

E9: 309 ADL 6-1-2021



Housekeeping

- * Midterm project III

- Abstract submission deadline (Jan 10th)

- ✓ Evaluation after final exam (1st week of Feb)

- * Final Exam (as per IISc schedule)

- ✓ Jan 23rd afternoon!



Deep learning on graphs



Lets start with an example

- * Citation dataset
- * Citeseer (collection of papers) or Cora

| | Cora | | CiteSeer |
|------------------------|------|--------|----------|
| Neural Networks | 726 | HCI | 304 |
| Case Based | 285 | IR | 532 |
| Reinforcement Learning | 214 | Agents | 463 |
| Probabilistic Methods | 379 | AI | 115 |
| Genetic Algorithms | 406 | ML | 308 |
| Rule Learning | 131 | DB | 388 |
| Theory | 344 | | |

Table 3: Class distribution for the citation datasets.

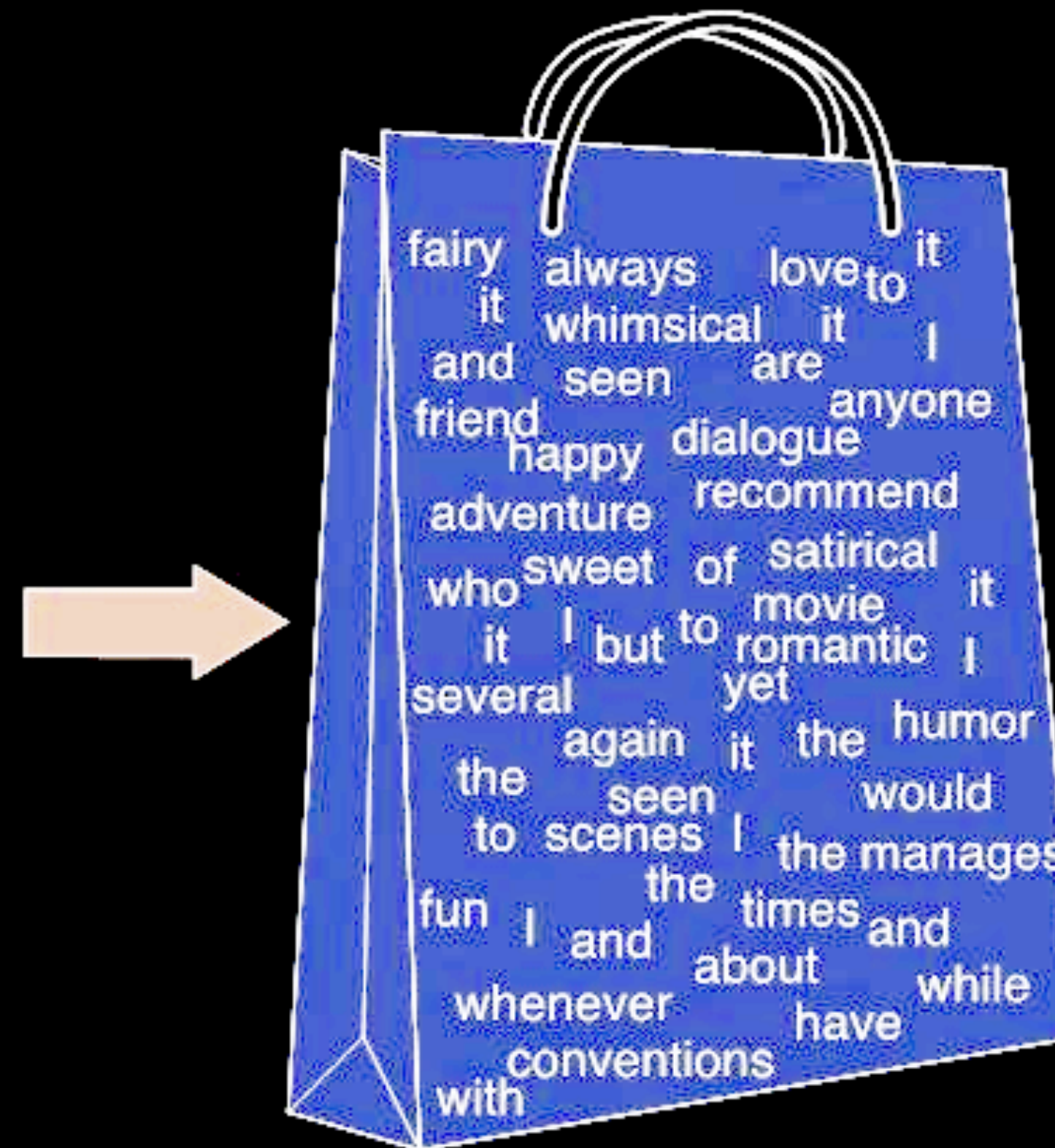


Input features

- * Each paper in the dataset is converted to features (say Bag of Words)

The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

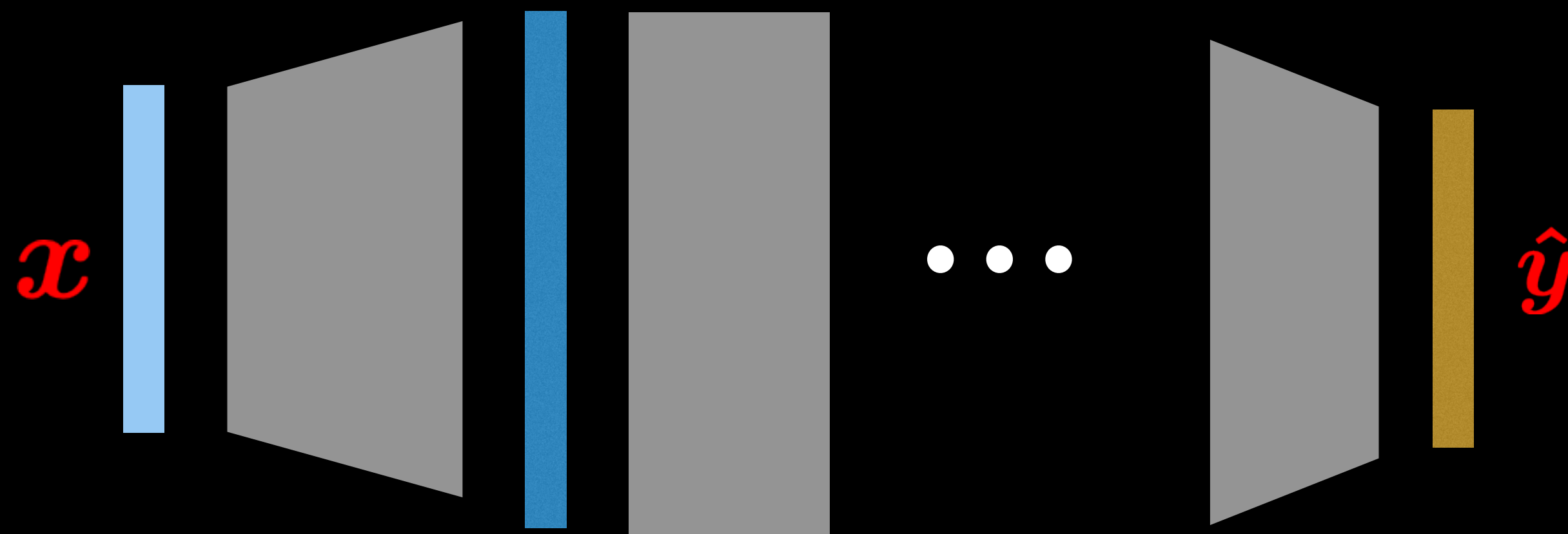


| | |
|-----------|-----|
| it | 6 |
| I | 5 |
| the | 4 |
| to | 3 |
| and | 3 |
| seen | 2 |
| yet | 1 |
| would | 1 |
| whimsical | 1 |
| times | 1 |
| sweet | 1 |
| satirical | 1 |
| adventure | 1 |
| genre | 1 |
| fairy | 1 |
| humor | 1 |
| have | 1 |
| great | 1 |
| ... | ... |



Standard deep learning

- * Find a relationship between the inputs and outputs



- * Find a relationship between the BoW features and output class labels
- * Ignored any dependence between samples.

