Node Representation Learning

Jian Tang
HEC Montreal
CIFAR AI Chair, Mila
Email: jian.tang@hec.ca
Outline

- Node Representation Methods
  - LINE, DeepWalk, node2vec

- Graph and High-dimensional Data Visualization
  - LargeVis

- Knowledge Graph Embedding
  - RotatE (Sun et al., ICLR’19)

- A High-performance Node Representation System (Zhu et al., WWW’19)
Problem Definition: Node Embedding

- Given a network/graph $G=(V, E, W)$, where $V$ is the set of nodes, $E$ is the set of edges between the nodes, and $W$ is the set of weights of the edges, the goal of node embedding is to represent each node $i$ with a vector $\tilde{u}_i \in \mathbb{R}^d$, which preserves the structure of networks.
Related Work

• Classical graph embedding algorithms
  • MDS, IsoMap, LLE, Laplacian Eigenmap, ...
  • Hard to scale up

• Graph factorization (Ahmed et al. 2013)
  • Not specifically designed for network representation
  • Undirected graphs only

• Neural word embeddings (Bengio et al. 2003)
  • Neural language model
  • word2vec (skipgram), paragraph vectors, etc.
LINE: Large-scale Information Network Embedding (Tang et al., Most Cited Paper of WWW 2015)

• Arbitrary types of networks
  • Directed, undirected, and/or weighted

• Clear objective function
  • Preserve the first-order and second-order proximity

• Scalable
  • Asynchronous stochastic gradient descent
  • Millions of nodes and billions of edges: a couple of hours on a single machine
First-order Proximity

- The local pairwise proximity between the nodes
- However, many links between the nodes are not observed
  - Not sufficient for preserving the entire network structure
Second-order Proximity

“The degree of overlap of two people’s friendship networks correlates with the strength of ties between them” --Mark Granovetter

“You shall know a word by the company it keeps” --John Rupert Firth

• Proximity between the neighborhood structures of the nodes
Preserving the First-order Proximity (LINE 1st)

• Distributions: (defined on the undirected edge $i - j$)

Empirical distribution of first-order proximity:

$$\hat{p}_1(v_i, v_j) = \frac{w_{ij}}{\sum_{(m,n)\in E} w_{mn}}$$

Model distribution of first-order proximity:

$$p_1(v_i, v_j) = \frac{\exp(\bar{u}_i^T \bar{u}_j)}{\sum_{(m,n)\in V\times V} \exp(\bar{u}_m^T \bar{u}_n)}$$

• Objective:

$$O_1 = KL(\hat{p}_1, p_1) = - \sum_{(i,j)\in E} w_{ij} \log p_1(v_i, v_j)$$
Preserving the Second-order Proximity (LINE 2nd)

• Distributions: (defined on the directed edge \( i \rightarrow j \))

  Empirical distribution of neighborhood structure:
  \[
  \hat{p}_2(v_j | v_i) = \frac{w_{ij}}{\sum_{k \in V} w_{ik}}
  \]

  Model distribution of neighborhood structure:
  \[
  p_2(v_j | v_i) = \frac{\exp(\tilde{u}_i^T \tilde{u}_j)}{\sum_{k \in V} \exp(\tilde{u}_k^T \tilde{u}_i)}
  \]

• Objective:

  \[
  O_2 = \sum_i KL(\hat{p}_2(\cdot | v_i), p_2(\cdot | v_i)) = - \sum_{(i,j) \in E} w_{ij} \log p_2(v_j | v_i)
  \]
Optimization Tricks

• Stochastic gradient descent + Negative Sampling
  • Randomly sample an edge and multiple negative edges
• The gradient w.r.t the embedding with edge \((i, j)\)
  \[
  \frac{\partial O_2}{\partial \tilde{u}_i} = w_{ij} \frac{\partial \log \hat{p}_2(v_j | v_i)}{\partial \tilde{u}_i}
  \]
• Problematic when the variances of weights of the edges are large
  • The variance of the gradients are large
• Solution: edge sampling
  • Sample the edges according to their weights and treat the edges as binary
• Complexity: \(O(d*K*|E|)\)
  • Linear to the dimensionality \(d\), the number of negative samples \(K\), and the number of edges
Discussion

• Embed nodes with few neighbors
  • Expand the neighbors by adding higher-order neighbors
  • Breadth-first search (BFS)
    • Adding only second-order neighbors works well in most cases

