AN INTRODUCTION TO DEEP LEARNING ON GRAPHS AND CURRENT DEVELOPMENTS

Pedro H.C. Avelar Luis C. Lamb

INSTITUTO DE INFORMÁTICA UFRGS

September 2019

1. Introduction to Graph-Based Learning

2. Graph-based Learning for Superpixel Images

3. Learning on Graphs for Continuous Domains

4. Side Projects
   - Measuring Academic Engagement with Ethics in the Area of AI
   - Second Language Acquisition Modelling with Attentional Models and Half-Life Regression

5. Not Covered Here

Introduction to Graph-Based Learning

[Sperduti and Starita, 1997] gave one of the first suggestions on using neural networks with graph-structure data

[Sperduti and Starita, 1997] gave one of the first suggestions on using neural networks with graph-structure data, but [Gori et al., 2005] constructed a model that is much akin to today's architectures

[Sperduti and Starita, 1997] gave one of the first suggestions on using neural networks with graph-structure data, but [Gori et al., 2005] constructed a model that is much akin to today's architecturesand in [Scarselli et al., 2009] they reaffirmed the model in its full potential.

Main Ideas:

Main Ideas:

1. Build the network of neural modules according to the input graph structure

Main Ideas:

1. Build the network of neural modules according to the input graph structure
2. Propagate information until it converges

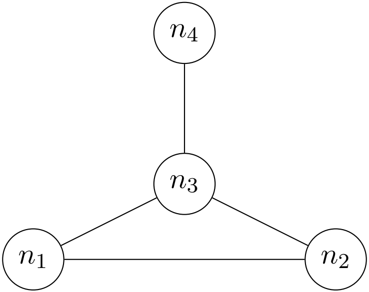
Main Ideas:

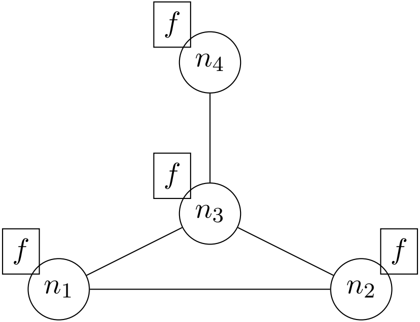
1. Build the network of neural modules according to the input graph structure
2. Propagate information until it converges
3. Readout desired information through another neural module

Main Ideas:

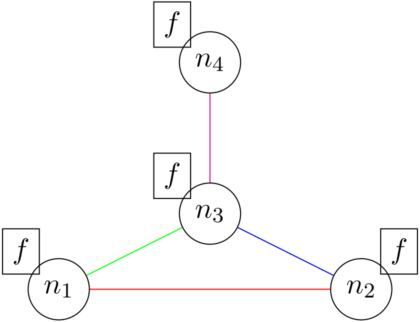
1. Build the network of neural modules according to the input graph structure
2. Propagate information until it converges
3. Readout desired information through another neural module
4. [Scarselli et al., 2009] One could adapt this to work with different “kinds” of nodes (i.e. the edge and graph nodes in [Battaglia et al., 2018])

