GRAPH ATTENTION NETWORKS

Velickovic et al.

ICLR 2018

Presenter: Liyan Xu

Graph Structured Data

- Data (node) has relationship; use graph to represent the relationship.
  - Two types of information available: features and network

- Data that has graph structure by nature: social, biological, citation networks …

- Comparison with grid structured data:
  - Number of neighbors varies
  - Usually no order among neighbors
  - A more general case of grid structure

Graph Structured Data

- Node classification: learn node representation using network information and learn a classifier
  - commonly used in semi-supervised learning
  - analogy to object mask in image

- Graph classification: aggregate node representation to graph representation
  - analogy to image classification

Graph Neural Network (GNN)

- Learn with graph structured data using network information.

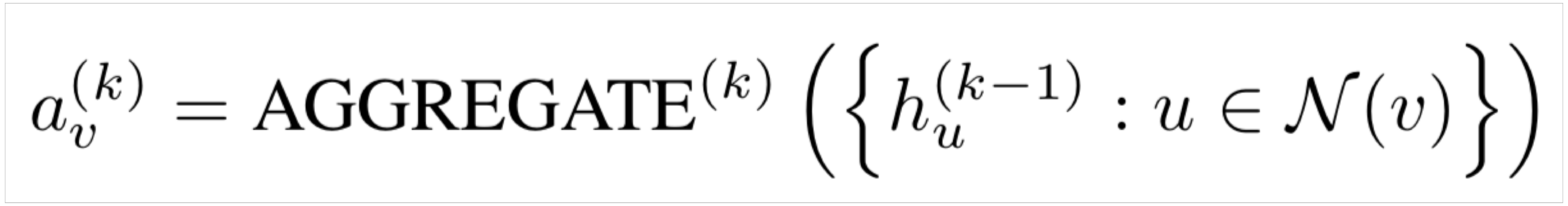
- Motivation: with the success of CNN that automatically extracts features (by using learnable filters locally), it is appealing to apply CNN-like models on graph tasks.

- Use “filter” to aggregate neighbors’ information across all nodes

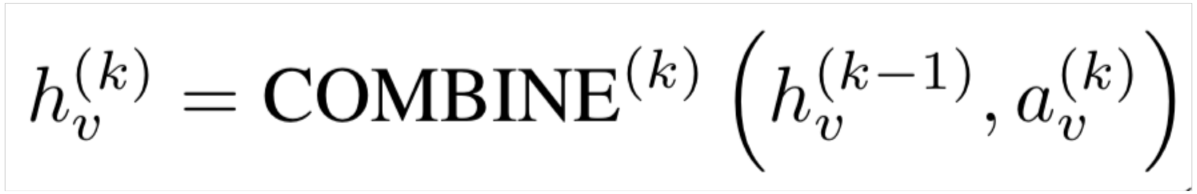
Graph Neural Network (GNN)

- Main idea: learn node’s representation by aggregating its neighbors iteratively.
    - Each node aggregates feature vectors of its neighbors to compute its new feature vector.
    - After k iterations/layers of aggregation, the transformed feature vector captures the structural information.

Graph Neural Network (GNN)



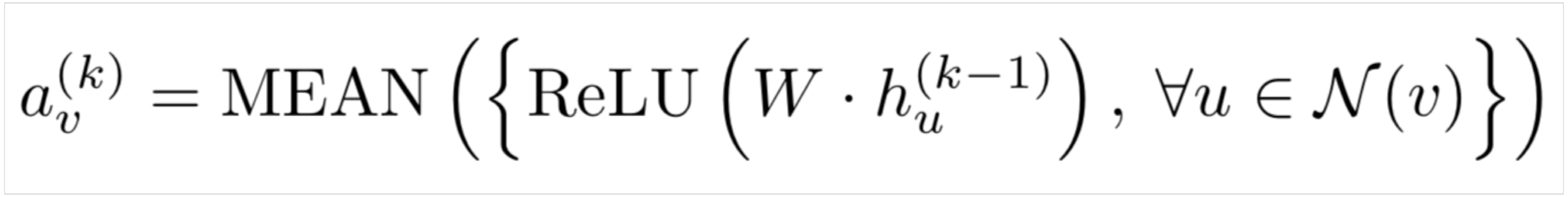
$$a_v^{(k)} = \text{AGGREGATE}^{(k)}\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$



$$h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right)$$

Graph Convolutional Network (GCN)

- Aggregation step:

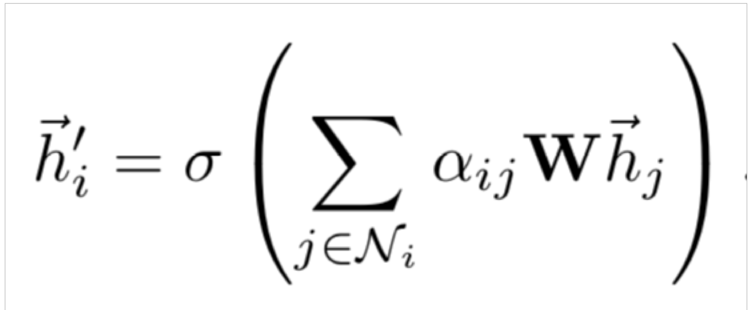


$$a_v^{(k)} = \mathrm{MEAN}\left(\left\{\mathrm{ReLU}\left(W \cdot h_u^{(k-1)}\right), \forall u \in \mathcal{N}(v)\right\}\right)$$

- Limitation:
  - Weights of neighbors are not learned.