• Embed new nodes
  • Fix the embeddings of existing nodes
  • Optimize the objective w.r.t. the embeddings of new nodes
DeepWalk (Perozzi et al. 2014)

- Learning node representations with the technique for learning word representations, i.e., Skipgram
- Treat *random walks on networks* as sentences

\[
p(v_j | v_i) = \frac{\exp(\mathbf{u}_i^T \mathbf{u}_j)}{\sum_{k \in V} \exp(\mathbf{u}_i^T \mathbf{u}_k)}
\]

Bryan Perozzi, Rami Al-Rfou, Steven Skiena. DeepWalk: Online Learning of Social Representations. KDD’14
Node2Vec (Grover and Leskovec, 2016)

Figure 1: BFS and DFS search strategies from node $u$ ($k = 3$).

- Find the node context with a hybrid strategy of
  - Breadth-first Sampling (BFS): *homophily*
  - Depth-first Sampling (DFS): *structural equivalence*

Aditya Grover and Jure Leskovec. node2vec: Scalable Feature Learning for Networks. KDD’16
Expand Node Contexts with Biased Random Walk

- Biased random walk with two parameters $p$ and $q$
  - $p$: controls the probability of revisiting a node in the walk
  - $q$: controls the probability of exploring “outward” nodes
  - Find optimal $p$ and $q$ through cross-validation on labeled data
- Optimized through similar objective as LINE with first-order proximity
Comparison between LINE, DeepWalk, and Node2Vec

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Neighbor Expansion</th>
<th>Proximity</th>
<th>Optimization</th>
<th>Validation Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINE</td>
<td>BFS</td>
<td>1&lt;sup&gt;st&lt;/sup&gt; or 2&lt;sup&gt;nd&lt;/sup&gt;</td>
<td>Negative Sampling</td>
<td>No</td>
</tr>
<tr>
<td>DeepWalk</td>
<td>Random</td>
<td>2&lt;sup&gt;nd&lt;/sup&gt;</td>
<td>Hierarchical Softmax</td>
<td>No</td>
</tr>
<tr>
<td>Node2Vec</td>
<td>BFS + DFS</td>
<td>1&lt;sup&gt;st&lt;/sup&gt;</td>
<td>Negative Sampling</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Applications

- Node classification (Perozzi et al. 2014, Tang et al. 2015a, Grover et al. 2015)
- Node visualization (Tang et al. 2015a)
- Link prediction (Grover et al. 2015)
- Recommendation (Zhao et al. 2016)
- Text representation (Tang et al. 2015a, Tang et al. 2015b)
- ...
Many Extensions …

• Leverage global structural information (Cao et al. 2015)
• Non-linear methods based on autoencoders (Wang et al. 2016)
• Matrix-factorization based approaches (Qiu et al. 2018)
• Directed network embedding (Ou et al. 2016)
• Signed network embedding (Wang et al. 2017)
• Multi-view networks (Qu and Tang et al. 2017)
• Networks with node attributes (Yang et al. 2015)
• Heterogeneous networks (Chang et al. 2015)
• Task-specific network embedding (Chen et al. 2017)
Outline

• Node Representation Methods
  • LINE, DeepWalk, node2vec

• Graph and High-dimensional Data Visualization
  • LargeVis

• Knowledge Graph Embedding
  • RotatE

• A High-performance Node Representation System
Extremely Low-dimensional Representations: 2D/3D for Visualizing Networks

K-Nearest Neighbor Graph (KNN-G) Construction → Graph Layout

High-dimensional Data → Networks → 2D/3D Layout

Scatter Plots, Network Diagrams, Heatmaps
t-SNE (Maarten and Hinton, 2008, 2014)

• State-of-the-art algorithm for high-dimensional data visualization
  • Deployed by Tensorflow

• Limitations
  • K-NNG construction: complexity grows $O(N \log N)$ to the number of data points $N$
  • Graph layout: complexity is $O(N \log N)$
  • Very sensitive parameters

TensorBoard Visualizations by t-SNE
LargeVis (Tang et al., Best Paper Nomination at WWW 2016)

• Efficient approximation of K-NNG construction
  • 30 times faster than t-SNE (3 million data points)
  • Better time-accuracy tradeoff

• Efficient probabilistic model for graph layout
  • $O(N \log N) \rightarrow O(N)$
  • 7 times faster than t-SNE (3 million data points)
  • Better visualization layouts
  • Stable parameters across different data sets

Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. Visualizing Large-scale and High-dimensional Data. WWW’16
Learning the Layout of KNN Graph

• Preserve the similarities of the nodes in 2D/3D space
  • Represent each node with a 2D/3D vector
  • Keep similar data close while dissimilar data far apart
• Probability of observing a binary edge between nodes (i,j):
  \[ p(e_{ij} = 1) = \frac{1}{1 + \| \tilde{y}_i - \tilde{y}_j \|^2} \]
• Likelihood of observing a weighted edge between nodes (i,j):
  \[ p(e_{ij} = w_{ij}) = p(e_{ij} = 1)^{w_{ij}} \]
A Probabilistic Model for Graph Layout

• Objective:

\[ O = \prod_{(i,j)\in E} p(e_{ij} = w_{ij}) \prod_{(i,j)\in E} (1 - p(e_{ij} = w_{ij}))^\gamma \]

\( \gamma \): an unified weight assigned to negative edge

• Randomly sample some negative edges
• Optimized through asynchronous stochastic gradient descent
• Time complexity: linear to the number of data points
10M Scientific Papers on One Slide
Computer Science vs. Mathematics
Computer Science vs. Physics
Wikipedia Articles
(color: semantic cluster)
Outline

• Node Representation Methods
  • LINE, DeepWalk, node2vec

• Graph and High-dimensional Data Visualization
  • LargeVis

• Knowledge Graph Embedding
  • RotatE

• A High-performance Node Representation System
Knowledge Graphs

• Knowledge graphs are **heterogeneous** graphs
  • Multiple types of relations

• A set of facts represented as triplets
  • (head entity, relation, tail entity)

• A variety of applications
  • Question answering
  • Search
  • Recommender Systems
  • Natural language understanding
  • …
Related Work on Knowledge Graph Embedding

- Representing entities as embeddings
- Representing relations as embeddings or matrices

<table>
<thead>
<tr>
<th>Model</th>
<th>Score Function</th>
<th>$h, r, t \in \mathbb{R}^k$, $W_{r,i} \in \mathbb{R}^{k \times k}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE (Bordes et al., 2011)</td>
<td>$- | W_{r,1} h - W_{r,2} t |$</td>
<td>$h, t \in \mathbb{R}^k$, $W_{r,i} \in \mathbb{R}^{k \times k}$</td>
</tr>
<tr>
<td>TransE (Bordes et al., 2013)</td>
<td>$- | h + r - t |$</td>
<td>$h, r, t \in \mathbb{R}^k$</td>
</tr>
<tr>
<td>TransX</td>
<td>$- | g_{r,1}(h) + r - g_{r,2}(t) |$</td>
<td>$h, r, t \in \mathbb{R}^k$</td>
</tr>
<tr>
<td>DistMult (Yang et al., 2014)</td>
<td>$\langle r, h, t \rangle$</td>
<td>$h, r, t \in \mathbb{R}^k$</td>
</tr>
<tr>
<td>ComplEx (Trouillon et al., 2016)</td>
<td>$\text{Re}(\langle r, h, \bar{t} \rangle)$</td>
<td>$h, r, t \in \mathbb{C}^k$</td>
</tr>
<tr>
<td>HolE (Nickel et al., 2016)</td>
<td>$\langle r, h \otimes t \rangle$</td>
<td>$h, r, t \in \mathbb{R}^k$</td>
</tr>
<tr>
<td>ConvE (Dettmers et al., 2017)</td>
<td>$\langle \sigma(\text{vec}(\sigma([r, h] * \Omega)) W), t \rangle$</td>
<td>$h, r, t \in \mathbb{R}^k$</td>
</tr>
<tr>
<td>RotatE</td>
<td>$- | h \circ r - t |$</td>
<td>$h, r, t \in \mathbb{C}^k$, $</td>
</tr>
</tbody>
</table>
Task: Knowledge Graph Completion

• A fundamental task: predicting missing links

• The Key Idea: model and infer the relation patterns in knowledge graphs according to observed knowledge facts.
  - The relationship between relations

• Example:

  Barack Obama BornIn United States

  Parents of Parents are Grandparents

  Barack Obama Nationality American
Relation Patterns

- **Symmetric/Antisymmetric** Relations
  - Symmetric: e.g., Marriage
  - Antisymmetric: e.g., Filiation

