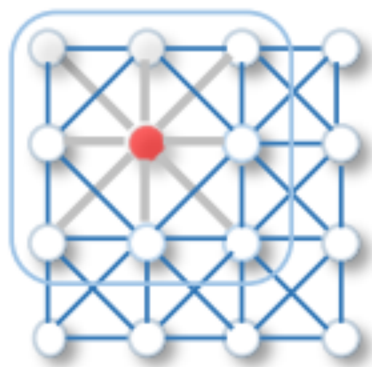
CMSC 510
Regularization Methods for Machine Learning

Graph Neural Networks

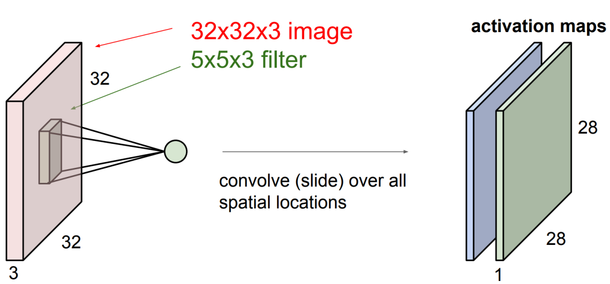
Instructor:
Dr. Tom Arodz

CNN vs GNN - similarities

- Convolutional Neural Networks
  - Input is provided on a regular lattice (pixels in a grid)

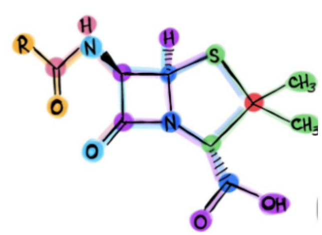


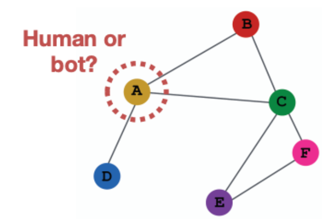
  - At every graph node (pixel) we have a vector of node features
    - In the input layer: RBG colors
    - In subsequent layers: filter activations



- Graph Neural Networks
  - Input is provided on a general graph (typically not regular)

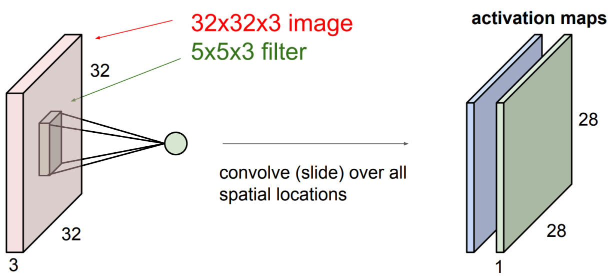
  - At every graph node we have a vector of node features
    - E.g. it's a social network, a node is a person, features describe that person
    - Or a chemical molecule, each node has some properties of that atom





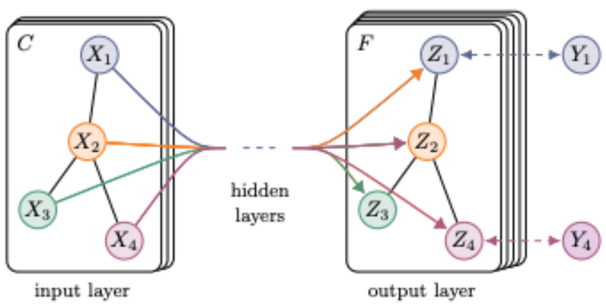
CNN vs GNN - differences

- Convolutional Neural Networks
  - Subsequent layers SOMEWHAT preserve the underlying lattice (pixel grid)
    - But: often there is pooling to reduce the lattice size