THE GRAPH NEURAL NETWORK MODEL: A GRAPH

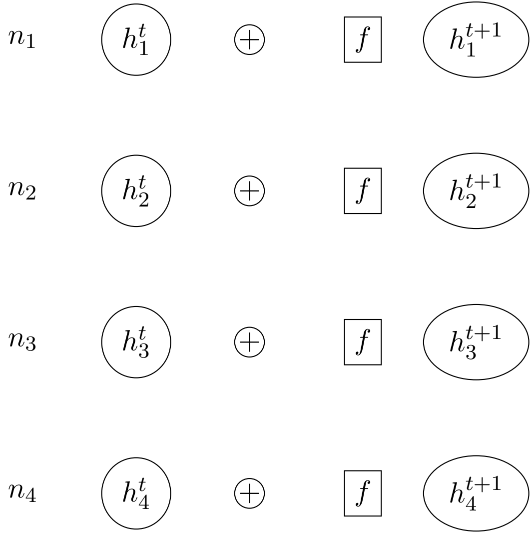



f
n4
f
n3
f
n1
f
n2

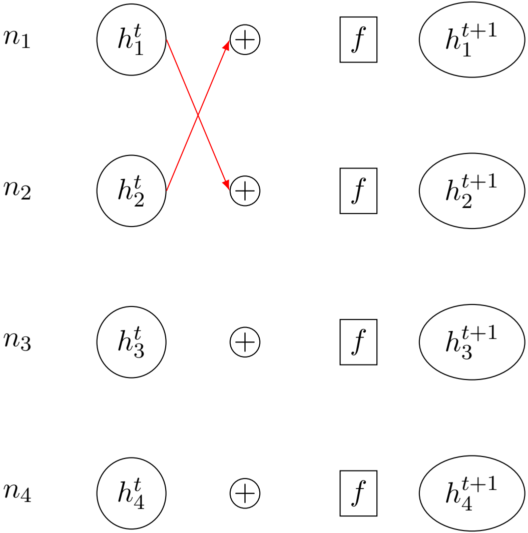
THE GRAPH NEURAL NETWORK MODEL: NEURAL MODULES CORRESPONDING TO EACH NODE



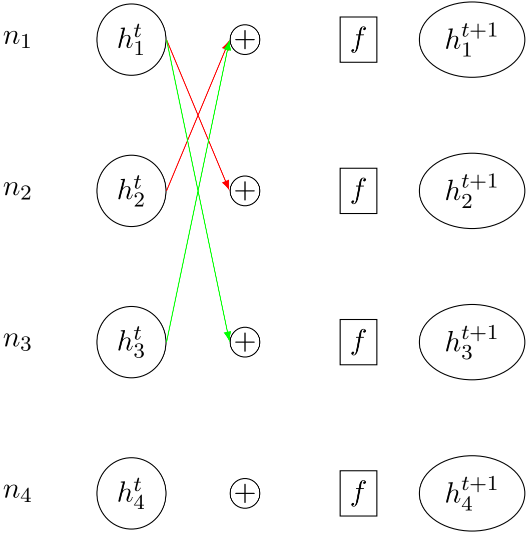
THE GRAPH NEURAL NETWORK MODEL: ONE PROPAGATION LAYER



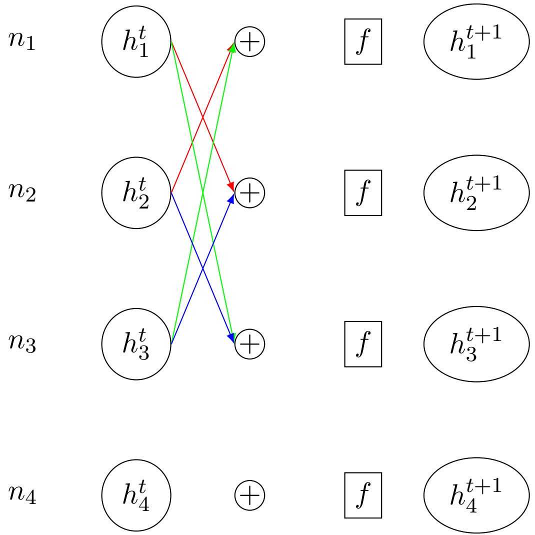
THE GRAPH NEURAL NETWORK MODEL: ONE PROPAGATION LAYER



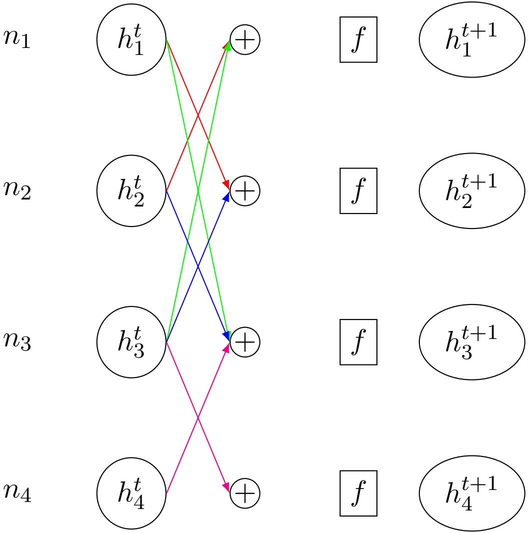
THE GRAPH NEURAL NETWORK MODEL: ONE PROPAGATION LAYER



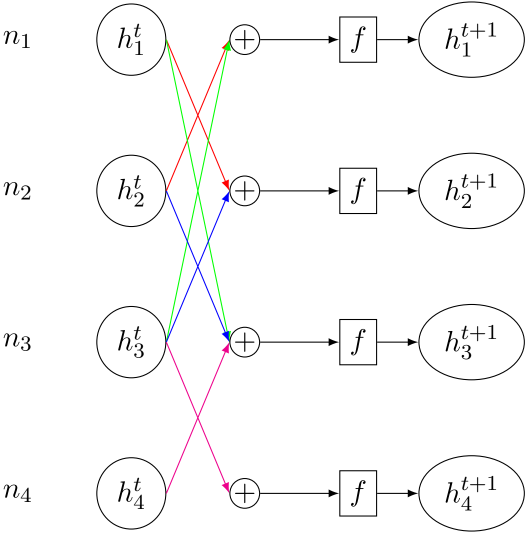