- Formally:
  
  \[
  r \text{ is Symmetric}: \quad r(x, y) \Rightarrow r(y, x) \text{ if } \forall x, y
  \]
  
  \[
  r \text{ is Antisymmetric}: \quad r(x, y) \Rightarrow \neg r(y, x) \text{ if } \forall x, y
  \]
Relation Patterns

- **Inverse** Relations
  - Hypernym and hyponym
  - Husband and wife

- Formally:

  \[ r_1 \text{ is inverse to relation } r_2: \quad r_2(x, y) \Rightarrow r_1(y, x) \text{ if } \forall x, y \]
Relation Patterns

- **Composition** Relations
  - My mother’s husband is my father
- Formally:

\[
\begin{align*}
\text{\( r_1 \) is a composition of relation \( r_2 \) and relation \( r_3 \):} & \quad r_2(x, y) \land r_3(y, z) \Rightarrow r_1(x, z) \text{ if } \forall x, y, z
\end{align*}
\]
Abilities in Inferring the Relation Patterns

- None of existing methods are able to model and infer all the three types of relation patterns

<table>
<thead>
<tr>
<th>Model</th>
<th>Score Function</th>
<th>Symmetry</th>
<th>Antisymmetry</th>
<th>Inversion</th>
<th>Composition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE</td>
<td>$- |W_r,1h - W_r,2t|$</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>TransE</td>
<td>$- |h + r - t|$</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TransX</td>
<td>$- |g_{r,1}(h) + r - g_{r,2}(t)|$</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>DistMult</td>
<td>$\langle h, r, t \rangle$</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>ComplEx</td>
<td>$\text{Re}(\langle h, r, \bar{t} \rangle)$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>RotatE</td>
<td>$- |h \circ r - t|$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
RotatE (Sun et al. 2019)

• A new knowledge graph embedding model RotatE
  • Each relation as a elementwise rotation from the source entity to the target entity in the complex vector space
• RotatE is able to model and infer all the three types of relation patterns
• An efficient and effective negative sampling algorithm for optimizing RotatE
• State-of-the-art results on all the benchmarks for link prediction on knowledge graphs

Relation as Elementwise Rotation in Complex Space

• Representing head and tail entities in complex vector space, i.e., $h, t \in \mathbb{C}^k$

• Define each relation $r$ as an element-wise rotation from the head entity $h$ to the tail entity $t$, i.e.,

$$t = h \circ r,$$

where $|r_i|=1$

• $\circ$ is the element-wise product. More specifically, we have $t_i = h_i r_i$, and

$$r_i = e^{i\theta_r,i},$$

• where $\theta_{r,i}$ is the phase angle of $r$ in the i-th dimension.
Geometric Interpretation

• Define the distance function of RotatE as

\[ d_r(h, t) = ||h^\circ r - t|| \]
Modeling the Relation Patterns with RotatE

• A relation $r$ is **symmetric** if and only if $r_i = \pm 1$, i.e.,

$$\theta_{r,i} = 0 \text{ or } \pi$$

• An example on the space of $\mathbb{C}$

$$r_i = -1 \text{ or } \theta_{r,i} = \pi$$
Modeling the Relation Patterns with RotatE

- A relation $r$ is **antisymmetric** if and only if $r^\circ r \neq 1$

- Two relations $r_1$ and $r_2$ are **inverse** if and only if $r_2 = \overline{r_1}$, i.e.,

  $$\theta_{2,i} = -\theta_{1,i}$$

- A relation $r_3 = e^{i\theta_3}$ is a **composition** of two relations $r_1 = e^{i\theta_1}$ and $r_2 = e^{i\theta_2}$ if only if $r_3 = r_1 \circ r_2$, i.e.,

  $$\theta_3 = \theta_1 + \theta_2$$
Optimization

• Negative sampling loss

\[
L = - \log \sigma(\gamma - d_r(h, t)) - \sum_{i=1}^{k} \frac{1}{k} \log \sigma(d_r(h'_i, t'_i) - \gamma)
\]

• \(\gamma\) is a fixed margin, \(\sigma\) is the sigmoid function, and \((h'_i, r, t'_i)\) is the \(i\)-th negative triplet.
Self-adversarial Negative Sampling

