[1] USC University of Southern California Information Sciences Institute

[2] Google AI

MixHop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing

Sami Abu-El-Haija[1], Bryan Perozzi[2], Amol Kapoor[2], Nazanin Alipourfard[1], Kristina Lerman[1], Hrayr Harutyunyan[1], Greg Ver Steeg[1], Aram Galstyan[1]

Code: http://github.com/samihaija/mixhop

Slides: http://sami.haija.org/icml19

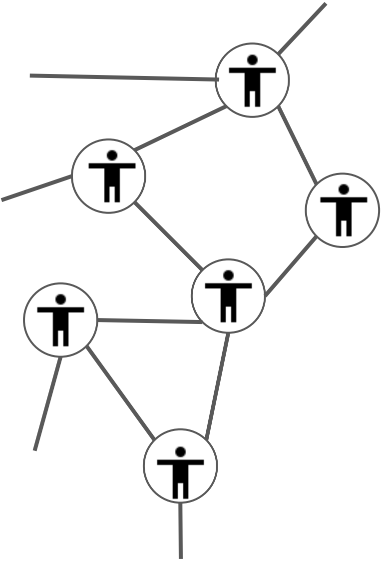
Agenda

- Review Graph Convolutional Networks (GCN)
  - Application Semi-Supervised Node Classification (SSNC)
  - Shortcoming of GCN
- MixHop: Higher-Order GCN
  - Sparsification
- MixHop Results on SSNC

Agenda

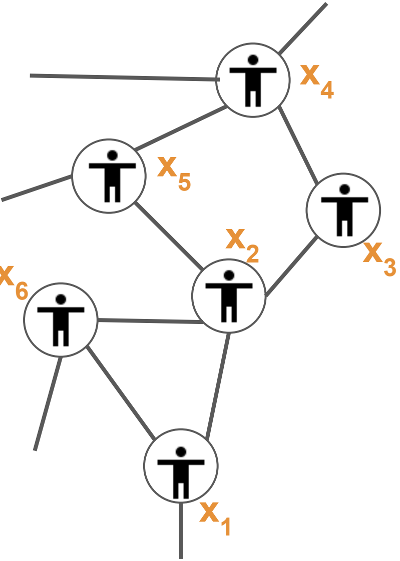
- Review Graph Convolutional Networks (GCN)
  - Application Semi-Supervised Node Classification (SSNC)
  - Shortcoming of GCN
- MixHop: Higher-Order GCN
  - Sparsification
- MixHop Results on SSNC

Graph Convolutional Network (GCN) [1]



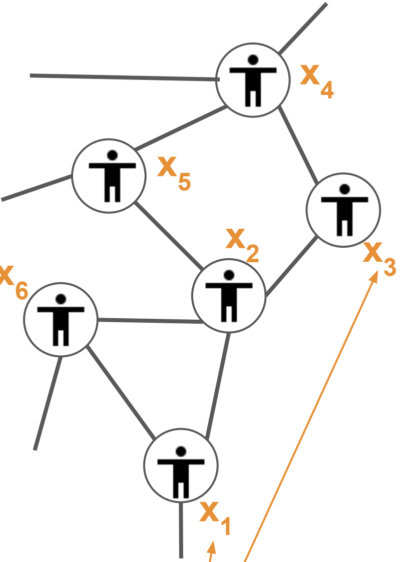
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



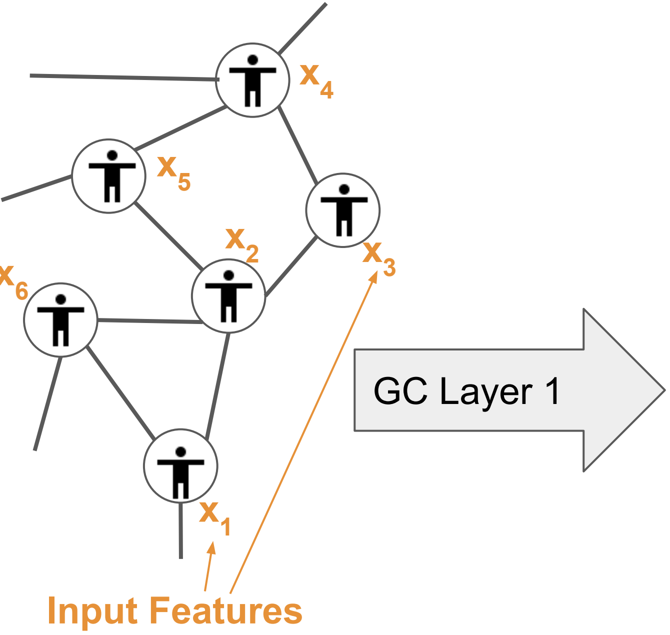
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



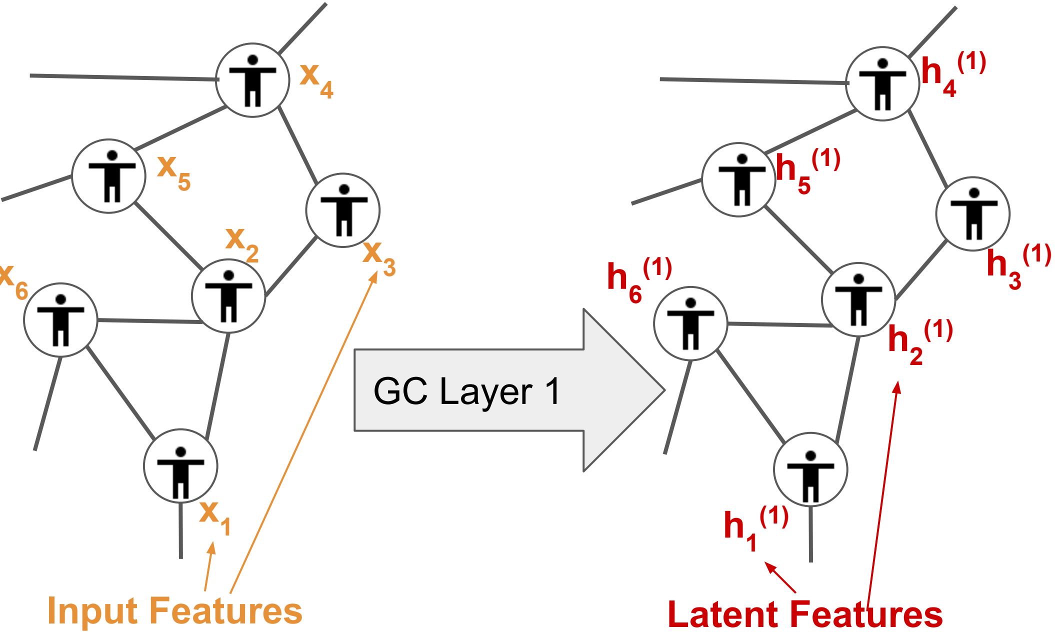
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



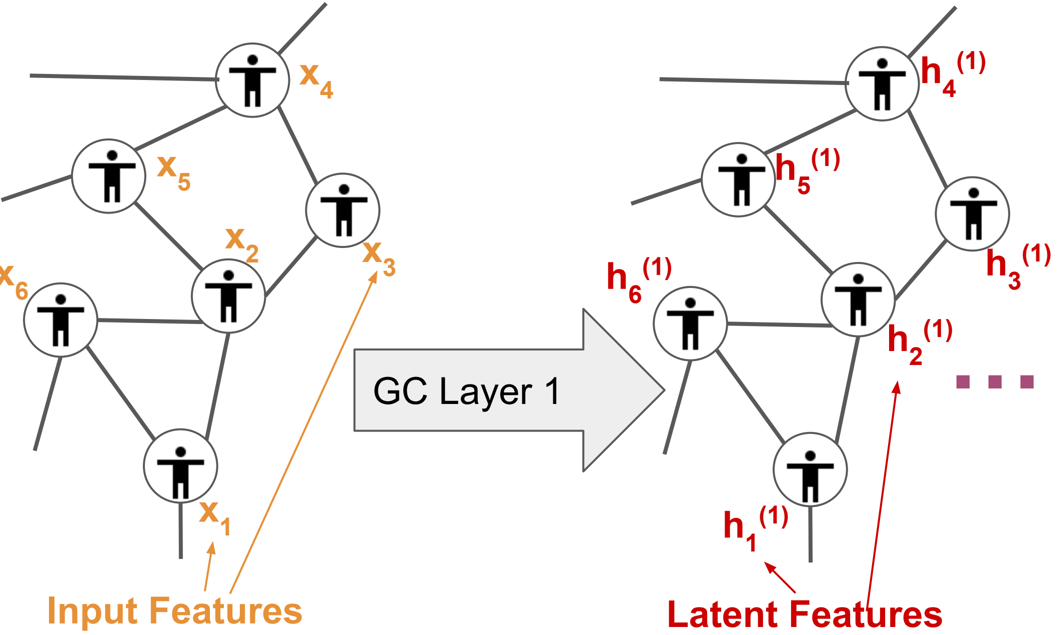
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



