') \mid t' \in E\}$$

Additionally, $\gamma > 0$ is a fixed scalar called the *margin*, the function $d(\cdot,\cdot)$ is the Euclidean distance, and $[\cdot]_+$ is the positive part function (defined as $\max(0,\cdot)$). Finally, TransE restricts **all the entity embeddings to have length** $1 : \|\mathbf{e}\|_2 = 1$ **for every** $e \in E$.
For reference, here is the TransE algorithm, as described in the original paper on page 3:

Algorithm 1 Learning TransE

input Training set $S = \{(h,\ell,t)\}$, entities and rel. sets E and L, margin γ, embeddings dim. k.

initialize $\boldsymbol{\ell} \leftarrow$ uniform$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each $\ell \in L$
  $\boldsymbol{\ell} \leftarrow \boldsymbol{\ell} / \|\boldsymbol{\ell}\|$ for each $\ell \in L$
  $\mathbf{e} \leftarrow$ uniform$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each entity $e \in E$
loop
  $\mathbf{e} \leftarrow \mathbf{e} / \|\mathbf{e}\|$ for each entity $e \in E$
  $S_{batch} \leftarrow$sample(S, b) // sample a minibatch of size b
  $T_{batch} \leftarrow \emptyset$ // initialize the set of pairs of triplets
  for $(h,\ell,t) \in S_{batch}$ **do**
    $(h',\ell,t') \leftarrow$sample$(S'_{(h,\ell,t)})$ // sample a corrupted triplet
    $T_{batch} \leftarrow T_{batch} \cup \{((h,\ell,t),(h',\ell,t'))\}$
  end for
  Update embeddings w.r.t. $\displaystyle\sum_{((h,\ell,t),(h',\ell,t')) \in T_{batch}} \nabla\left[\gamma + d(\boldsymbol{h}+\boldsymbol{\ell},\boldsymbol{t}) - d(\boldsymbol{h'}+\boldsymbol{\ell},\boldsymbol{t'})\right]_+$
end loop

2.1 Simplified Objective [3 points]

Say we were intent on using a simpler loss function. Our objective function (1) includes a term maximizing the distance between $\mathbf{h}' + \boldsymbol{\ell}$ and $\mathbf{t}'$. If we instead simplified the objective, and just tried to minimize

$$\mathcal{L}_{\text{simple}} = \sum_{(h,\ell,t)\in S} d(\mathbf{h}+\boldsymbol{\ell},\mathbf{t}), \tag{2}$$

we would obtain a useless embedding. Give an example of a simple graph and corresponding embeddings which will minimize the new objective function (2) all the way to zero, but still give a completely useless embedding.
Hint: Your graph should be non-trivial, i.e., it should include at least two nodes and at least one edge. Assume the embeddings are in 2 dimensions, i.e., $k = 2$. What happens if $\boldsymbol{\ell} = \mathbf{0}$?

★ **Solution** ★

2.2 Utility of γ [5 points]

We are interested in understanding what the margin term γ accomplishes. If we removed the margin term γ from our loss, and instead optimized

$$\mathcal{L}_{\text{no margin}} = \sum_{(h,\ell,t)\in S} \sum_{(h',\ell t')\in S'_{(h,\ell,t)}} \left[d(\mathbf{h}+\boldsymbol{\ell},\mathbf{t}) - d\left(\mathbf{h}'+\boldsymbol{\ell},\mathbf{t}'\right)\right]_+ , \tag{3}$$

it turns out that we would again obtain a useless embedding. Give an example of a simple graph and corresponding embeddings which will minimize the new objective function (3) all the way to zero, but still give a completely useless embedding. By useless, we mean that in your example, you cannot tell just from the embeddings whether two nodes are linked by a particular relation (Note: your graph should be non-trivial, i.e., it should include at least two nodes and at least one edge. Assume the embeddings are in 2 dimensions, i.e., $k = 2$.)

★ **Solution** ★

2.3 Normalizing the embeddings [5 points]

Recall that TransE normalizes every entity embedding to have unit length (see line 5 of the algorithm). The quality of our embeddings would be much worse if we did not have this step. To understand why, imagine running the algorithm with line 5 omitted. What could the algorithm do to trivially minimize the loss in this case? What would the embeddings it generates look like?

★ **Solution** ★

2.4 Expressiveness of TransE embeddings [7 points]

Give an example of a simple graph for which no perfect embedding exists, i.e., no embedding perfectly satisfies $\mathbf{u}+\boldsymbol{\ell}=\mathbf{v}$ for all $(u,\ell,v)\in S$ and $\mathbf{u}+\boldsymbol{\ell}\neq\mathbf{v}$ for $(u,\ell,v)\notin S$, for any choice of entity embeddings ($\mathbf{e}$ for $e\in E$) and relationship embeddings ($\boldsymbol{\ell}$ for $\ell\in L$). Explain why this graph has no perfect embedding in this system, and what that means about the expressiveness of TransE embeddings. As before, assume the embeddings are in 2 dimensions ($k = 2$).
Hint: By expressiveness of TransE embeddings, we want you to talk about which type of relationships TransE can/cannot model with an example. (Note that the condition for this question is slightly different from that for Question 2.1 and what we ask you to answer is different as well).

★ **Solution** ★

3 Expressive Power of Knowledge Graph Embeddings [10 points]

TransE is a common method for learning representations of entities and relations in a knowledge graph. Given a triplet (h, ℓ, t), where entities embedded as h and t are related by a relation embedded as ℓ, TransE trains entity and relation embeddings to make $h + \ell$ close to t. There are some common patterns that relations form:

- Symmetry: A is married to B, and B is married to A.
- Inverse: A is teacher of B, and B is student of A. Note that teacher and student are 2 different relations and have their own embeddings.
- Composition: A is son of B; C is sister of B, then C is aunt of A. Again note that son, sister, and aunt are 3 different relations and have their own embeddings.

3.1 TransE Modeling [3 points]

For each of the above relational patterns, can TransE model it perfectly, such that $h + \ell = t$ for all relations? Explain why or why not. Note that here **0** embeddings for relation are undesirable since that means two entities related by that relation are identical and not distinguishable.

★ Solution ★

3.2 RotatE Modeling [3 points]

Consider a new model, RotatE. Instead of training embeddings such that $h + \ell \approx t$, we train embeddings such that $h \circ \ell \approx t$. Here $\circ$ means rotation. You can think of h as a vector of dimension $2d$, representing d 2D points. ℓ is a d-dimensional vector specifying rotation angles. When applying $\circ$, For all $i \in 0 \dots d-1$, (h_{2i}, h_{2i+1}) is rotated clockwise by l_i. Similar to TransE, the entity embeddings are also normalized to L2 norm 1. Can RotatE model the above 3 relation patterns perfectly? Why or why not?

★ Solution ★

3.3 Failure Cases [4 points]

Give an example of a graph that RotatE cannot model perfectly. Can TransE model this graph perfectly? Assume that relation embeddings cannot be **0** in either model.

★ Solution ★