• Traditionally, the negative samples are drawn in an uniform way
  • Inefficient as training goes on since many samples are obviously false
  • Does not provide useful information
• A self-adversarial negative sampling
  • Sample negative triplets according to the current embedding model
  • Starts from easier samples to more and more difficult samples
  • Curriculum Learning

\[ p(h_j', r, t_j'|{(h_i, r_i, t_i)}) = \frac{\exp \alpha f_r(h_j', t'_j)}{\sum_i \exp \alpha f_r(h'_i, t'_i)} \]

• \(\alpha\) is the temperature of sampling. \(f_r(h'_j, t'_j)\) measures the salience of the triplet
The Final Objective

• Instead of sampling, treating the sampling probabilities as weights.

\[ L = - \log \sigma(\gamma - d_r(h, t)) - \sum_{i=1}^{n} p(h'_i, r, t'_i) \log \sigma(d_r(h'_i, t'_i) - \gamma) \]
Experiments: Data Sets

- **FB15K**: a subset of Freebase. The main relation types are *symmetry/antisymmetry* and *inversion* patterns.
- **WN18**: a subset of WordNet. The main relation types are *symmetry/antisymmetry* and *inversion* patterns.
- **FB15K-237**: a subset of FB15K, where inversion relations are deleted. The main relation types are *symmetry/antisymmetry* and *composition* patterns.
- **WN18RR**: a subset of WN18, where inversion relations are deleted. The main relation types are *symmetry/antisymmetry* and *composition* patterns.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#entity</th>
<th>#relation</th>
<th>#training</th>
<th>#validation</th>
<th>#test</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB15k</td>
<td>14,951</td>
<td>1,345</td>
<td>483,142</td>
<td>50,000</td>
<td>59,071</td>
</tr>
<tr>
<td>WN18</td>
<td>40,943</td>
<td>18</td>
<td>141,442</td>
<td>5,000</td>
<td>5,000</td>
</tr>
<tr>
<td>FB15k-237</td>
<td>14,541</td>
<td>237</td>
<td>272,115</td>
<td>17,535</td>
<td>20,466</td>
</tr>
<tr>
<td>WN18RR</td>
<td>40,943</td>
<td>11</td>
<td>86,835</td>
<td>3,034</td>
<td>3,134</td>
</tr>
</tbody>
</table>
Results on FB15k and WN18

- RotatE performs the best
- pRotatE performs similarly to RotatE

<table>
<thead>
<tr>
<th>Model</th>
<th>FB15k MR</th>
<th>FB15k MRR</th>
<th>FB15k H@1</th>
<th>FB15k H@3</th>
<th>FB15k H@10</th>
<th>WN18 MR</th>
<th>WN18 MRR</th>
<th>WN18 H@1</th>
<th>WN18 H@3</th>
<th>WN18 H@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransE [♥]</td>
<td>- .463</td>
<td>.297</td>
<td>.578</td>
<td>.749</td>
<td>-</td>
<td>- .495</td>
<td>.113</td>
<td>.888</td>
<td>.943</td>
<td></td>
</tr>
<tr>
<td>DistMult [♦]</td>
<td>42</td>
<td>.798</td>
<td>-</td>
<td>-</td>
<td>.893</td>
<td>655</td>
<td>.797</td>
<td>-</td>
<td>-</td>
<td>.946</td>
</tr>
<tr>
<td>HolE</td>
<td>- .524</td>
<td>.402</td>
<td>.613</td>
<td>.739</td>
<td>-</td>
<td>- .938</td>
<td>.930</td>
<td>.945</td>
<td>.949</td>
<td></td>
</tr>
<tr>
<td>ComplEx</td>
<td>- .692</td>
<td>.599</td>
<td>.759</td>
<td>.840</td>
<td>-</td>
<td>- .941</td>
<td>.936</td>
<td>.945</td>
<td>.947</td>
<td></td>
</tr>
<tr>
<td>ConvE</td>
<td>51</td>
<td>.657</td>
<td>.558</td>
<td>.723</td>
<td>.831</td>
<td>374</td>
<td>.943</td>
<td>.935</td>
<td>.946</td>
<td>.956</td>
</tr>
<tr>
<td>pRotatE</td>
<td>43</td>
<td>.799</td>
<td>.750</td>
<td>.829</td>
<td>.884</td>
<td>254</td>
<td>.947</td>
<td>.942</td>
<td>.950</td>
<td>.957</td>
</tr>
<tr>
<td>RotatE</td>
<td>40</td>
<td>.797</td>
<td>.746</td>
<td>.830</td>
<td>.884</td>
<td>309</td>
<td>.949</td>
<td>.944</td>
<td>.952</td>
<td>.959</td>
</tr>
</tbody>
</table>
Results on FB15k-237 and WN18RR