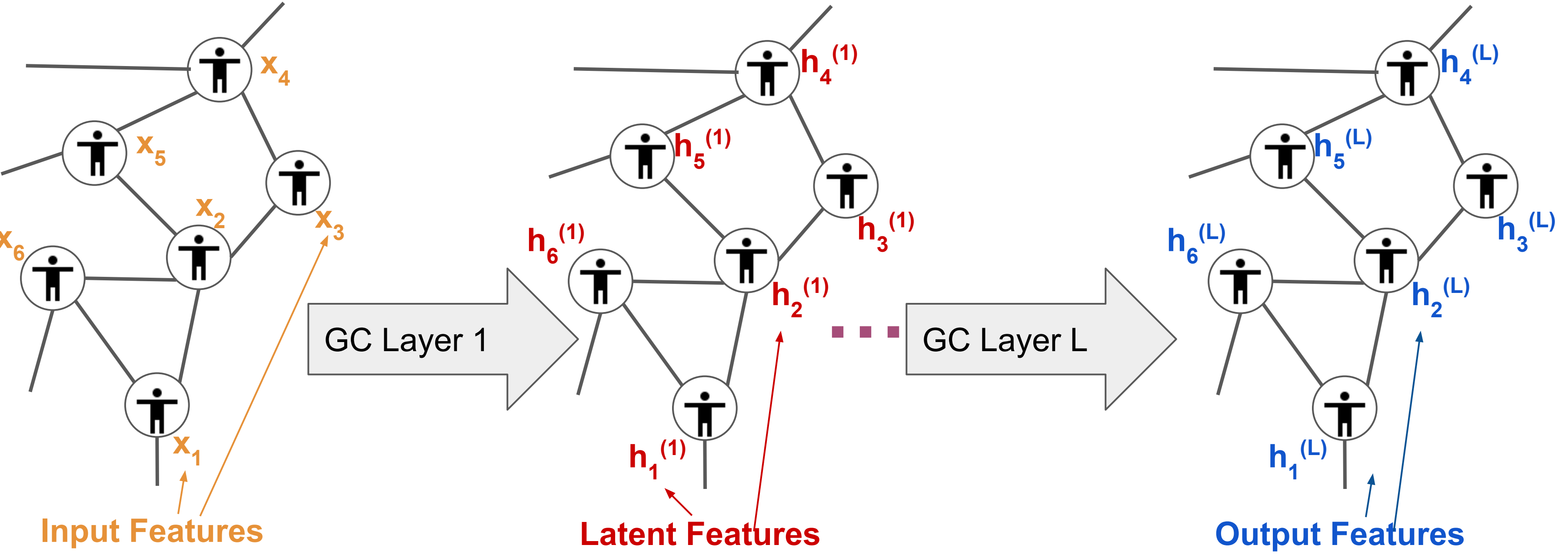
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]

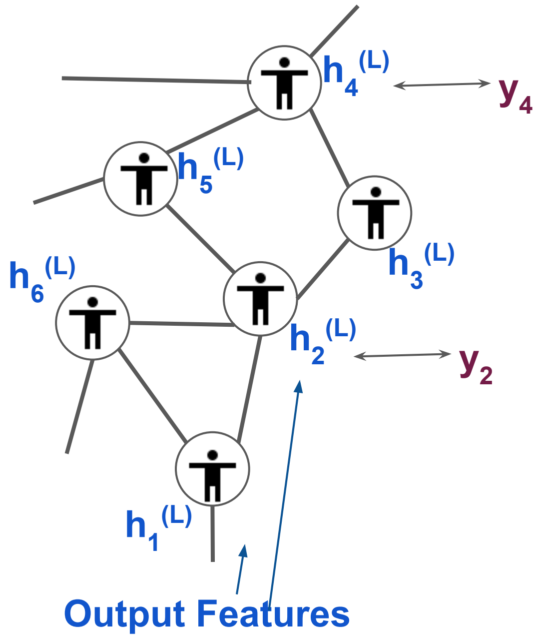


[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]

Train on semi-supervised node classification:

- measure ***Loss*** on labeled nodes (**y_4, y_2**)

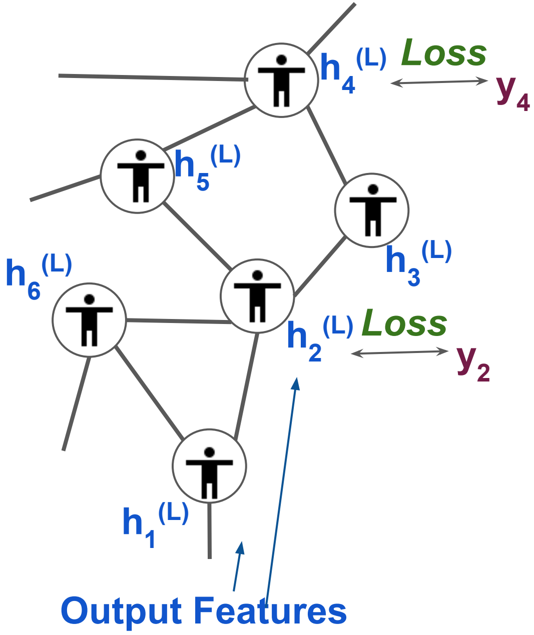


[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]

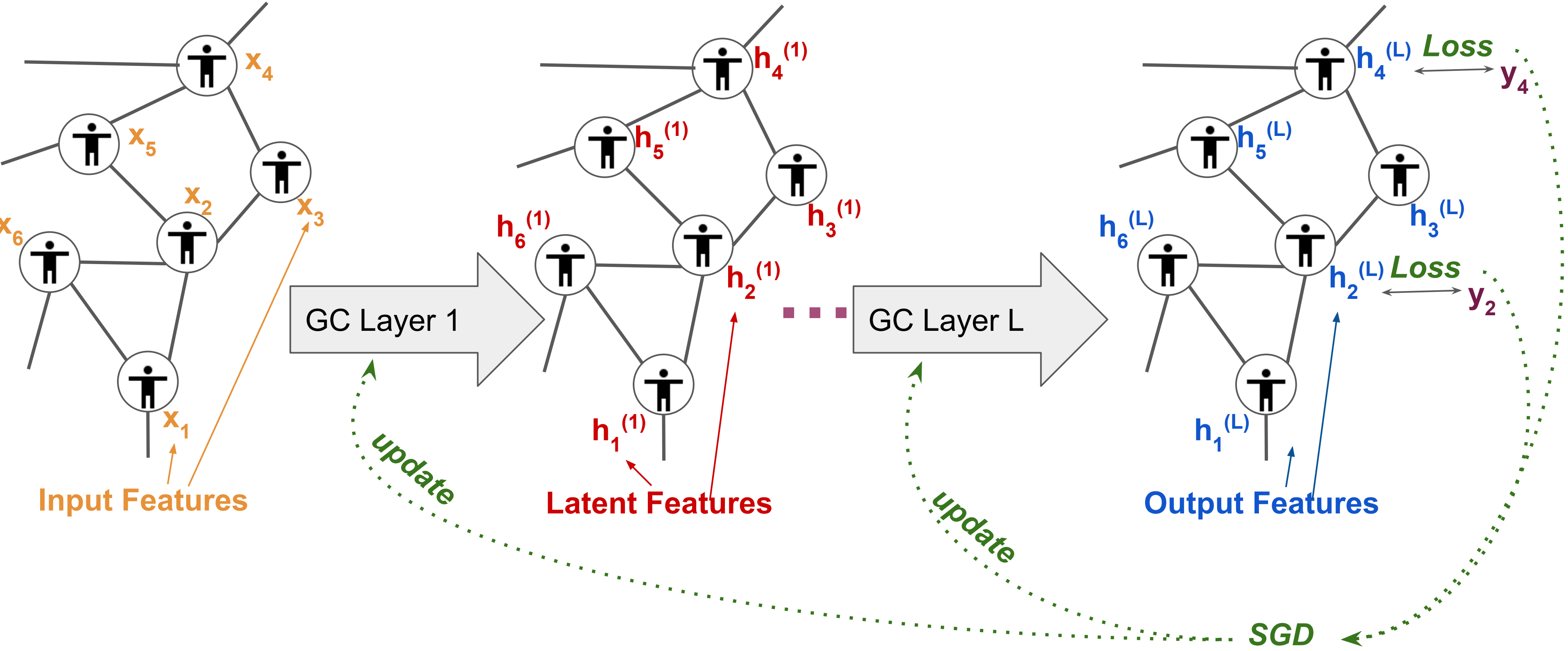
Train on semi-supervised node classification:

- measure ***Loss*** on labeled nodes ($\mathbf{y}_4$, $\mathbf{y}_2$)
- Backprop to learn GC layers.