Given additional information between inputs

* For example in the citation dataset with N papers

→ If we know which papers cite which other papers

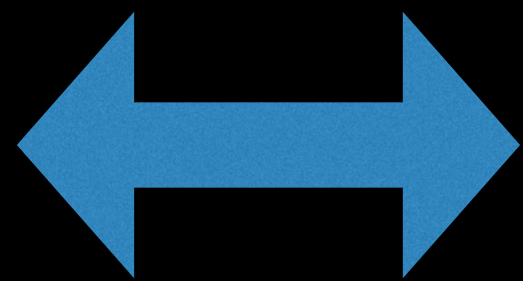
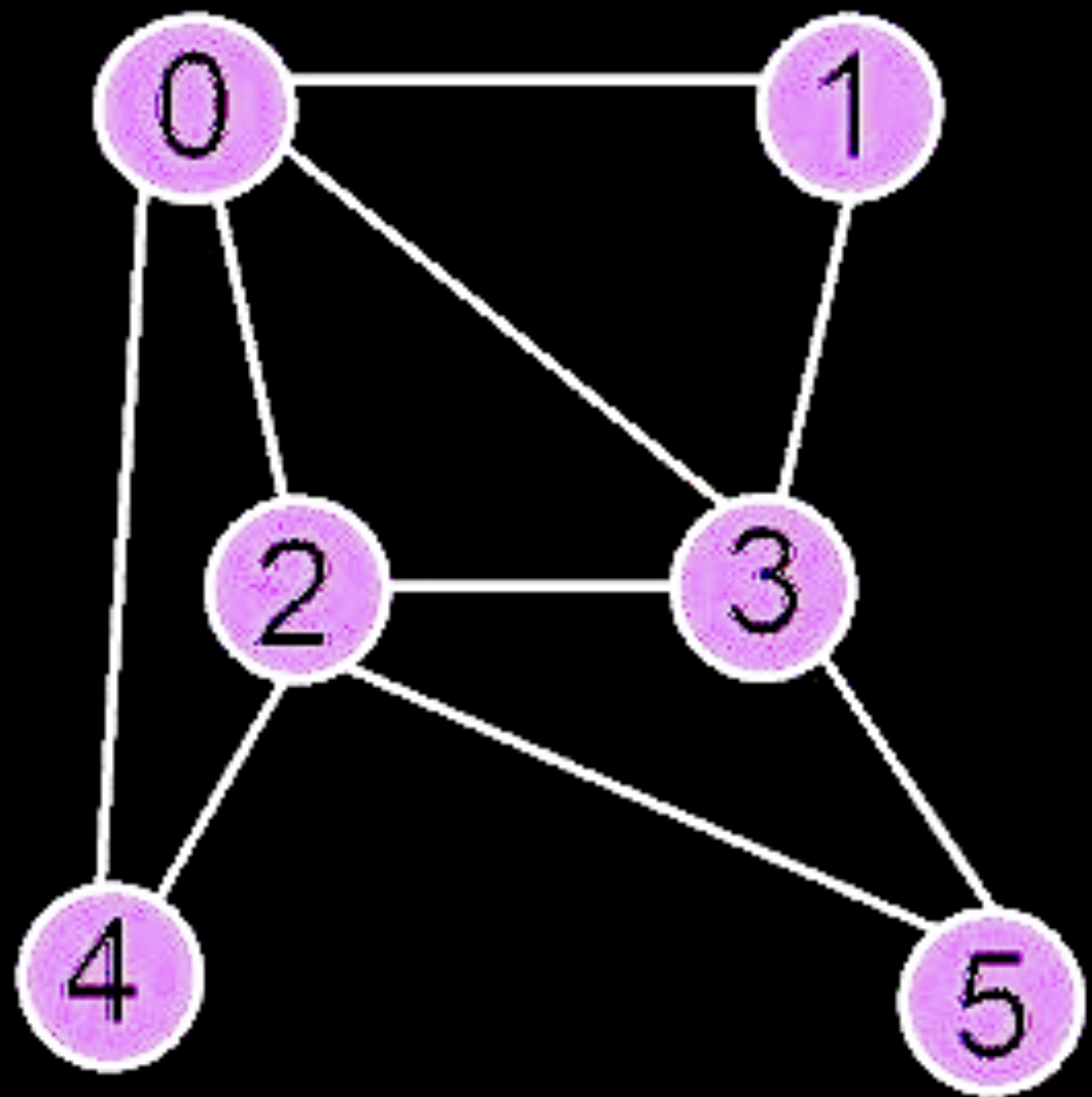
→ This can be represented as a binary matrix of size $N \times N$

