Assume we have graph G

- V is a vertex set
- A is a adjacency matrix (Assume binary)
- $X \in \mathbb{R}^{m \times |V|}$ is a matrix of node features
- v : a node in V
- $N(v)$: the set of neighbors of v

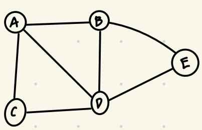
What are the node features?

- Social media: User profile, User image, ...
- Biological networks: Gene expression profiles, gene functional information

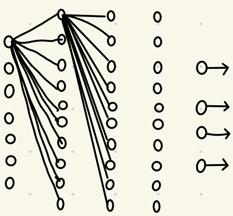
Naive approach

1) Join Adjacency matrix and features
2) Feed them into a deep neural network:

ex)



	A	B	C	D	E	Feat	
A	0	1	1	1	0	1	0
B	1	0	0	1	1	0	0
C	1	0	0	1	0	0	1
D	1	1	1	0	1	1	1
E	0	1	0	1	0	1	0

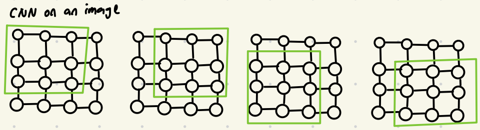


Issues w/ this idea

1) $O(|V|)$ parameters
2) Not applicable to graphs of different sizes
3) Sensitive to node ordering

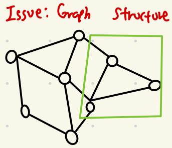
Idea: Convolutional Networks

CNN on an image



Goal for GCN: Generalize convolutions beyond simple matrices to graphs and leverages node features and attributes

Issue: Graph Structure

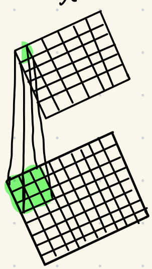


- There's no fixed notion of locality or sliding window on the graph
- Graph is permutation invariant

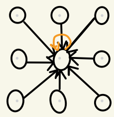
Key idea for GCN: Transform information at the neighbors and combine it ("Message Passing")

- Transform "messages" h_i from neighbors : $W_i h_i$
- Add them up : $\sum_i W_i h_i$

Image



Graph



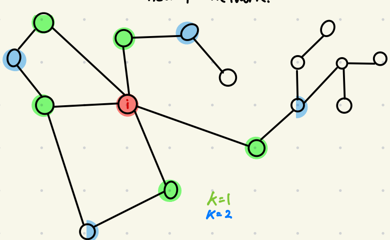
: Transforming information from the neighbors, Combining them, and creating a new kind of message

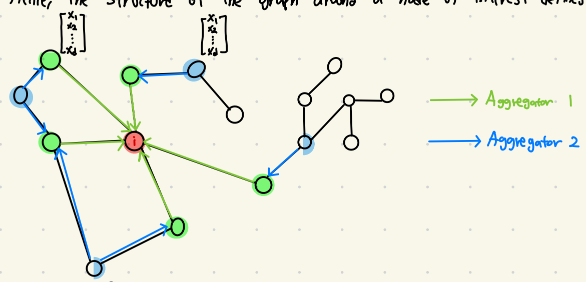
General Concept for Convolutional operator

: It takes 3X3 pixels and apply some transformation and creates a new pixel

Graph Convolutional Networks (GCN)

Idea: Node's neighborhood defines a computation graph. Hence, the structure of the graph around a node of interest defines the structure of a neural network.





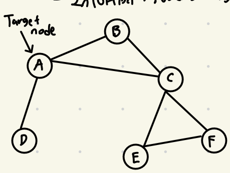
① Determine node computation graph

↳ Defines the architecture or the structure of Neural Network

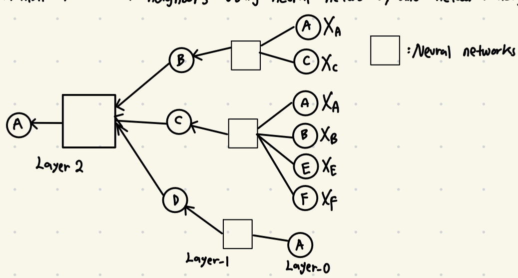
② Propagate and transform information

Aggregate Neighbors

- Intuition : Nodes aggregate information from their neighbors using neural networks, and network neighborhood defines a computation graph



Input graph



☆ Every node defines a computation graph based on its neighborhood.

Deep Models : Many layers

: Model can be arbitrary depth:

- Nodes have embeddings at each layer
- Layer-0 embedding of node v is its input feature, x_n
- Layer-k embedding gets information from nodes that are K hops away.

Basic approach for neighborhood aggregation

- Average information from neighbors and apply a neural network.
  (1) Average messages from neighbors
  (2) Apply neural network

$h_v^0 = x_v$: Initial 0^{th} layer embeddings are equal to node features

$$= \quad h_v^{(l+1)} = \sigma\left(W_l \boxed{\sum_{u \in N(v)} \frac{h_u^{(l)}}{|N(v)|}} + B_l \boxed{h_v^{(l)}}\right), \quad \forall l \in \{0, \dots, L-1\}$$

ex) ReLU

Average of neighbors' previous layer embedding

embedding of v at layer l

$z_v = h_v^{(L)}$

W_l and B_l : Trainable weight matrices

Notation:

① h_v^l : the hidden representation of node v at layer l

② W_K : Weight matrix for neighborhood aggregation

③ B_K : Weight matrix for transforming hidden vector of self

Aggregations can be performed efficiently by matrix operations

① Let $H^{(l)} = [h_1^{(l)} \dots h_{|V|}^{(l)}]^T$

② Then, $\sum_{u \in N(v)} h_u^{(l)} = A \cdot H^{(l)}$

③ Let D be diagonal matrix where

$D_{v,v} = Deg(v) = |N(v)|$

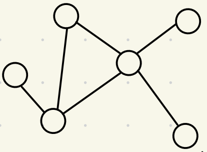
$D_{v,v}^{-1} = \frac{1}{|N(v)|}$

Hence, $\sum_{u \in N(v)} \frac{h_u^{(l-1)}}{|N(v)|} = H^{(l+1)} = D^{-1} A H^{(l)} = \hat{A} H^{(l)}$

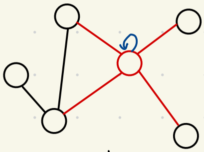
Then, $H^{(l+1)} = \sigma(\hat{A} H^{(l)} W_l^T + H^{(l)} B_l^T)$

GCN in Summary:

Consider this undirected graph



Calculate update for node in red



★ Desirable properties:

① Weight sharing over all locations

② Invariance to permutations

③ Linear complexity $O(E)$

at depth $\ell+1$

Update rule : $h_i^{(\ell+1)} = \sigma\left(h_i^{(\ell)} W_0^{(\ell)} + \sum_{j \in N_i} \frac{1}{c_{ij}} h_j^{(\ell)} W_1^{(\ell)} \right)$

Node index — itself — 1st neighbors.

N_i : Neighbor indices

c_{ij} : norm/constant (trainable/fixed)

Message Passing

Permutation invariant Aggregation.

: $h_i^{(\ell+1)} = \phi\left(h_i^{\ell}, \boxed{}_{j \in N_i} \psi(h_i^{\ell}, h_j^{\ell})\right)$

Learnable functions.

Classification and Link Prediction with GNNs/GCNs

Inputs: $X \in \mathbb{R}^{N \times E}$: Feature matrix.

$A \in \mathbb{R}^{|N| \times |N|}$

$\hat{A} = D^{-1} A$