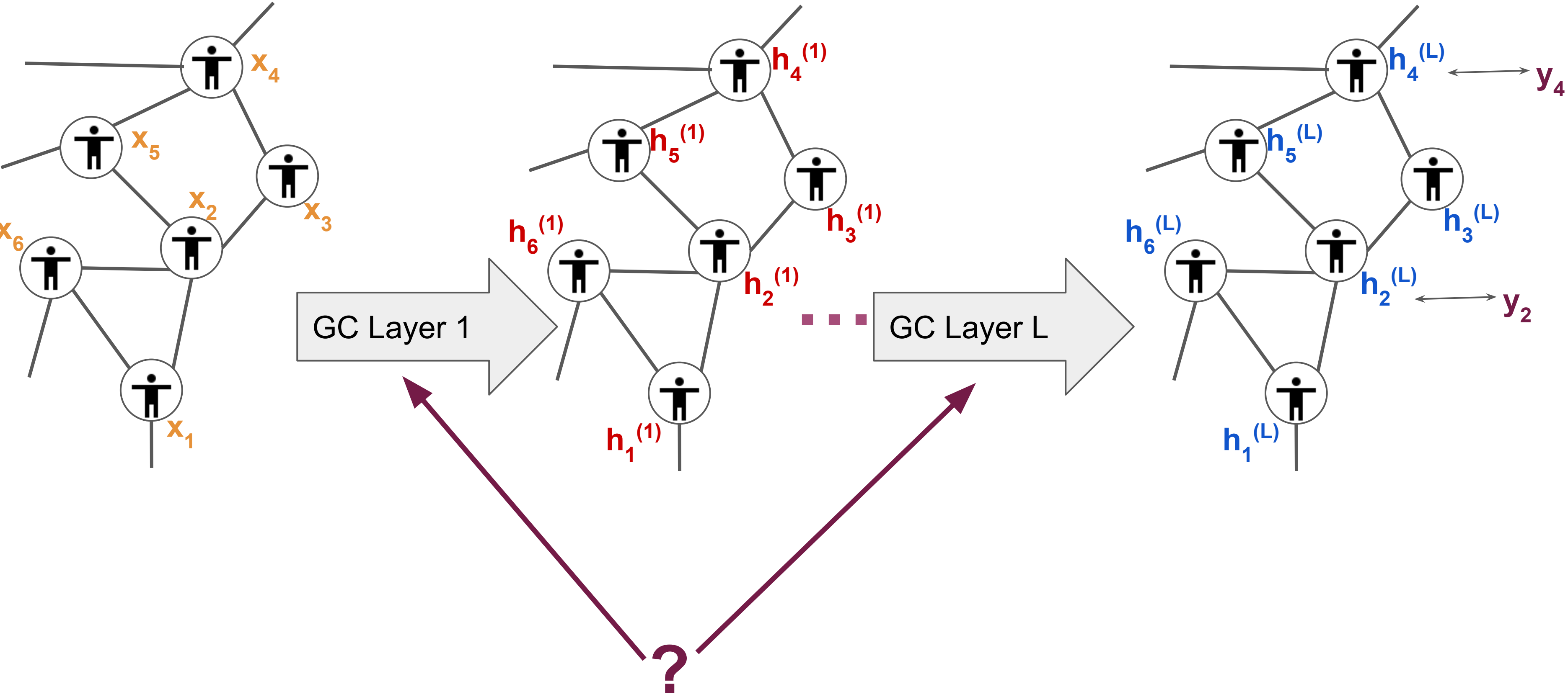
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



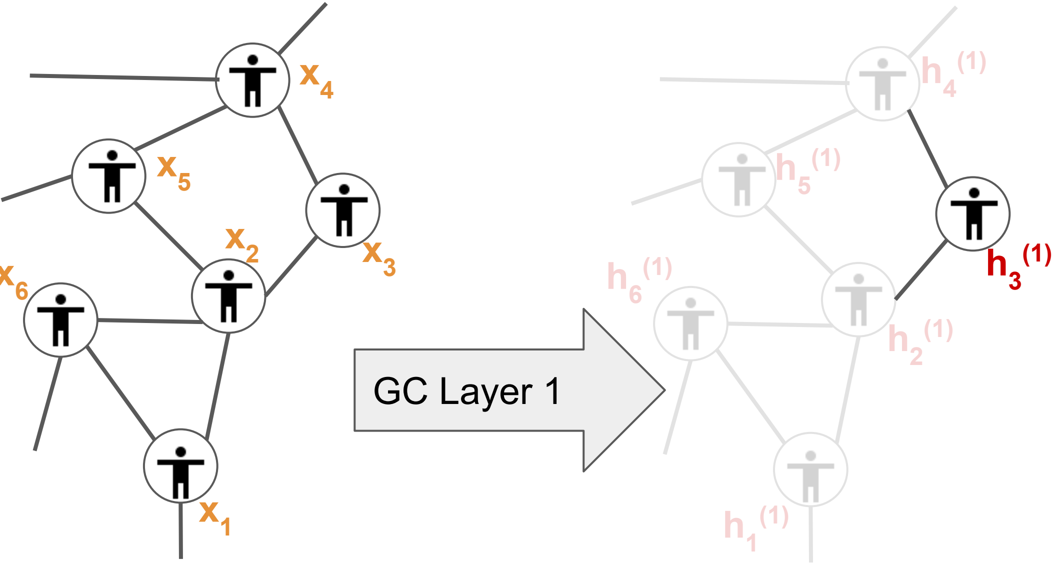
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



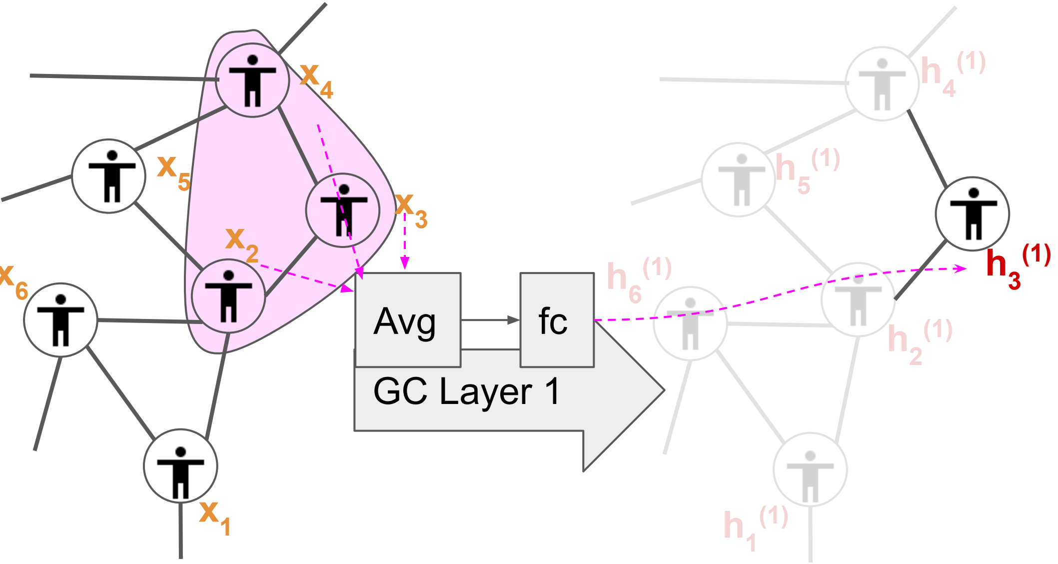
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



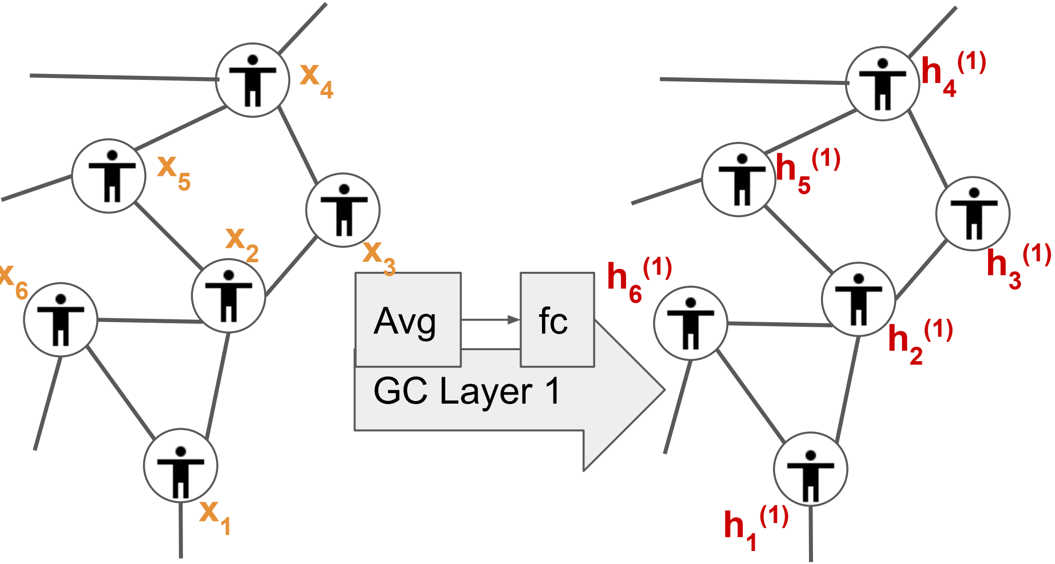
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



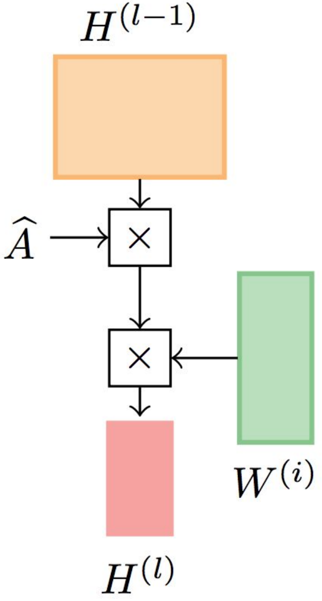
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