THE GRAPH NEURAL NETWORK MODEL: ONE PROPAGATION LAYER



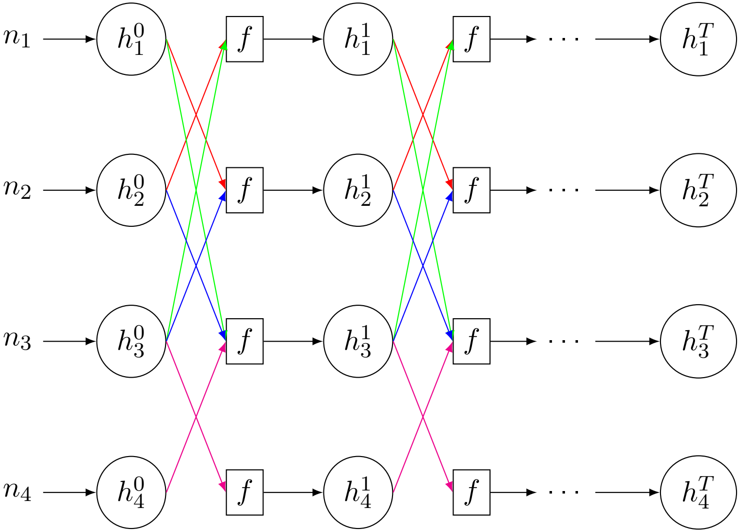
THE GRAPH NEURAL NETWORK MODEL: ONE PROPAGATION LAYER



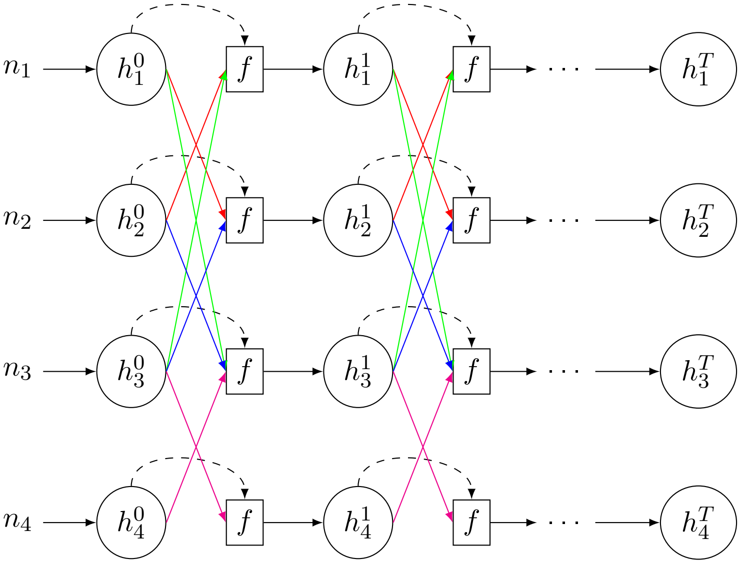
THE GRAPH NEURAL NETWORK MODEL: ONE PROPAGATION LAYER



THE GRAPH NEURAL NETWORK MODEL: UNROLLING IN TIME



THE GRAPH NEURAL NETWORK MODEL: RECURRENT CONNECTIONS



With sum-aggregation, it is easy to organise the propagation as matrix multiplication. Given an adjacency matrix $\mathbf{M}$ with shape (n, n), the learned function f, and the current node "state" $\mathbf{h}^t$ with shape (n, d), we have:

$$\mathbf{h}^{t+1} = f(\mathbf{M} \times \mathbf{h}^t)$$

Which translates, on a node-to-node basis as:

$$\mathbf{h}_n^{t+1} = f(\sum_{j \in \mathcal{N}(n)} \mathbf{h}_j^t)$$

With $\mathcal{N}(n)$ being the neighbourhood of node n.

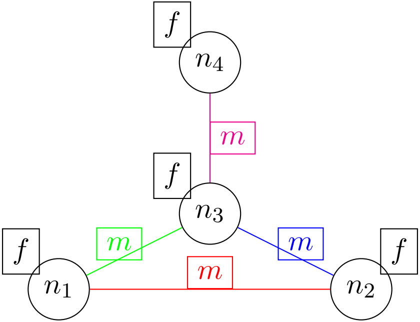
Same principles as [Gori et al., 2005, Scarselli et al., 2009]:

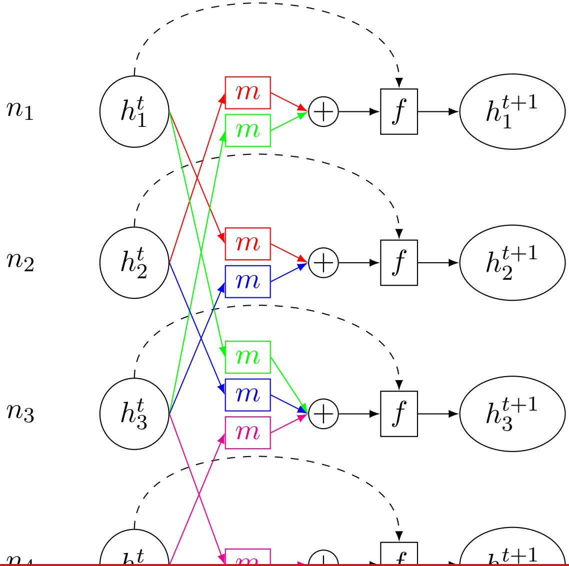
Same principles as [Gori et al., 2005, Scarselli et al., 2009]:

- A separate “messaging” function added between communicating nodes.

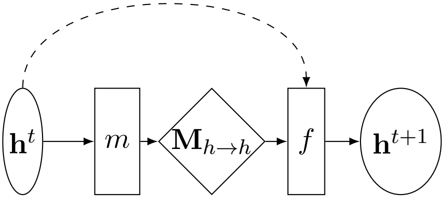
Same principles as [Gori et al., 2005, Scarselli et al., 2009]:

- A separate "messaging" function added between communicating nodes.
- "Messages" propagated for a given number of iterations instead


f
n_4
m
f
n_3
f
m
m
f
m
n_1
n_2


n_1
h_1^t
m
m
f
h_1^{t+1}
n_2
h_2^t
m
m
f
h_2^{t+1}
n_3
h_3^t
m
m
m
f
h_3^{t+1}

We'll use the following graphical representation for simplifying the MPNN blocks, ellipses are state tensors, rectangles are learned functions, diamonds represent matrix multiplication (or any other type of neighbourhood aggregation), circles (not present) are pointwise operations, and if two arrows join at a point their tensors are most likely concatenated for every tuple of node:



DETAILS LEFT OUT

- Messaging functions often take more than just the source's embedding.
- Neighbourhood aggregation can be made in many ways:
  - Sum-aggregation [Gori et al., 2005, Scarselli et al., 2009, Gilmer et al., 2017, Selsam et al., 2018, Xu et al., 2019]
  - Average-aggregation [Kipf and Welling, 2017]
  - Attentional sum-aggregation [Velickovic et al., 2018]
  - Product-aggregation
  - Or any other order-invariant function
- Readout functions can be really diverse, using only specific nodes, all nodes of a kind/type, pairs of nodes, etc...
- Batching of different input graphs (done as a block-diagonal matrix)

Independently, generalisations of convolutions to graph-domains, such as [Kipf and Welling, 2017], led to an architecture similar to [Scarselli et al., 2009], only with no recurrent connection, untied weights, and average-aggregation instead of sum-aggregation.

