Heterogeneous Graph Attention Network

1 HETEROGENEOUS GRAPH ATTENTION NETWORK - WANG ET AL.

1.1 Goals

- We first propose a novel heterogeneous graph neural network based on the hierarchical attention, including node-level and semantic-level attentions

1.2 Preliminaries

- ...

1.3 Challenges

- Real-world graph usually comes with multi-types of nodes and edges
- Different node types may have different attributes

1.4 Previous Work / Citations

- metapath2vec: randomwalk -> skipgram (context based) (negative sampling)
- hin2vec: max likelihood based on path count/probability approximation (but employs a negative sampling like approach)
  - Uses multiple prediction training tasks which learn the latent vectors of nodes and meta-paths simultaneously
- PME projects different types of node into the same relation space and conducts heterogeneous link prediction.
- **This Work:**
  - We introduce node-level attention can learn the importance of meta-path based neighbors for each node in a heterogeneous graph and aggregate the representation of these meaningful neighbors to form a node embedding.
  - To address the challenge of meta-path selection and semantic fusion in a heterogeneous graph, we propose a novel semantic-level attention to automatically learn the importance of different meta-paths and fuse them for the specific task.

1.5 Definitions

- **Semantic-level attention**: aims to learn the importance of each meta-path and assign proper weights to them.
- **Node-level attention**: aims to learn the importance of meta-path based neighbors and assign different attention values to them
- **Heterogeneous network**: $H = \{V, E, A, R, \phi, \psi\}$
  - $v_i \in V$: vertices, $e_{ij} \in E$: edges
  - $\phi(v_i)$: Node type, $\psi(e_{ij})$: Link type
  - A_i^o: Node attribute, U_{ij}^o: Link attribute
- **Meta-Path**: Path $o_1 \rightarrow^{l_1} o_2 \rightarrow^{l_2} ...o_m \rightarrow^{l_{m+1}} o_{m+1}$
  - Where o and l are node/link types
  - Carries semantics (composed relation)

Author's address:

  - Allows computing **multi-modal proximity**
- **Network embedding**: $\Phi : V \rightarrow \mathbb{R}^{|V| \times d}$
- **Heterogenous network embedding**: $\{\Phi_k : V \rightarrow \mathbb{R}^{|V_k| \times d}\}_{k=1}^{K}$
  - where K is number of node types

Notation	Explanation
Φ	Meta-path
$\mathbf{h}$	Initial node feature
$\mathbf{M}_{\phi}$	Type-specific transformation matrix
$\mathbf{h}'$	Projected node feature
e_{ij}^{Φ}	Importance of meta-path based node pair (i, j)
$\mathbf{a}_{\Phi}$	Node-level attention vector for meta-path Φ
α_{ij}^{Φ}	Weight of meta-path based node pair (i, j)
$\mathcal{N}^{\Phi}$	Meta-path based neighbors
Z_{Φ}	Semantic-specific node embedding
q	Semantic-level attention vector
w_{Φ}	Importance of meta-path Φ
β_{Φ}	Weight of meta-path Φ
Z	The final embedding

1.6 Outline / Structure

- Node-level attention: node-level attention can learn the importance of meta-path based neighbors for each node in a heterogeneous graph and aggregate the representation of these meaningful neighbors to form a node embedding.
- Project node features into a common space: $h_i' = \mathbf{M}_{\phi_i} \cdot h_i$

1.6.1 Node-level Embeddings

- Importance of meta-path based node pair (**node-level attention**): $e_{ij}^{\Phi} = att_{node}(h_i', h_j'; \Phi)$
  - att_{node} is MLP and is shared
  - e_{ij}^{Φ} is asymmetric; node level attention can preserve asymmetry
- Inject structural information via **masked attention** (by calculating neighbor weights):
  - $\alpha_{ij}^{\Phi} = \mathrm{softmax}_j\left(e_{ij}^{\Phi}\right) = \frac{\exp\left(\sigma\left(\mathbf{a}_{\Phi}^{\mathrm{T}} \cdot \left[\mathbf{h}_i' \| \mathbf{h}_j'\right]\right)\right)}{\sum_{k \in \mathcal{N}_i^{\Phi}} \exp\left(\sigma\left(\mathbf{a}_{\Phi}^{\mathrm{T}} \cdot \left[\mathbf{h}_i' \| \mathbf{h}_k'\right]\right)\right)}$
    * σ: activation
    * $||$: concatenation operation
    * Referred to as Weight Coefficient of Meta-path node pair
- Meta-path based embedding: aggregated based on neighbors and their weight coefs:
  - $\mathbf{z}_i^{\Phi} = \sigma\left(\sum_{j \in \mathcal{N}_i^{\Phi}} \alpha_{ij}^{\Phi} \cdot \mathbf{h}_j'\right)$

Close up of meta-path based embedding calculation

- Extend node-level attention to multi-head attention:
  - Since heterogeneous graph present the property of scale free, the variance of graph data is quite high
  - Process becomes more stable
  - Repeat node-level attention for K times
  - $\mathbf{z}_i^{\Phi} = ||_{k=1}^{K} \sigma\left(\sum_{j \in N_i^{\Phi}} \alpha_{ij}^{\Phi} \cdot \mathbf{h}_j'\right)$

1.6.2 Semantic-level attention

- Calculate weight of each meta-path node pair: $\left(\beta_{\Phi_1}, \ldots, \beta_{\Phi_P}\right) = att_{\text{sem}} \left(\mathrm{Z}_{\Phi_1}, \ldots, \mathrm{Z}_{\Phi_P}\right)$
- att_{sem}: DNN performing semantic level attention
  - Calculate importance of a meta path (by averaging the pair weights)
    - $w_{\Phi_p} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \mathbf{q}^{\mathrm{T}} \cdot \tanh\left(\mathbf{W} \cdot \mathbf{z}_i^{\Phi_p} + \mathbf{b}\right)$
  - Weight of meta-path is obtained by normalizing the importance of all meta-paths
    - $\beta_{\Phi} = \frac{\exp\left(w_{\Phi_p}\right)}{\sum_{p=1}^{P} \exp\left(w_p\right)}$
    - can be interpreted as the contribution of the meta-path Φ_p for specific task
- Final embedding:
  - $\mathrm{Z} = \sum_{p=1}^{P} \beta_{\Phi_p} \cdot \mathrm{Z}_{\Phi_p}$
- Loss function: minimizing Cross-Entropy over all labeled nodes

1.7 Evaluation

- Does outperform other algs by a (substantial) margin.
- Uses: DBLP, ACM, IMDB
- Evaluates node classification task
- Evaluates for clustering task (apply K-Means afterwards)

1.8 Code

- https://github.com/Jhy1993/HAN
  - Code is a little messy
  - a_{ij}^{Φ} : https://github.com/Jhy1993/HAN/blob/71bac29a07fb8fab908d50a806a7bc38aa6c6611/models/g
  - w_{Φ}: https://github.com/Jhy1993/HAN/blob/71bac29a07fb8fab908d50a806a7bc38aa6c6611/models/g
  - semantic level attention calculation ?
    - https://github.com/Jhy1993/HAN/blob/71bac29a07fb8fab908d50a806a7bc38aa6c6611/models/

1.9 Resources

- …

1.10 Discussion

- That algorithms seems so compute heavy
  - Especially that softmax (which is not adressed in analysis?)
- Query based attention / importance would be cool
  - But that is a transformer like thing
-