Graph Attention Network (GAT)

- Aggregation step: use attention mechanism.



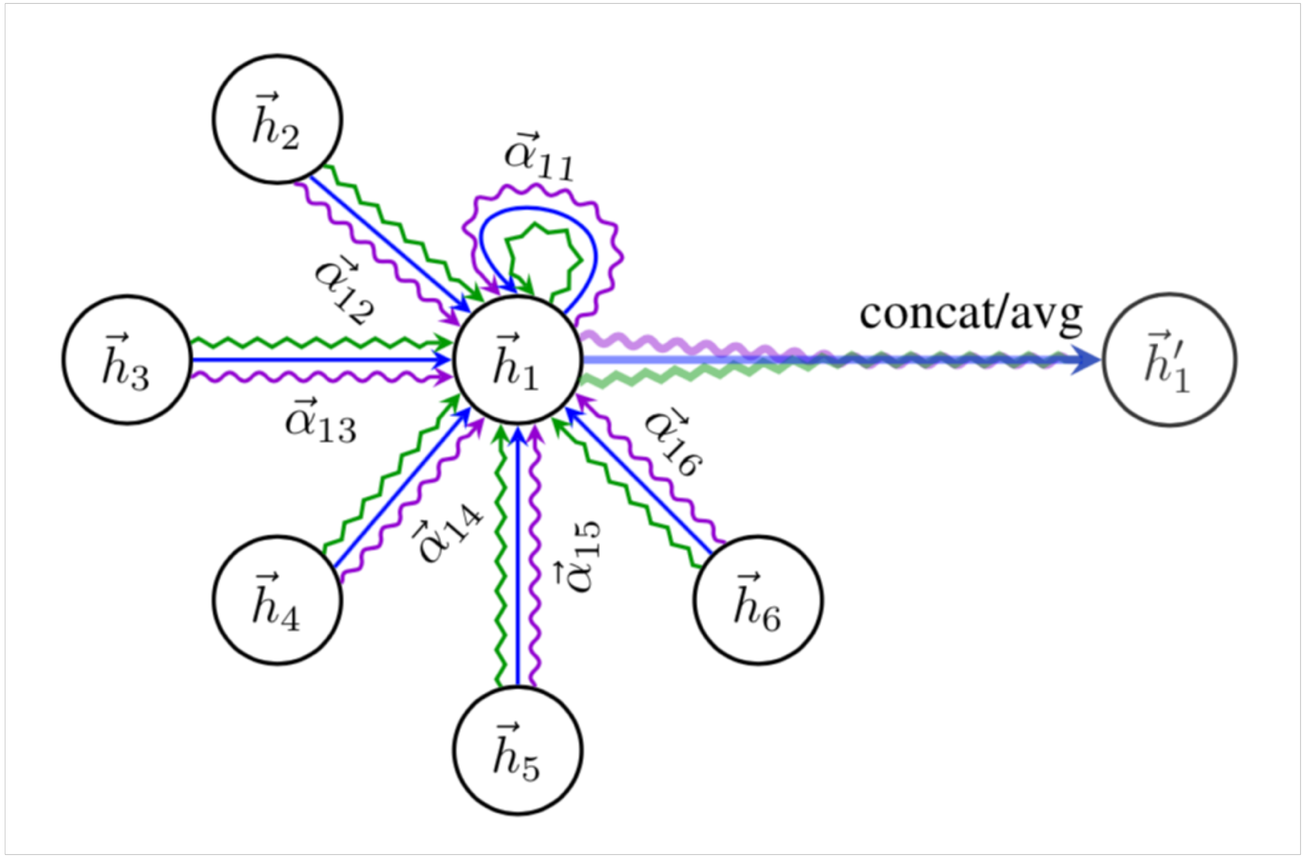
$$\vec{h}'_i = \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}\mathbf{W}\vec{h}_j\right).$$

- 1: apply linear transformation W on all nodes to obtain higher-order feature representation (dimensions can be different).
- 2: compute α as the attention coefficient.
- 3: compute weighted sum of neighbors by attention coefficients.
- 4: apply non-linearity.

Graph Attention Network (GAT)

- Multi-head Attention: use K independent attentional mechanisms (can be computed in parallel)

$$\vec{h}'_i = \operatorname*{\Big\Vert}_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^{k}\mathbf{W}^{k}\vec{h}_j\right)$$



Attention Coefficient

- Shared attention vector ***a*** with non-linearity (single-layer FFNN) to compute coefficients
- Normalization by SoftMax

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$

Performance

- State-of-the-art accuracy on standard citation network datasets and protein-protein interaction dataset.

- Citation network datasets: node classification.
  - Two-layer GAT
  - 8 attention heads
  - 8 features for higher-order representation

Comparison with CNN

- Aggregation by neighbors is a convolution-like (but essentially different) operation.

- Weights are not directly learned; additional computational cost.
  - CNN learns filter weights directly.
  - GAT cannot learn weights directly because neighbor numbers and positions are not fixed.

Learnable Graph Convolution Network (LGCN)

- True convolution operation on graph structured data, by modifying graph structure to grid structure.
- State-of-the-art performance.
- http://arxiv.org/abs/1808.03965