→ Relationship among documents exist which are not exploited in standard deep models.

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 2 | 1 | 0 | 0 | 1 | 1 | 1 |
| 3 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4 | 1 | 0 | 1 | 0 | 0 | 0 |
| 5 | 0 | 0 | 1 | 1 | 0 | 0 |

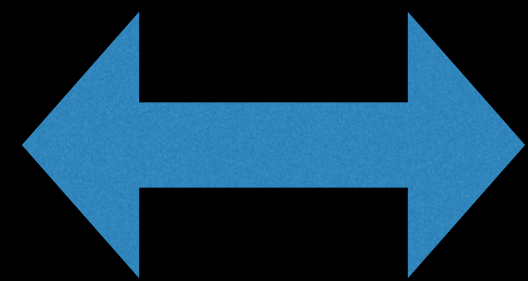
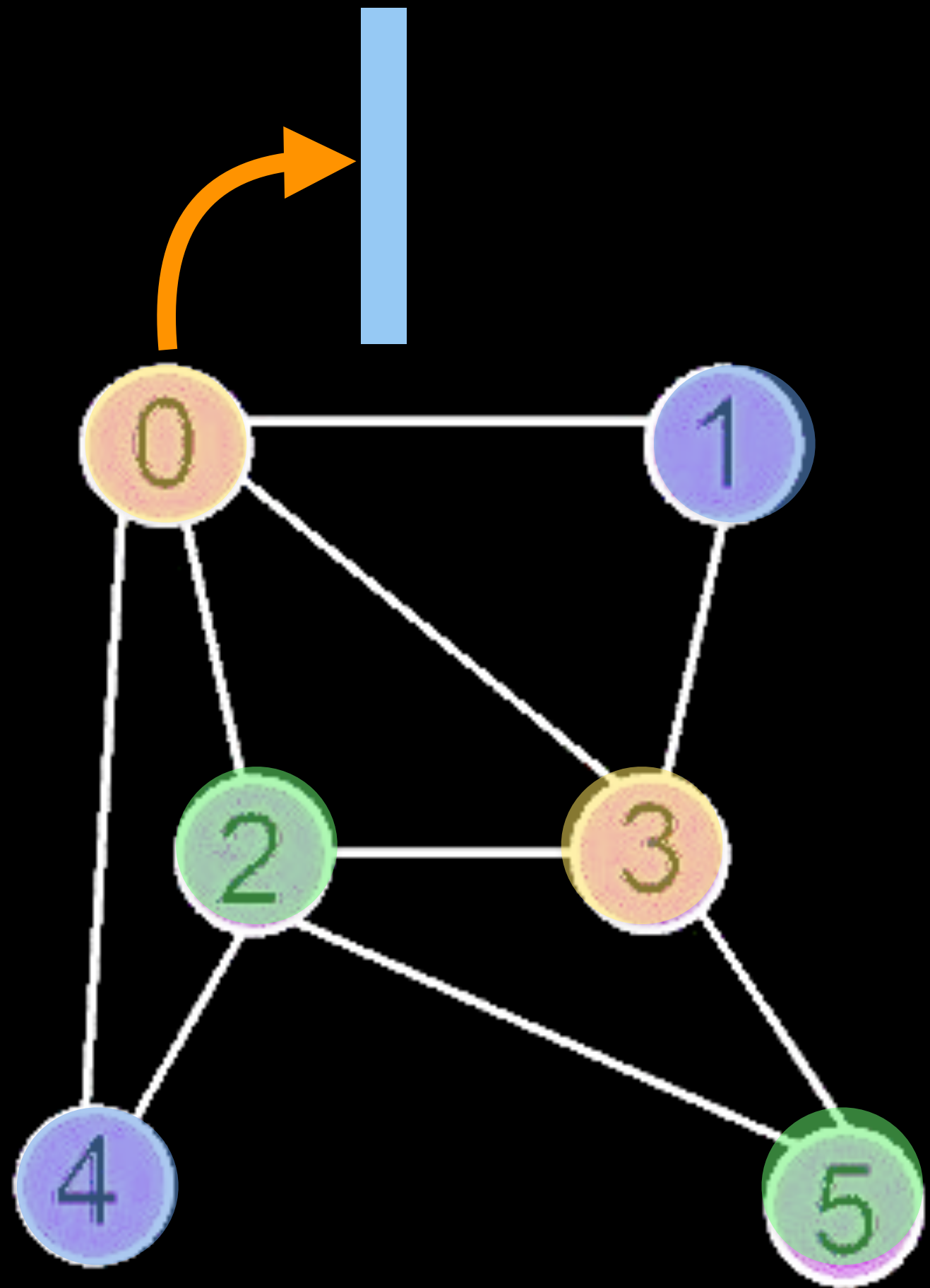