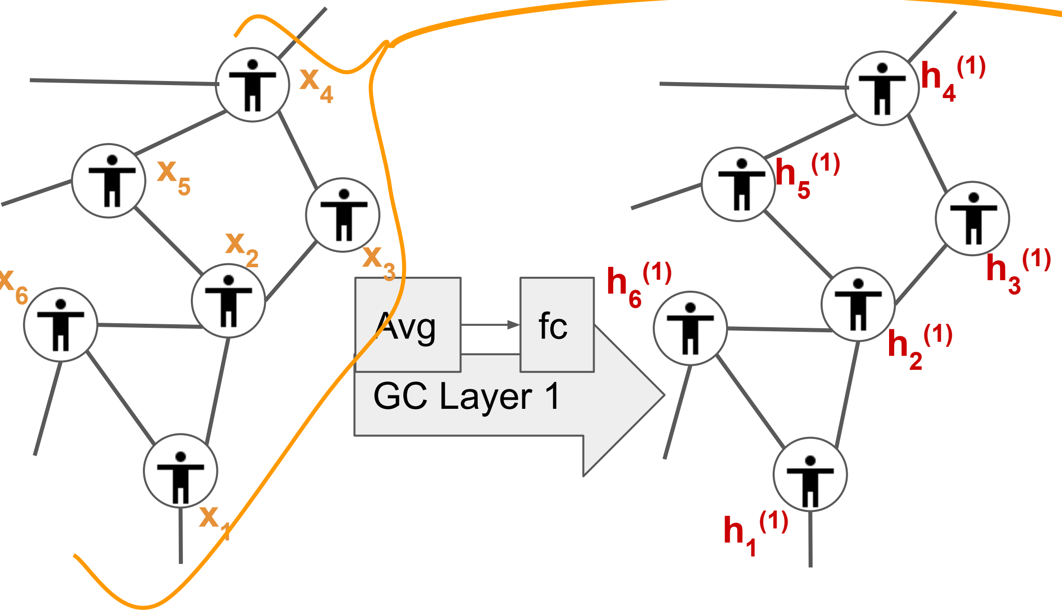
Tensor Graph

$$H^{(1)} = \sigma(\widehat{A} X W^{(1)})$$



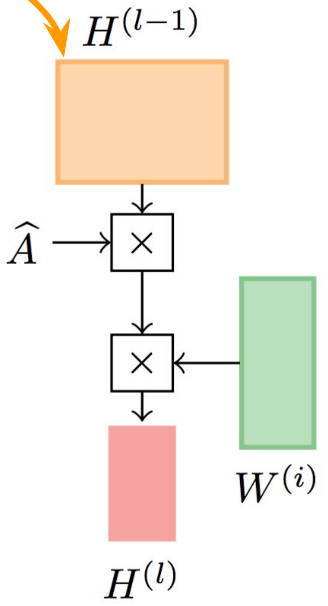
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



Tensor Graph

$$H^{(1)} = \sigma(\widehat{A}XW^{(1)})$$

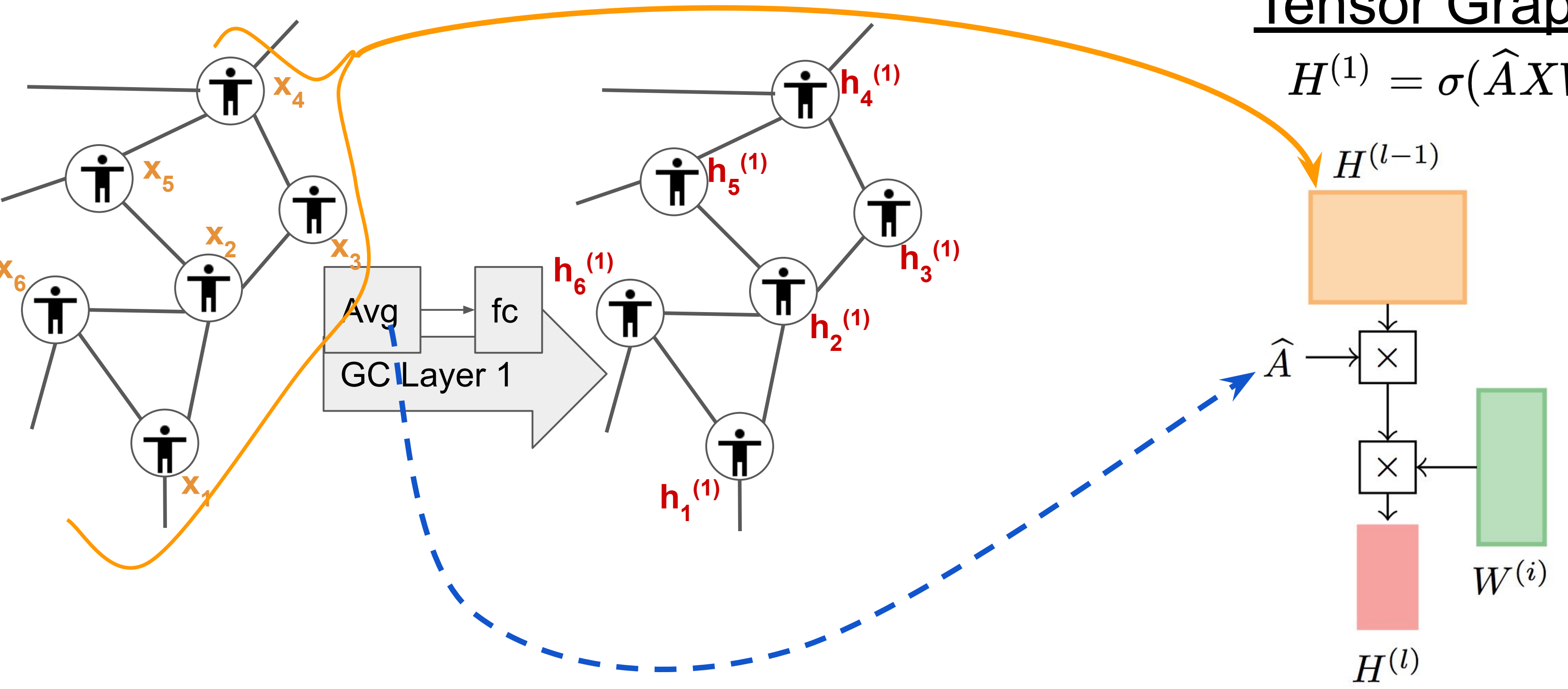


[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]

