Learning Attention-based Embeddings for Relation Prediction in Knowledge Graphs

Intro.

- KGs는 missing relations를 추론하는 것이 가장 중요한 task 중 하나이다.
- 이를 흔히 knowledge base completion 또는 relation prediciton이라고 한다. 이 tasks는 주어진 triple(h, r, t)이 valid 한지 그렇지 않은지에 대한 예측도 수반하고 있다.
  여기서 h,r,t는 각각 head entity, relation, tail entity이다.
- 대표적인 methods는 knowledge embedding based models이 있다.
- 이는 크게 *translational models* 와 *CNN based models* 로 나뉜다.
- translational models는 간단한 operations와 limited parameters를 가지고 대체로 성능이 낮다.
- CNN 기반은 좀 더 복잡한 relations도 embedding이 가능하다.
- 하지만 두 방법 모두 triple independently 하게 진행이되고 그래서 주어진 entity 주변에 있는 풍부하고 잠재적인 relations를 "encapsulate" 하지 못 한다는 단점이 있다.
- 이 논문에서는 그래서 `generalized attention-based graph embedding for relation prediction` 을 제안한다.
- node classification에서 GATs는 node의 1-hop 이웃들에 대하여 가장 연관성이 높은 부분에 집중을 하는(가중치를 더 주는) 아이디어를 이용하고 있다.
- KG에서 relaiton prediciton task를 수행함에 있어서, 이 논문이 제안하는 방법은 *generalizes and extends* 한 attention mechanism 을 보이고 있다. 이는 주어진 entity에서 multi-hop neighborhoods의 entity(node)와 relation(edge) features 모두에 attention을 적용한다.
  1. 주어진 노드 주변에 multi-hop relations를 수집한다.
  2. 다양한 관계속에서 node(entity)의 역할을 encapsulate(통합) 한다.
  3. 의미론적으로 유사한 관계 cluster에 존재하는 기존 지식을 통합한다.

     > to consolidate the existing knowledge present in semantically similar relation clusters
- 모델의 depth 가 증가함에 따라 떨어져있는 entites의 기여도는 지수적으로 감소한다. 이 문제를 해결하기 위해 저자는 `relation compostion`(관계 구조(?))를 이용한다. 이는 auxiliary edge(보조 edge)를 n-hop neighbors에 추가해 주는 것으로, 좀 더 쉽게 entities 사이에 knowledge(information)이 흐를 수 있도록 해준다.
- 이 논문에서 제안하는 architecture는 encoder-decoder model이다.
  encoder : generalized graph attention model
  decoder : ConvKB
- 논문에서 주장하는 contributions
  1. 새로운 graph attention based embeddings를 학습하여 KG에서 relaiton prediciton을 수행 함.
  2. 주어진 entity의 multi-hop neighborhood의 entity와 relation features에 대하여 graph attention을 수행 함.
  3. 다양한 dataset에서 모델을 평가함.

3 Our Approach

3.1 Background

- G = (E, R), E와 R은 각각 entities(nodes) 집합과 relations(edges) 집합을 나타낸다.
- Triple (e_s, r, e_o) 는 G에서 node e_s 와 e_o 사이의 관계 r을 나타낸다.
- scoring function f 는 input triple t=(e_s, r, e_o) 이 주어졌을때, valid triple이 존재할 likelihood 를 f(t)로 나타낸다.

Graph Attention Networks(GATs)

- GAT는 모든 node의 이웃들에 대하여 각각 어느정도의 중요도를 할당할지를 학습하게 된다.
- The input feature set of nodes to a layer is $\mathbf{x} = \{\vec{x_1}, \vec{x_2}, \dots, \vec{x_N}\}$
- A layer produces a transformed set of node feature vectors $\mathbf{x}' = \{\vec{x_1'}, \vec{x_2'}, \dots, \vec{x_N'}\}$

  여기서 $\vec{x_i}$ 와 $\vec{x_i'}$ 는 각각 entity e_i 의 input 과 output embedding이다. 그리고 N은 entities의 개수이다.
- A single GAT layer