Representing relationships in the input as graph



| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 2 | 1 | 0 | 0 | 1 | 1 | 1 |
| 3 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4 | 1 | 0 | 1 | 0 | 0 | 0 |
| 5 | 0 | 0 | 1 | 1 | 0 | 0 |

Representing relationships in the input as graph



| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 2 | 1 | 0 | 0 | 1 | 1 | 1 |
| 3 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4 | 1 | 0 | 1 | 0 | 0 | 0 |
| 5 | 0 | 0 | 1 | 1 | 0 | 0 |

Graph convolutional networks

SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS

Thomas N. Kipf
University of Amsterdam
T.N.Kipf@uva.nl

Max Welling
University of Amsterdam
Canadian Institute for Advanced Research (CIFAR)
M.Welling@uva.nl

Simplifying Graph Convolutional Networks

Felix Wu^{*1} Tianyi Zhang^{*1} Amauri Holanda de Souza Jr.^{*12} Christopher Fifty¹ Tao Yu¹
Kilian Q. Weinberger¹



Graph definition (Undirected)

- * (\mathbf{V}, \mathbf{E}) denotes the vertices and edges in a graph
 - ✓ $|\mathbf{V}|$ - denotes the number of vertices N
- * $\mathbf{A} = [a_{ij}]$ denote the adjacency matrix of the graph
 - ✓ Similarity or affinity of the vertices. Can also be binary matrix.
 - ✓ Symmetric and typically sparse matrix
- * $\mathbf{D} = \text{diag}[d_1, d_2, \dots, d_N]$ denote the degree matrix

$$d_i = \sum_j a_{ij}$$



Defining graphs

* Input features

$$\mathbf{x}_i \in \mathcal{R}^D \quad i = 1 \dots N$$

* Input feature space

$$\mathbf{X} \in \mathcal{R}^{N \times D}$$

* Hidden layer initialization

$$\mathbf{H}^0 = \mathbf{X}$$



3-steps in Graph convolutional networks

* I. Feature propagation

$$\bar{\mathbf{h}}_i^k = \frac{\mathbf{h}_i}{d_i + 1} + \sum_{j=1}^N \frac{a_{ij}}{\sqrt{(d_i + 1)(d_j + 1)}} \mathbf{h}_j^{k-1}$$