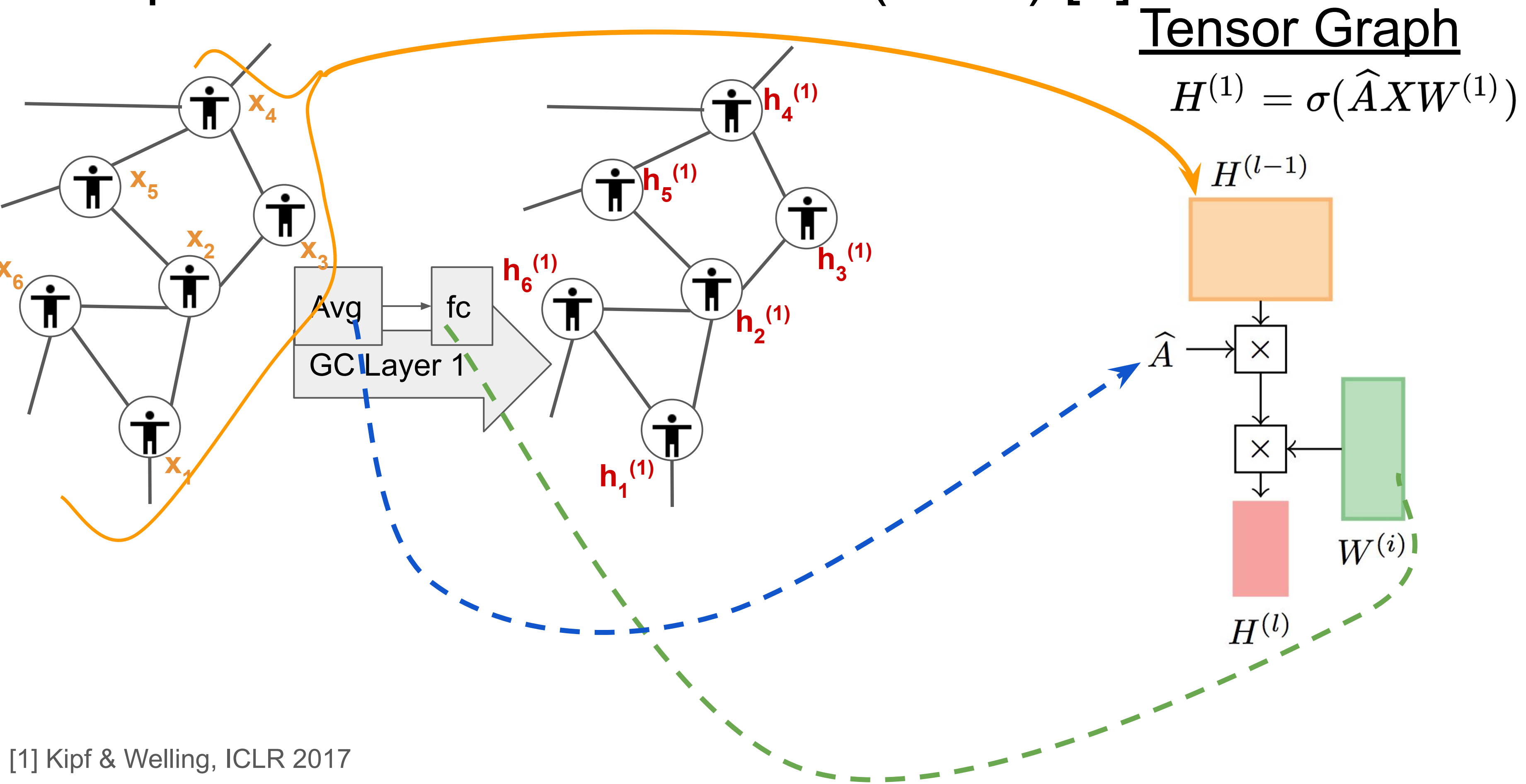
Tensor Graph

$$H^{(1)} = \sigma(\widehat{A}XW^{(1)})$$



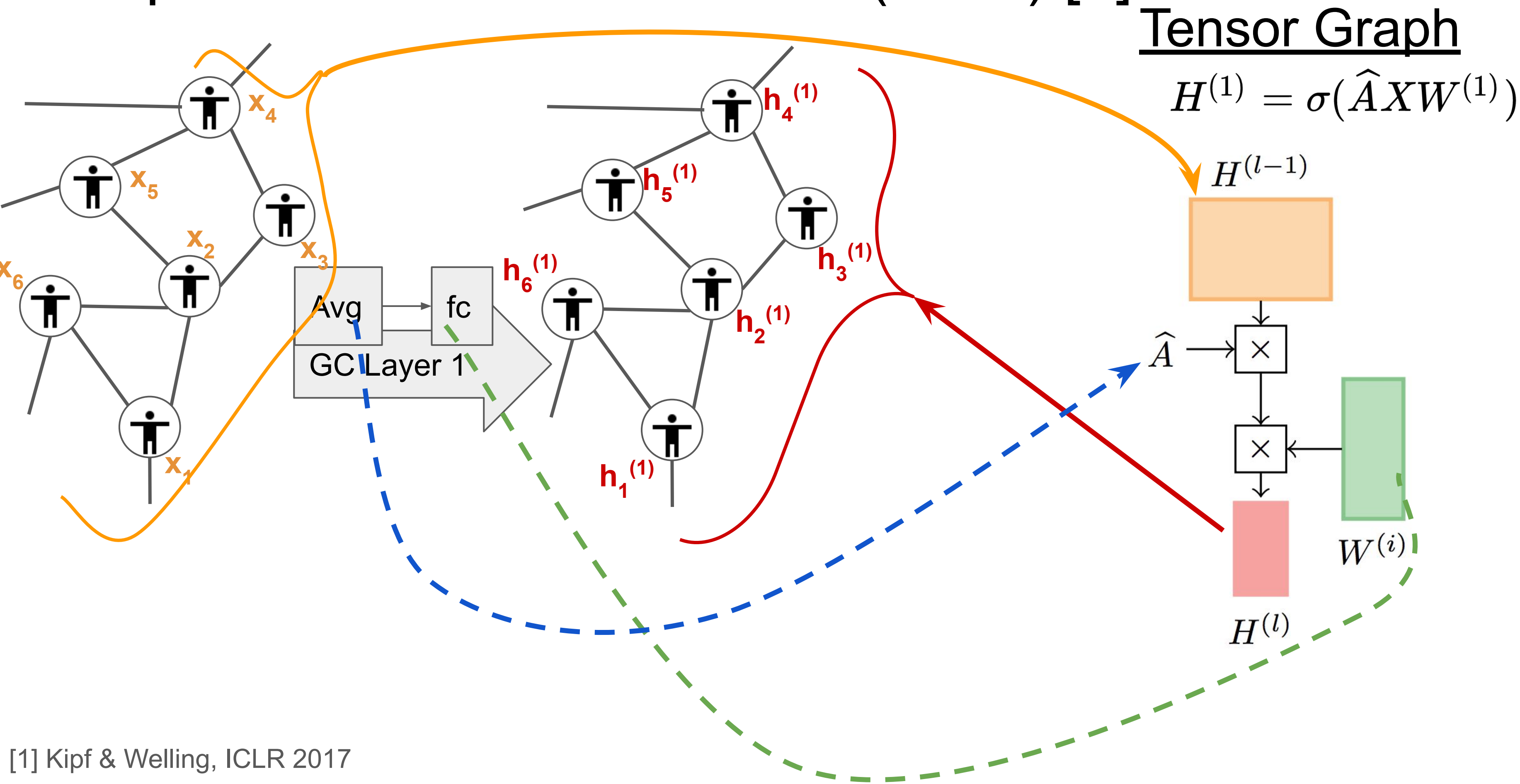
[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]



[1] Kipf & Welling, ICLR 2017

Graph Convolutional Network (GCN) [1]

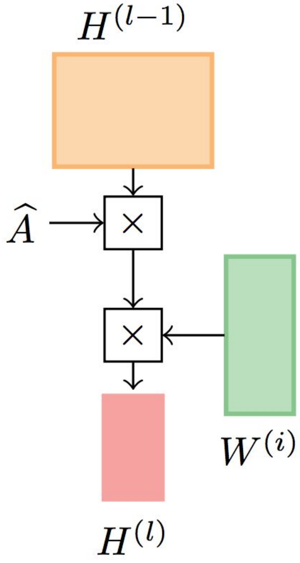


[1] Kipf & Welling, ICLR 2017

Shortcoming of Vanilla GCN

Vanilla GC Layer

$$H^{(1)} = \sigma(\widehat{A} X W^{(1)})$$



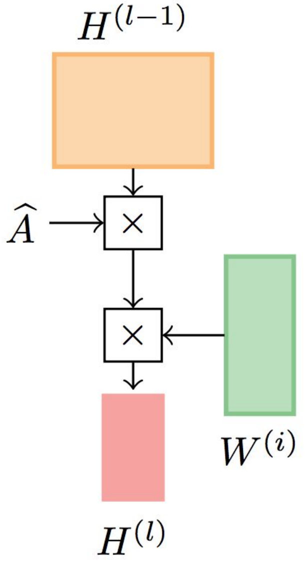
[1] Kipf & Welling, ICLR 2017

Shortcoming of Vanilla GCN

😃 fc is shared ⇒ inductive

Vanilla GC Layer

$$H^{(1)} = \sigma(\widehat{A}XW^{(1)})$$



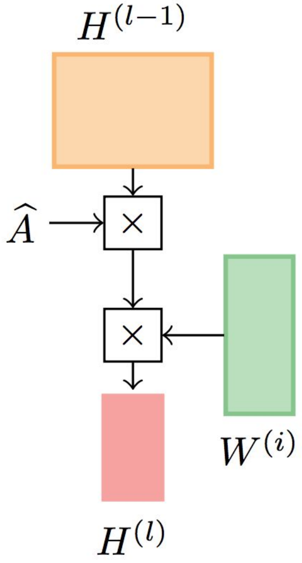
[1] Kipf & Welling, ICLR 2017

Shortcoming of Vanilla GCN

- 😀 fc is shared ⇒ inductive
- 😢 Appendix Experiments of [1] shows no gains beyond 2 layers

Vanilla GC Layer

$$H^{(1)} = \sigma(\widehat{A}XW^{(1)})$$



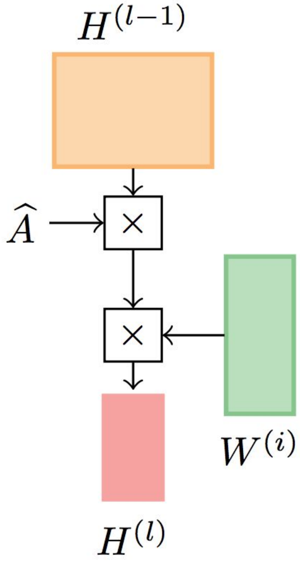
[1] Kipf & Welling, ICLR 2017

Shortcoming of Vanilla GCN

- 😀 fc is shared ⇒ inductive
- 😢 Appendix Experiments of [1] shows no gains beyond 2 layers
- 😢 cannot mix neighbors from various distances in arbitrary linear combinations

<u>Vanilla GC Layer</u>

$$H^{(1)} = \sigma(\widehat{A} X W^{(1)})$$



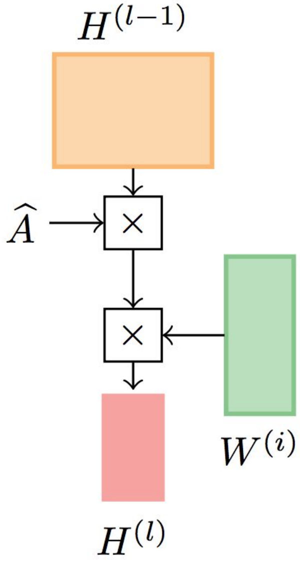
[1] Kipf & Welling, ICLR 2017

Shortcoming of Vanilla GCN

- 😀 fc is shared ⇒ inductive
- 😢 Appendix Experiments of [1] shows no gains beyond 2 layers
- 😢 cannot mix neighbors from various distances in arbitrary linear combinations e.g. **cannot learn Gabor Filters**!