  - The output is unrelated to input graph – it's vector of class probabilities

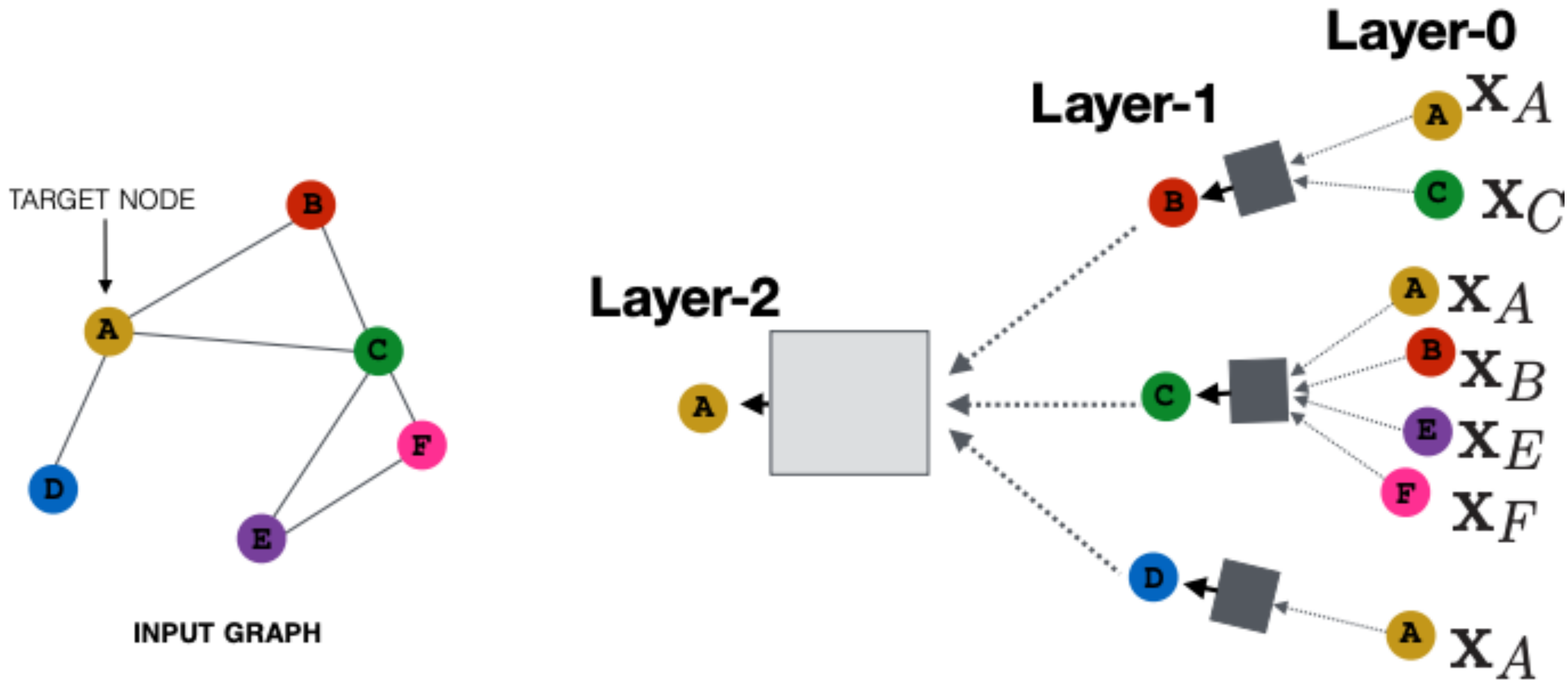
- Graph Neural Networks
  - Subsequent layers TYPICALLY preserve the underlying graph
    - Often there is no pooling, the structure is preserved all they way till the end



  - Often, the output is the same graph as input – we're predicting some missing information (e.g. class) for each node
    - Each node is like a "sample", some have class, some don't, like in semi-supervised learning

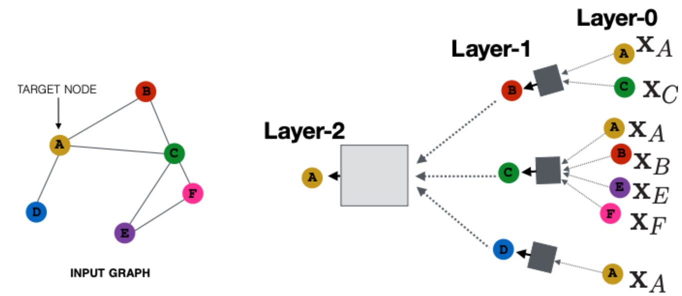
Graph Neural Networks

- Key intuition
  - Information about a node come
    - From its features
    - But also from features of its neighbors in the graph



Graph Neural Networks

- Regularization
  - Like in semi-supervised learning
    - Leverage information from neighbors in a graph
    - Leverage information from unlabeled samples
  - Like in CNNs
    - "Weight sharing" – single set of weights applied at different nodes



$$\boxed{\mathbf{h}_v^0 = \mathbf{x}_v}$$

Initial "layer 0" embeddings are equal to node features

$$\boxed{\mathbf{h}_v^k} = \boxed{\sigma}\left(\mathbf{W}_k \boxed{\sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|}} + \mathbf{B}_k \boxed{\mathbf{h}_v^{k-1}}\right), \quad \forall k > 0$$

previous layer embedding of *v*

kth layer embedding of *v*

non-linearity (e.g., ReLU or tanh)

average of neighbor's previous layer embeddings

http://snap.stanford.edu/proj/embeddings-www/files/nrltutorial-part2-gnns.pdf

Graph Neural Networks

- Challenges
  - In CNNs, filters can have a fixed structure (e.g. 3x3)
    - Graphs are irregular – how to account for that?
  - Graphs (e.g. social networks) are often small-world
    - After a few hops, everything is connected to everything