[Battaglia et al., 2018] suggests the generalisation of the concept of graph neural networks to models other than neural networks, builds upon this idea of inductive bias and argues for three specific kinds/types of nodes: Vertex, Edge and Graph-level nodes.

RECAP

We can:

- Assemble neural modules in the configuration of the input graphs and propagate information through neighbours.

We can:

- Assemble neural modules in the configuration of the input graphs and propagate information through neighbours.
- Work with different "kinds"/types of nodes by training different "update" and "message" functions, working on different sets of embeddings, and different adjacency matrices (we'll see examples soon).

RECAP

We can:

- Assemble neural modules in the configuration of the input graphs and propagate information through neighbours.
- Work with different "kinds"/types of nodes by training different "update" and "message" functions, working on different sets of embeddings, and different adjacency matrices (we'll see examples soon).
- Read out the embeddings at the last layer and use them to provide an answer to the problem at hand.

Graph-based Learning for Superpixel Images

MAIN IDEAS

- Images have superpixel representations

MAIN IDEAS

- Images have superpixel representations
- These representations can be seen as a graph of superpixels

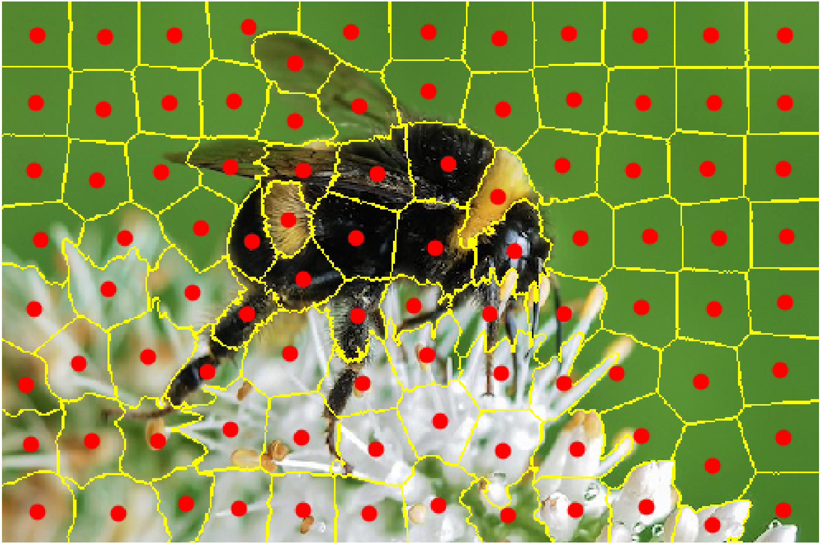
MAIN IDEAS

- Images have superpixel representations
- These representations can be seen as a graph of superpixels
- Use graph-based methods to work with this data