- RotatE performs the best
- RotatE performs significantly better than pRotatE
  - A lot of composition patterns on the two data sets
  - Modulus information are important for modeling the composition patterns

<table>
<thead>
<tr>
<th></th>
<th>FB15k-237</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>WN18RR</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MR</td>
<td>MRR</td>
<td>H@1</td>
<td>H@3</td>
<td>H@10</td>
<td>MR</td>
<td>MRR</td>
<td>H@1</td>
<td>H@3</td>
<td>H@10</td>
</tr>
<tr>
<td>TransE [❤️]</td>
<td>357</td>
<td>.294</td>
<td>-</td>
<td>-</td>
<td>.465</td>
<td>3384</td>
<td>.226</td>
<td>-</td>
<td>-</td>
<td>.501</td>
</tr>
<tr>
<td>DistMult</td>
<td>254</td>
<td>.241</td>
<td>.155</td>
<td>.263</td>
<td>.419</td>
<td>5110</td>
<td>.43</td>
<td>.39</td>
<td>.44</td>
<td>.49</td>
</tr>
<tr>
<td>ComplEx</td>
<td>339</td>
<td>.247</td>
<td>.158</td>
<td>.275</td>
<td>.428</td>
<td>5261</td>
<td>.44</td>
<td>.41</td>
<td>.46</td>
<td>.51</td>
</tr>
<tr>
<td>ConvE</td>
<td>244</td>
<td>.325</td>
<td>.237</td>
<td>.356</td>
<td>.501</td>
<td>4187</td>
<td>.43</td>
<td>.40</td>
<td>.44</td>
<td>.52</td>
</tr>
<tr>
<td>RotatE</td>
<td>177</td>
<td>.338</td>
<td>.241</td>
<td>.375</td>
<td>.533</td>
<td>3340</td>
<td>.476</td>
<td>.428</td>
<td>.492</td>
<td>.571</td>
</tr>
</tbody>
</table>
Results on Countries (Bouchard et al. 2015)

• A carefully designed dataset to explicitly test the capabilities for modeling the composition patterns
  • Three subtasks S1, S2, S3
  • From easy to difficult

<table>
<thead>
<tr>
<th></th>
<th>Countries (AUC-PR)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DistMult</td>
</tr>
<tr>
<td>S1</td>
<td>1.00 ± 0.00</td>
</tr>
<tr>
<td>S2</td>
<td>0.72 ± 0.12</td>
</tr>
<tr>
<td>S3</td>
<td>0.52 ± 0.07</td>
</tr>
</tbody>
</table>
Summary

• Modeling relation patterns is critical for knowledge base completion
  • Symmetric/Antisymmetric, Inverse, and composition
• RotatE: define each relation as a **elementwise rotation** from the head entity to the tail entity in the complex vector space
  • Capable of modeling and inferring all the three types of relation patterns
• A self-negative sampling techniques for training knowledge graph embeddings
• State-of-the-art results on all existing benchmark data sets
Software

**LINE:**  
(C++)  
https://github.com/tangjianpku/LINE  
(593 stars, released since 2015.3)

**LargeVis:**  
(C++&Python)  
https://github.com/lferry007/LargeVis  
(459 stars, released since 2016.7)

**RotatE:**  
(Pytorch)  
https://github.com/DeepGraphLearning/KnowledgeGraphEmbedding  
(just released!!)
Outline

• Node Representation Methods
  • LINE, DeepWalk, node2vec

• Graph and High-dimensional Data Visualization
  • LargeVis

• Knowledge Graph Embedding
  • RotatE

• A High-performance Node Representation System
A High-Performance CPU-GPU Hybrid System for Node Embedding (Zhu et al. 2019)

• A specific system designed for node embeddings through algorithm and system co-design
  • CPUs: online random walk generation
  • GPUs: training node embeddings
  • Efficient and effective collaboration strategies between CPUs and GPUs
• 50 times faster than existing systems
• Take only one minute for a network with one million node

Summary

• Node Representation Methods
  • LINE, DeepWalk, node2vec

• Graph and High-dimensional Data Visualization
  • LargeVis

• Knowledge Graph Embedding
  • RotatE

• A High-performance Node Representation System
Thanks!
Contact: jian.tang@hec.ca