$$\bar{\mathbf{A}} = \mathbf{A} + \mathbf{I}$$

Adding
self
connections

Normalized Adjacency matrix

$$\bar{\mathbf{D}} = \mathbf{D} + \mathbf{I}$$

$$\mathbf{S} = \bar{\mathbf{D}}^{-\frac{1}{2}} \bar{\mathbf{A}} \bar{\mathbf{D}}^{-\frac{1}{2}}$$

$$\bar{\mathbf{H}}^{l+1} = \mathbf{S} \mathbf{H}^l$$

* Intuitively, this step smoothes the hidden representations locally along the edges of the graph and encourages similar predictions among locally connected nodes.



Graph convolutions

* II. Weights and non-linearity

$$\mathbf{H}^{l+1} = \sigma(\bar{\mathbf{H}}^{l+1} \mathbf{W}^{l+1})$$

- Sigma is any non-linearity like sigmoid or ReLU
- A GCN layer is identical to a standard MLP. Each layer is associated with a learned weight matrix, and the smoothed hidden feature representations are transformed linearly.
- Finally, a nonlinear activation function such as ReLU is applied pointwise before outputting feature representation



Graph convolutions

* II. Output layer

$$\hat{\mathbf{Y}} = \text{Softmax}(\mathbf{S}\mathbf{H}^{L-1}\mathbf{W}^L)$$

→ where the predicted model outputs and the true targets are

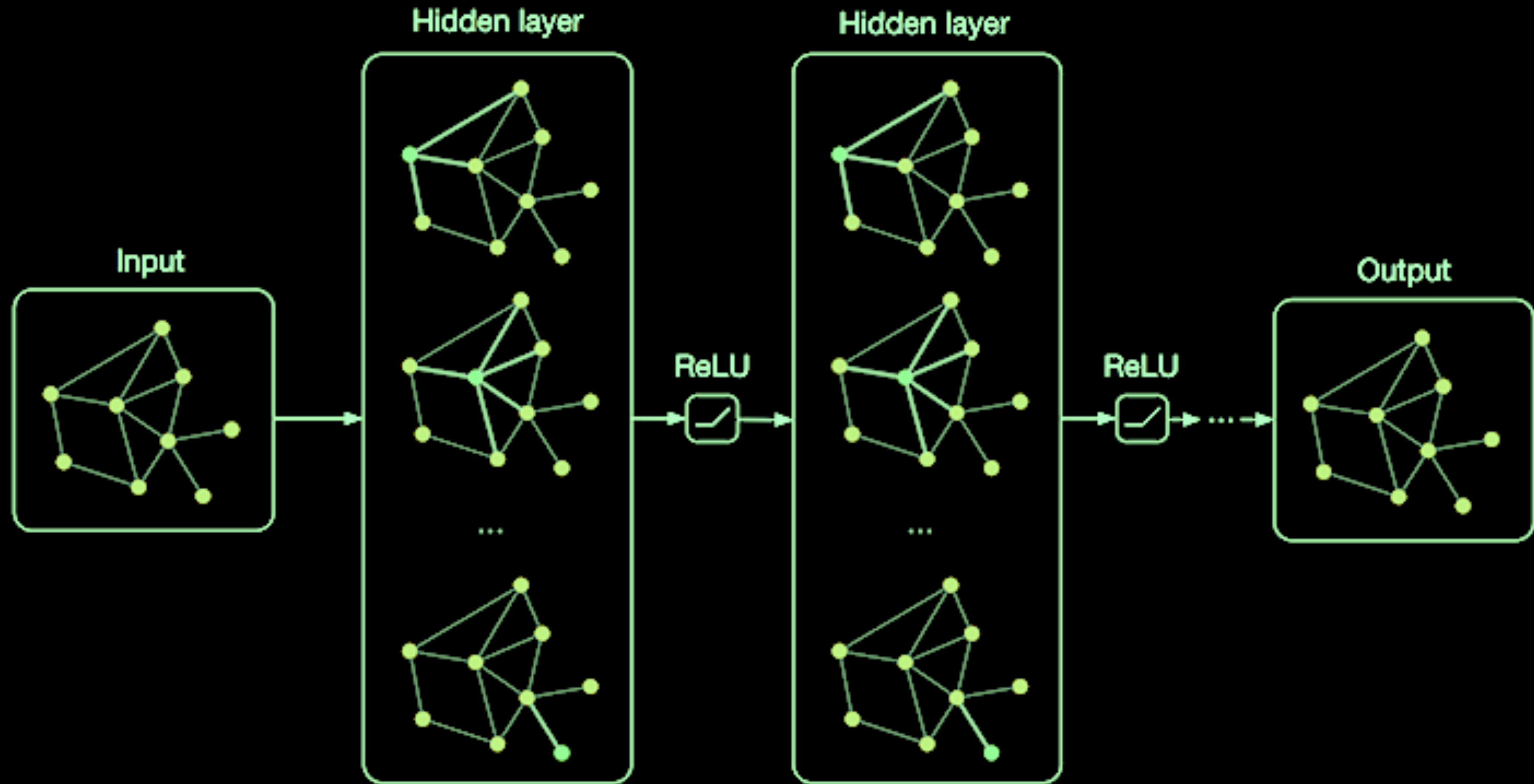
$$(\hat{\mathbf{Y}}, \mathbf{Y}) \in \mathcal{R}^{N \times C}$$