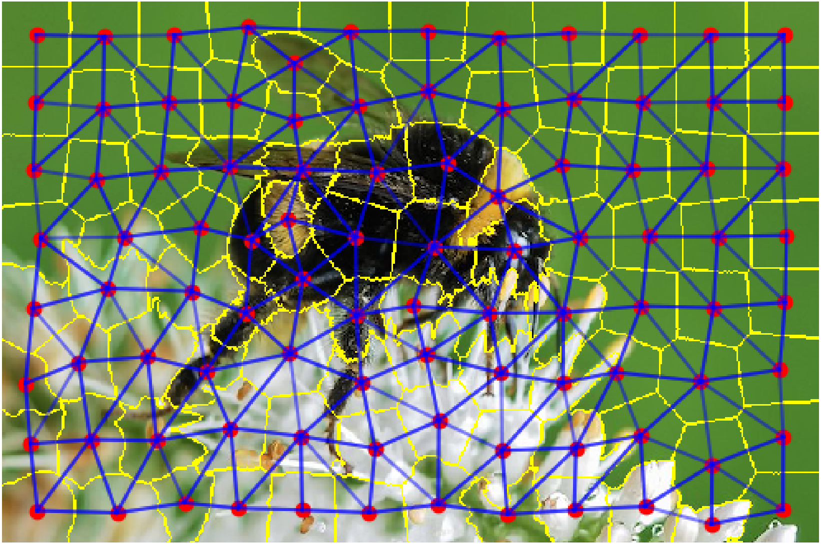
SUPERPIXEL GRAPH

SUPERPIXEL GRAPH

SUPERPIXEL GRAPH



SUPERPIXEL GRAPH



POSSIBLE FEATURES

- Color mean/std/Gram matrix (variance matrix)

POSSIBLE FEATURES

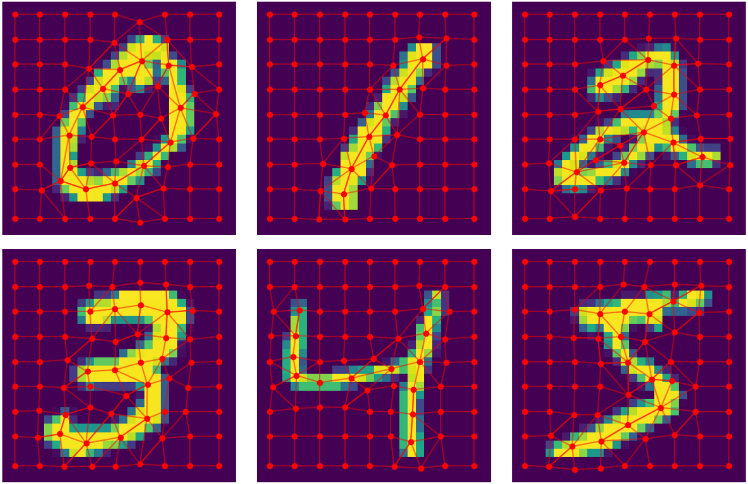
- Color mean/std/Gram matrix (variance matrix)
- Position mean/std/Gram matrix (variance matrix)

POSSIBLE FEATURES

- Color mean/std/Gram matrix (variance matrix)
- Position mean/std/Gram matrix (variance matrix)
- Superpixel Compactness

SANITY CHECKING – MNIST

SANITY CHECKING – MNIST



- No color information, so we use only:

- No color information, so we use only:
  - Average brightness

- No color information, so we use only:
  - Average brightness
  - Average (cartesian) position

CURRENT ARCHITECTURE

- No color information, so we use only:
  - Average brightness
  - Average (cartesian) position
- 3 1-headed GAT layers

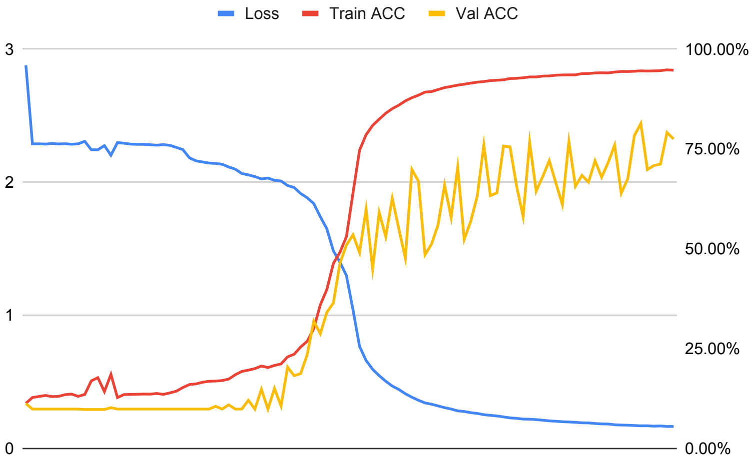
CURRENT ARCHITECTURE

- No color information, so we use only:
  - Average brightness
  - Average (cartesian) position
- 3 1-headed GAT layers
- Sum pooling