<u>Vanilla GC Layer</u>

$$H^{(1)} = \sigma(\widehat{A}XW^{(1)})$$



[1] Kipf & Welling, ICLR 2017

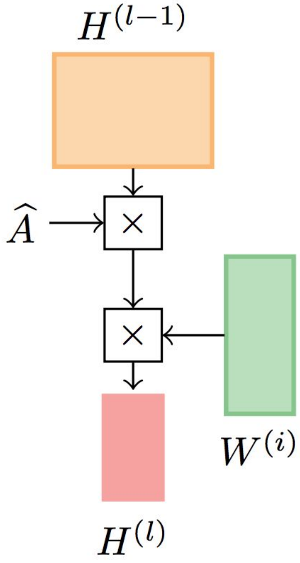
Shortcoming of Vanilla GCN

- 😀 fc is shared ⇒ inductive
- 😢 Appendix Experiments of [1] shows no gains beyond 2 layers
- 😢 cannot mix neighbors from various distances in arbitrary linear combinations e.g. **cannot learn <u>Gabor Filters</u>**!

?

<u>Vanilla GC Layer</u>

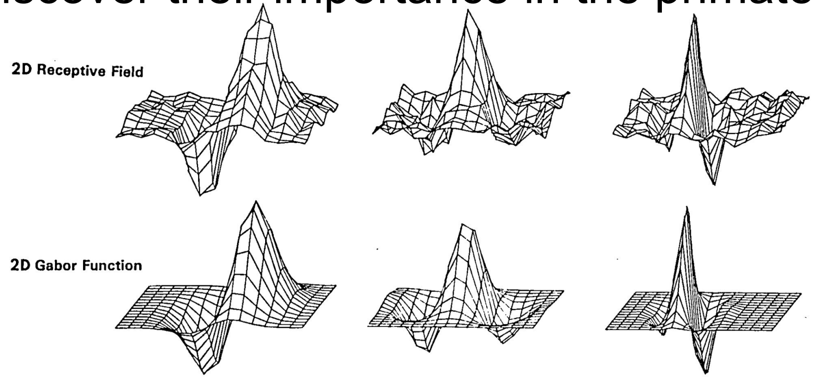
$$H^{(1)} = \sigma(\widehat{A}XW^{(1)})$$



[1] Kipf & Welling, ICLR 2017

Detour: Review Gabor Filters

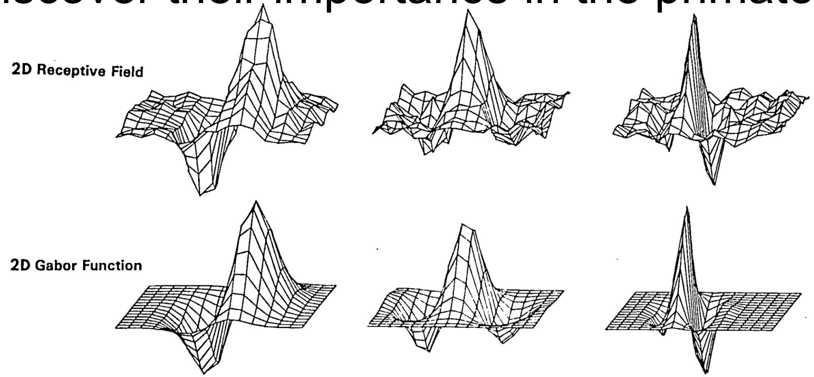
Neuroscientists discover their importance in the primate visual cortex [2, 3]:



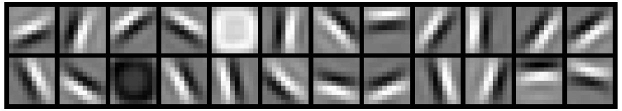
[2] Daugman, Vision Research,1980
[3] Daugman, Journal of the Optical Society of America, 1985
[4] Honglak Lee et al, ICML, 2009
[5] Alex Krizhevsky et al, NeurIPS 2012

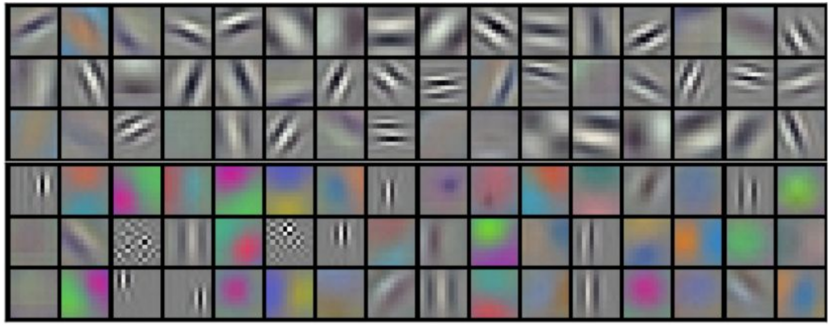
Detour: Review Gabor Filters

Neuroscientists discover their importance in the primate visual cortex [2, 3]:



Further, they are automatically recovered by training CNNs on images [4, 5]





[2] Daugman, Vision Research,1980
[3] Daugman, Journal of the Optical Society of America, 1985
[4] Honglak Lee et al, ICML, 2009
[5] Alex Krizhevsky et al, NeurIPS 2012

Main Motivation

Main Motivation

Extend the class of representations realizable by GCNs
e.g. to learn Gabor Filters

Agenda

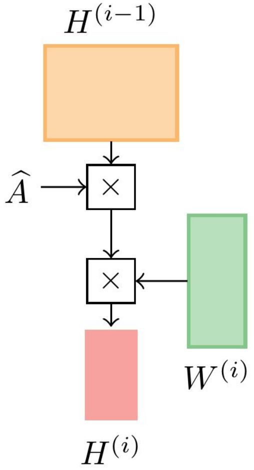
- Review Graph Convolutional Networks (GCN)
  - Application Semi-Supervised Node Classification (SSNC)
  - Shortcoming of GCN
- MixHop: Higher-Order GCN
  - Sparsification
- MixHop Results on SSNC

Our Model: MixHop

MixHop GC Layer

Vanilla GC Layer

$$H^{(1)} = \sigma(\widehat{A} X W^{(1)})$$



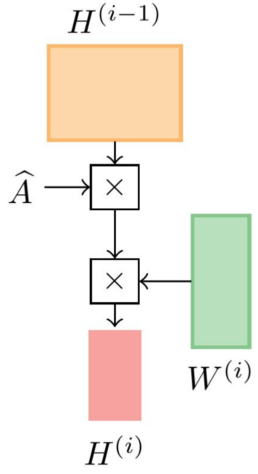
Our Model: MixHop

<u>MixHop GC Layer</u>

<u>Vanilla GC Layer</u>

$$H^{(1)} = \sigma(\widehat{A}XW^{(1)})$$

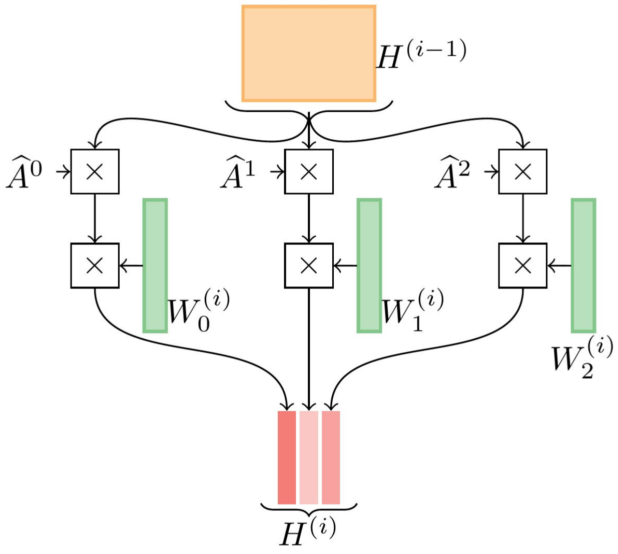
Couple of code lines
implements concatenation



Our Model: MixHop

MixHop GC Layer

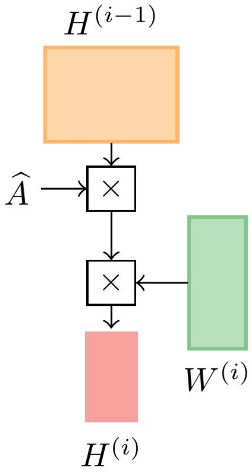
$$H^{(1)} = \Big\Vert_{j \in P} \sigma\left(\widehat{A}^j X W_j^{(1)}\right)$$



Couple of code lines implements concatenation

Vanilla GC Layer

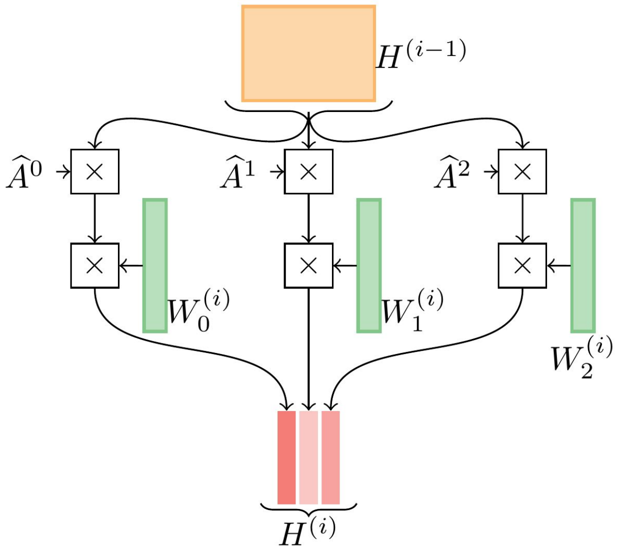
$$\boldsymbol{H^{(1)} = \sigma(\widehat{A} X W^{(1)})}$$