→ The model can be trained using a cross entropy loss

- ◉ or other types of logistic loss functions



Overview of GCN

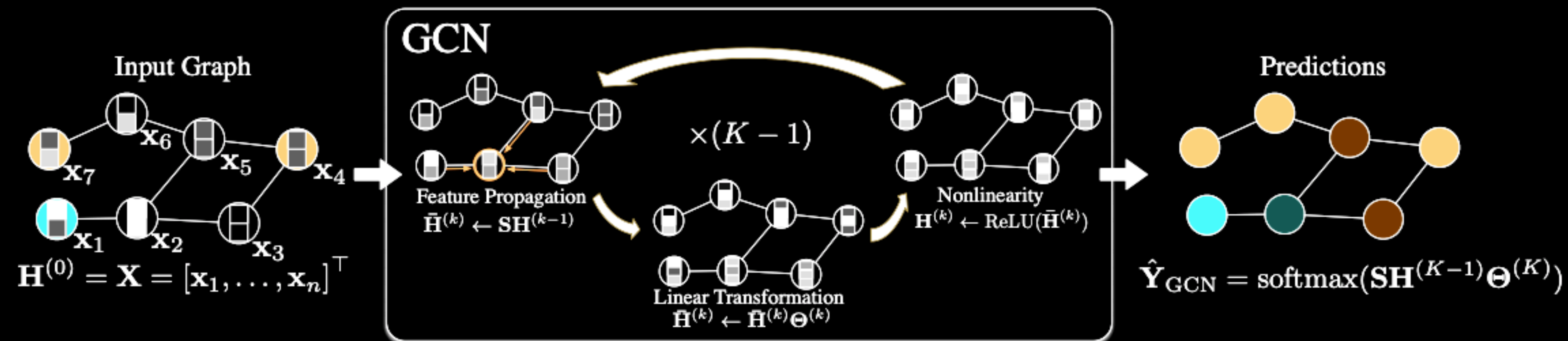


Application

- * Note - GCN cannot be used for predictions of test data like standard MLP
 - The test data must be embedded in the graph and must be learned during training.
- * Semi-supervised learning
 - Only a subset of the nodes have labels
 - ✓ Model can still be learned using loss function defined on the subset of the labelled nodes
 - ✓ Learning in GCN happens as full-batch learning on entire training set.



Semi-supervised learning using GCN



Class +1: ● Class -1: ● Feature Vector: 

Feature Value: 