  $e_{ij} = a(\mathbf{W}\vec{x_i}, \mathbf{W}\vec{x_j})$
  - 여기서 e_{ij} 는 edge (e_i, e_j) 의 attention value이다.
  - $\mathbf{W}$ 는 linear transformation matrix이다.
  - $a(\cdot)$ 는 *attention function* 이다.
- attention value는 source node e_i 에 대하여 각각의 edge (e_i, e_j) features의 중요도를 나타낸다.
- 연결된 edge 들의 attention values set에 softmax를 취하여 α_{ij} (normalized attention coefficients) 를 구한다.
- GAT layer의 output

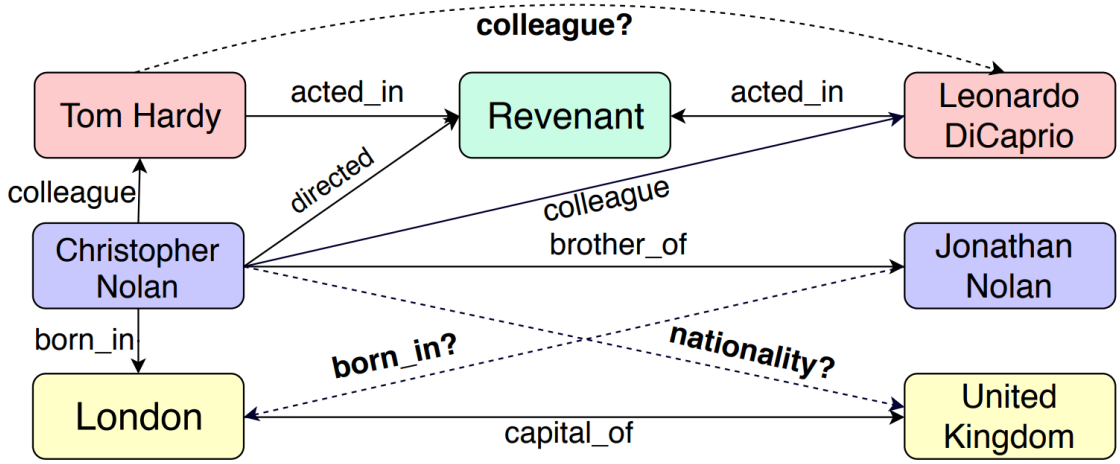
  $\vec{x_i'} = \sigma(\sum_{j \in N_i} \alpha_{ij} \mathbf{W}\vec{x_j})$
  - source node e_i 의 이웃 edges 들의 중요도를 가중치로 선형 변환된 이웃 노들의 hidden states($\mathbf{W}\vec{x_j}$) 에 weight sum을 계산한다 이후 activation function을 통과하고 최종 output을 구한다.
- The multihead attention (K attention heads)

  $\vec{x_i'} = \Big\|_{k=1}^{K} \sigma(\sum_{j \in N_i} \alpha_{ij}^k \mathbf{W}^k \vec{x_j})$
  - $\|$ 는 concatenation이다.
  - α_{ij}^k 는 k-th attentnion mechanism으로 계산된 normalized attention coefficients이다.
  - $\mathbf{W}^k$ 는 k-th attention mechanism의 linear transforamtion matrix 이다.
- 마지막 layer의 output embedding 은 concatenate하지 않고 averaging 한다.

  $\vec{x_i'} = \sigma(\frac{1}{K} \sum_{k=1}^{K} \sum_{j \in N_i} \alpha_{ij}^k \mathbf{W}^k \vec{x_j})$

Relations are Important

- GATs가 우수한 성능을 보이지만 KGs에 바로 적용되기엔 어려움이 있다. 왜냐하면 relation features를 반영하지 않아 불안정하기 때문이다.
- KGs에서 entites는 관련되 relation에 따라 다른 역할을 가지게된다. 예를 들어, Christopher Nolan은 brother 이자 director이다.