Our Model: MixHop

MixHop GC Layer

$$H^{(1)} = \Big\Vert_{j \in P} \sigma\left(\widehat{A}^j X W_j^{(1)}\right)$$

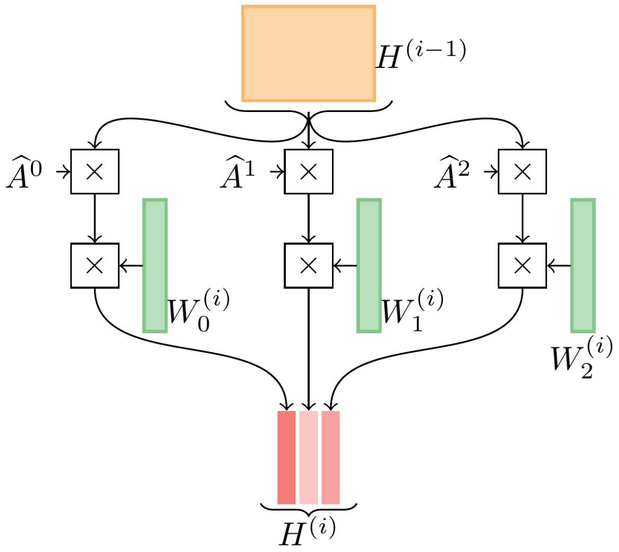


Our Model: MixHop

😀 Inductive

MixHop GC Layer

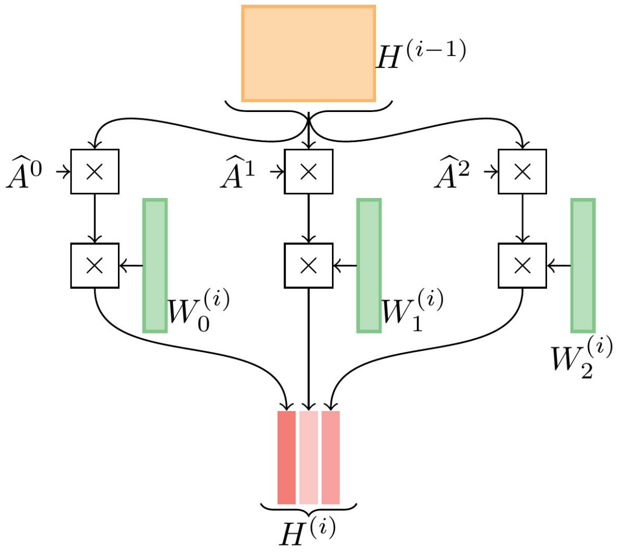
$$H^{(1)} = \Big\Vert_{j \in P} \sigma\left(\widehat{A}^j X W_j^{(1)}\right)$$



Our Model: MixHop

MixHop GC Layer

$$H^{(1)} = \Big\Vert_{j \in P} \sigma\left(\widehat{A}^j X W_j^{(1)}\right)$$



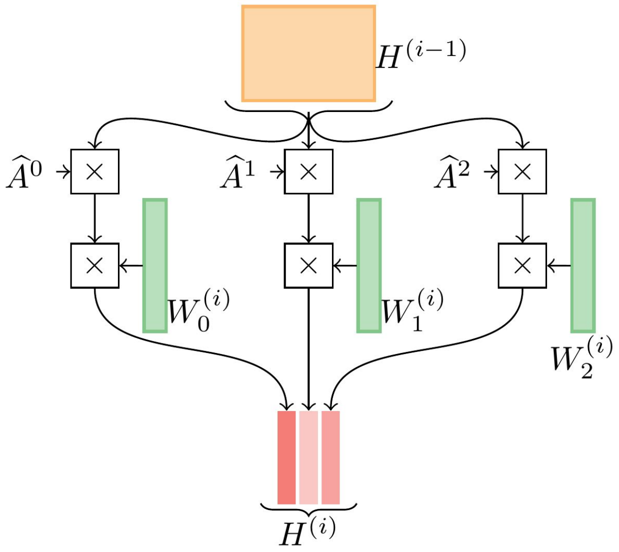
😀 Inductive

😀 Can incorporate distant nodes

Our Model: MixHop

<u>MixHop GC Layer</u>

$$H^{(1)} = \Big\Vert_{j \in P} \sigma\left(\widehat{A}^j X W_j^{(1)}\right)$$



😀 Inductive

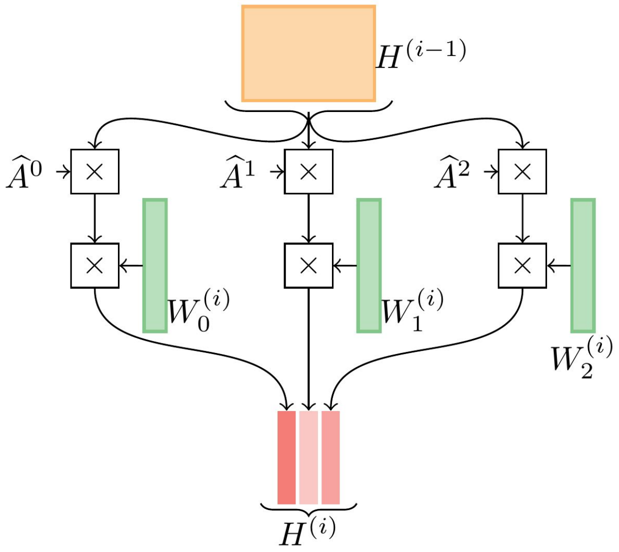
😀 Can incorporate distant nodes

😀 Can mix neighbors across distances in arbitrary linear combinations

Our Model: MixHop

<u>MixHop GC Layer</u>

$$H^{(1)} = \Big\Vert_{j \in P} \sigma\left(\widehat{A}^j X W_j^{(1)}\right)$$



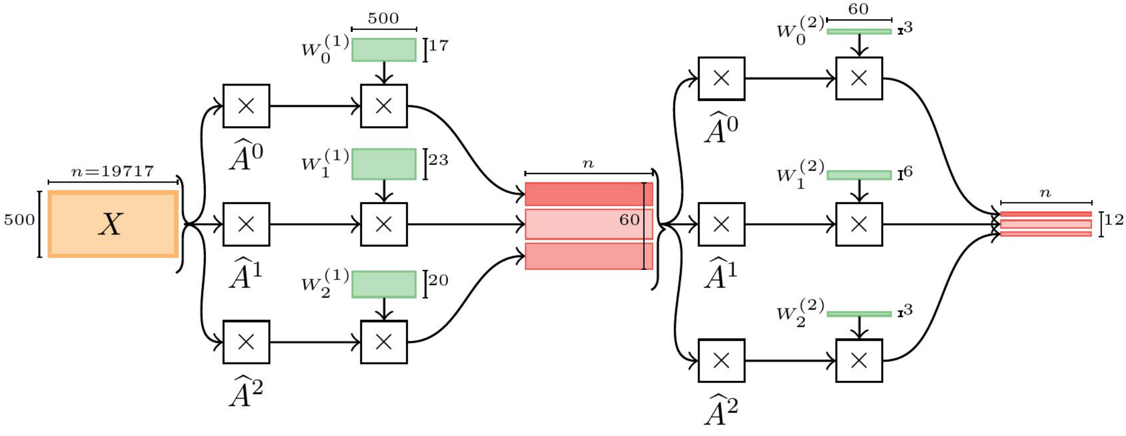
- 😀 Inductive
- 😀 Can incorporate distant nodes
- 😀 Can mix neighbors across distances in arbitrary linear combinations i.e. **can learn <u>Gabor Filters</u>**!

Sparsification

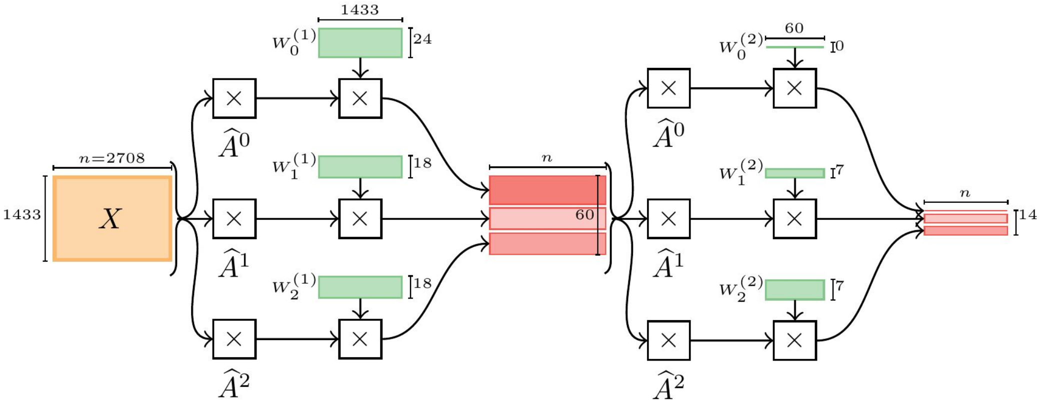
We add group L2-Lasso Regularization to drop-out columns feature matrices, similar to [6]

[images are rotated space]