- 이러한 이유때문에 저자는 relation과 neighboring node features를 통합하여 embedding하는 attention mechanism을 제안한다.
- 각 layer들은 input으로 두 개의 embedding matrices를 받게된다.
  1. *Entity embeddings* : $\mathbf{H} \in \mathbb{R}^{N_e \times T}$
     - 이 mtrix에서 i-th row 는 entity e_i 의 embedding 이다.
     - N_e 는 전체 entities 개수이다.
     - T 는 각 entity embedding의 feature dimensiond이다.
  2. *relation embeddings* : $\mathbf{G} \in \mathbb{R}^{N_r \times P}$
     - entity embedding 과 유사하다.
- 각 embedding matrices의 output은 아래와 같다.

  $\mathbf{H}' \in \mathbb{R}^{N_e \times T'}, \mathbf{G}' \in \mathbb{R}^{N_r \times P'}$
- 임의의 triple 을 $t_{ij}^k = (e_i, r_k, e_j)$ 라 하자.
- attention mechanism

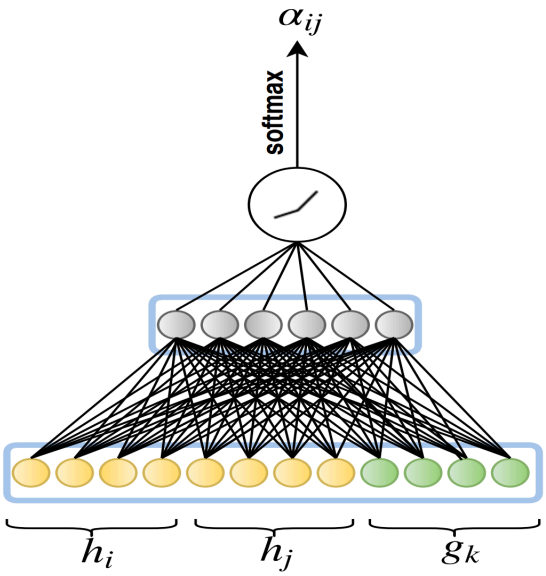


Figure 3: Attention Mechanism

  - Triple t_{ij}^k 의 vector representation

    $\vec{c}_{ijk} = \mathbf{W}_1[\vec{h}_i \parallel \vec{h}_j \parallel \vec{g}_k]$

    - $\vec{h}_i, \vec{h}_j, \vec{g}_k$ 는 각각 entities와 relation의 embedding 이다.
  - The importance of each triple t_{ij}^k 를 b_{ijk} 로 나타낸다.

    $b_{ijk} = \mathrm{LeakyReLU}(\mathbf{W}_2 c_{ijk})$

    - source(head) entity와 연결된 tail entity와 relation에 대한 중요도를 계산한다.
  - relative attention values α_{ijk} 를 구한다.

    $\alpha_{ijk} = \mathrm{softmax}_{jk}(b_{ijk})$

    $= \frac{exp(b_{ijk})}{\sum_{n \in N_i} \sum_{r \in R_{in}} exp(b_{inr})}$

    - N_i 는 entity e_i 의 이웃 entity 들이다.
    - R_{ij} 는 entities e_i 와 e_j 사이에 연결되어 있는 relation set이다.
  - entity e_i의 새로운 embedding

    $\vec{h}_i' = \sigma(\sum_{j \in N_i} \sum_{k \in R_{ij}} \alpha_{ijk} \vec{c}_{ijk})$

  - Multihead attention (M independent attention mechanisms)

    $\vec{h}_i' = \Vert_{m=1}^{M} \sigma(\sum_{j \in N_i} \alpha_{ijk}^m c_{ijk}^m)$

- relation embedding matrix update

  $G' = G\mathbf{W}^R$

  - $W^R \in \mathbb{R}^{T \times T'}$
- The final embedding vectors

  $\vec{h}_i' = \sigma(\frac{1}{M} \sum_{m=1}^{M} \sum_{j \in N_i} \sum_{k \in R_{ij}} \alpha_{ijk}^m c_{ijk}^m)$