[6] Gordon et al, CVPR, 2018



(a) Pubmed



(b) Cora

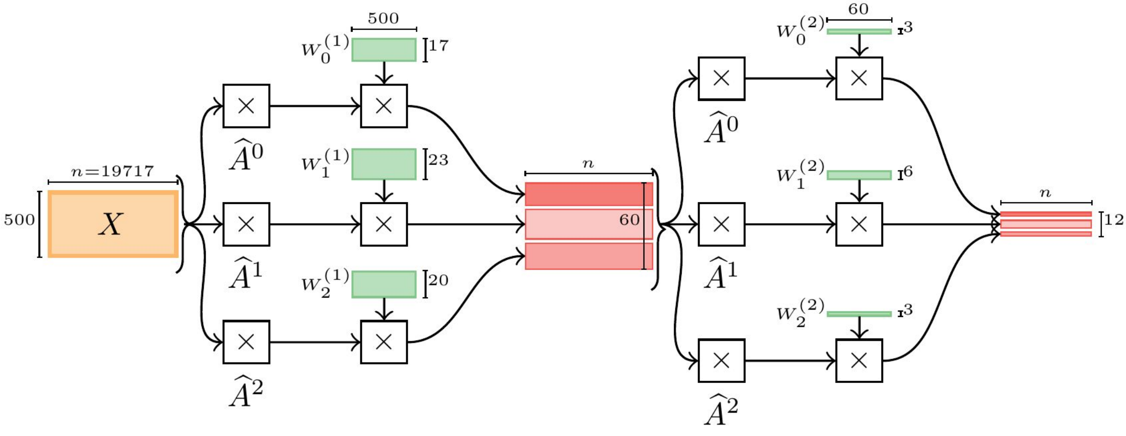
Sparsification

We add group L2-Lasso Regularization to drop-out columns feature matrices, similar to [6]

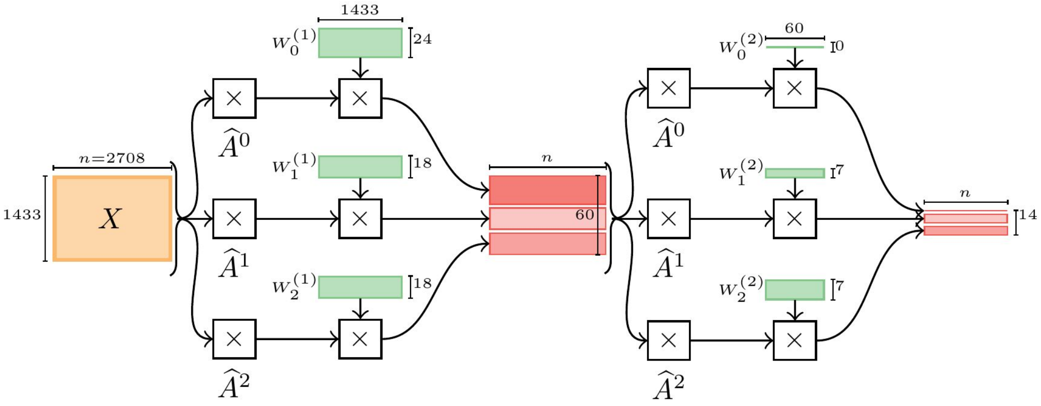
2nd layer of Cora drops-out zeroth-power completely.

[images are rotated space]

[6] Gordon et al, CVPR, 2018



(a) Pubmed



(b) Cora

Agenda

- Review Graph Convolutional Networks (GCN)
  - Application Semi-Supervised Node Classification (SSNC)
  - Shortcoming of GCN
- MixHop: Higher-Order GCN
  - Sparsification
- MixHop Results on SSNC

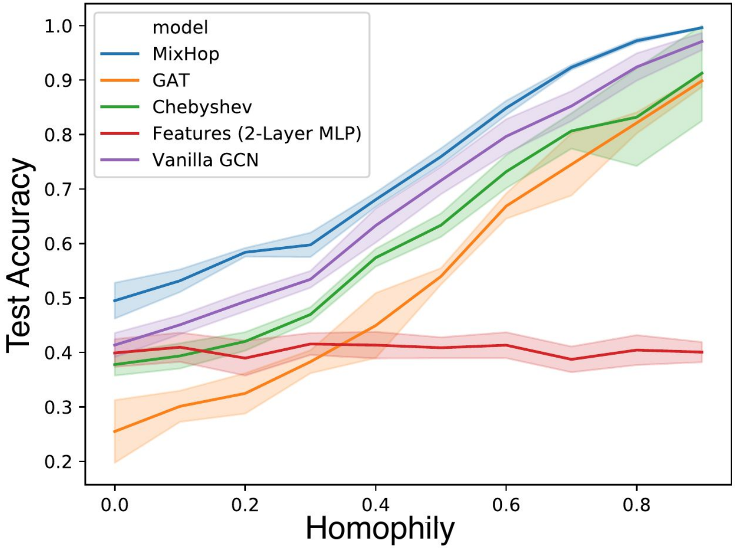
Results on Citation Datasets

Model	Citeseer	Cora	Pubmed
2-Layer MLP	70.6±1	69.0±1.1	78.3±0.54
Chebyshev (Defferrard et al., 2016)	74.2±0.5	85.5±0.4	81.8±0.5
Vanilla GCN (Kipf & Welling, 2017)	76.7±0.43	86.1±0.34	82.2±0.29
GAT (Velickovic et al., 2018)	74.8±0.42	83.0±1.1	81.8±0.18
MixHop: default architecture (ours)	76.3±0.41	87.0±0.51	83.6±0.68
MixHop: learned architecture (ours)	**77.0**±0.54	**87.2**±0.32	**83.8**±0.44

Table 3: Classification results on random partitions of (Yang et al., 2016) datasets.

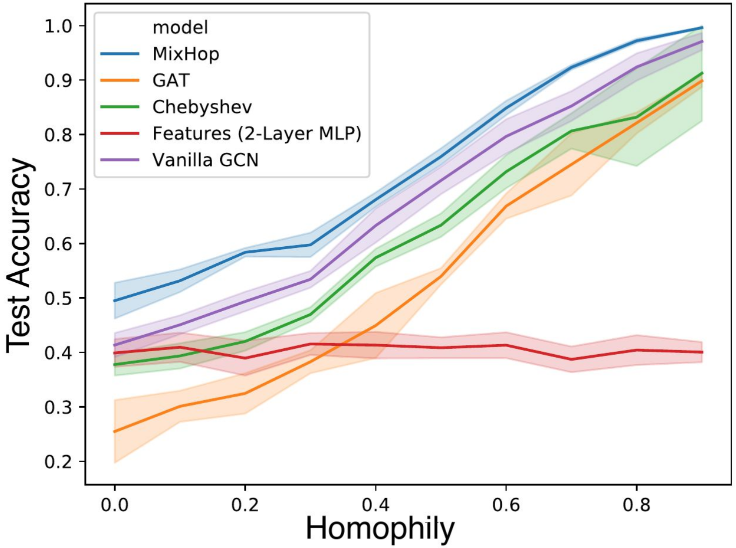
Results on (Synthetic) Homophily Datasets

With less homophily, our performance gap increases

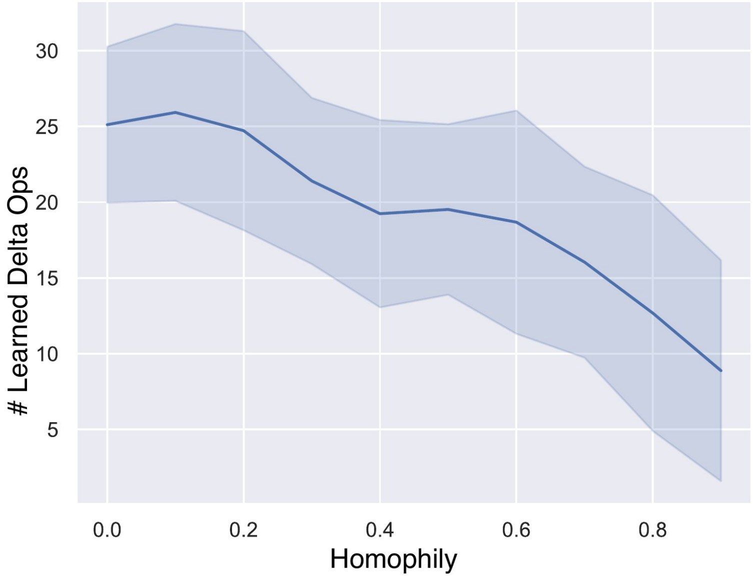


Results on (Synthetic) Homophily Datasets

With less homophily, our performance gap increases



With less homophily, our method learns more feature differences (i.e. Gabor-like Filters)



References

[1] Kipf & Welling, “Semi-Supervised Classification with Graph Convolutional Networks, ICLR”, 2017
[2] Daugman, “Two-dimensional spectral analysis of cortical receptive field profiles”, Vision Research,1980
[3] Daugman, “Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters”, Journal of the Optical Society of America, 1985
[4] Honglak Lee et al, “Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations”, ICML, 2009
[5] Alex Krizhevsky et al, “ImageNet Classification with Deep Convolutional Neural Networks”, NeurIPS 2012
[6] Gordon et al, “Morphnet: Fast & simple resource-constrained structure learning of deep networks” CVPR 2018

Conclusion

- With just a couple of lines, Kipf's model can be extended to incorporate powers of (normalized) adjacency matrix
- Allowing it to learn general neighborhood mixing, and its special cases: Gabor-like Filter and Delta Ops
- Inspection shows Delta Ops are indeed learned with lower levels of homophily.

Thank you for listening!

Poster #88

Slides at: http://sami.haija.org/icml19