CURRENT ARCHITECTURE

- No color information, so we use only:
  - Average brightness
  - Average (cartesian) position
- 3 1-headed GAT layers
- Sum pooling
- An MLP with 2 layers

PRELIMINARY RESULTS



Profiling results give us out that performing online segmentation to transform to graphs is too slow:

Code	Time Relative to Batch
Gen Graphs from Images	55.91%
Batching Graphs	18.59%
To Pytorch/GPU	23.21%
GAT	1.03%
Metrics	0.05%
Backprop	1.21%

PLANNED IMPROVEMENTS/NEXT STEPS

- Generate graphs offline

PLANNED IMPROVEMENTS/NEXT STEPS

- Generate graphs offline
- More attention heads

PLANNED IMPROVEMENTS/NEXT STEPS

- Generate graphs offline
- More attention heads
- Denser graphs

PLANNED IMPROVEMENTS/NEXT STEPS

- Generate graphs offline
- More attention heads
- Denser graphs
- Data Aug: Segmentations with different number of pixels

PLANNED IMPROVEMENTS/NEXT STEPS

- Generate graphs offline
- More attention heads
- Denser graphs
- Data Aug: Segmentations with different number of pixels
- Other datasets: Pascal VOC, CoCo (segmentation), Transfer to Spherical Images

Suggestions?

Learning on Graphs for Continuous Domains

- Neural Networks can be put as residual blocks

MAIN IDEAS

- Neural Networks can be put as residual blocks
- Residual blocks can be interpreted as discrete derivatives

MAIN IDEAS

- Neural Networks can be put as residual blocks
- Residual blocks can be interpreted as discrete derivatives
- We can integrate residual blocks to give them continuity

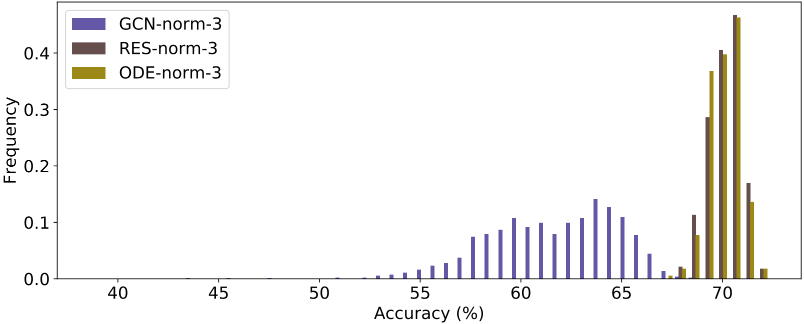
MAIN IDEAS

- Neural Networks can be put as residual blocks
- Residual blocks can be interpreted as discrete derivatives
- We can integrate residual blocks to give them continuity
- Why not do this with graphs?

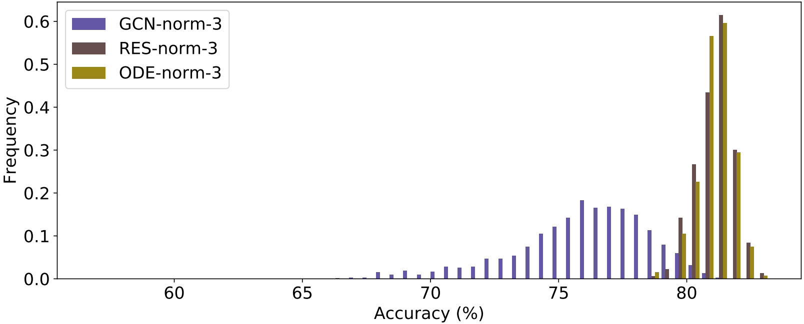
- First tests done with traditional GCN [Kipf and Welling, 2017]

- First tests done with traditional GCN [Kipf and Welling, 2017]
- Performance almost the same as discrete residual models

- First tests done with traditional GCN [Kipf and Welling, 2017]
- Performance almost the same as discrete residual models
- Tests performed on traditional node classification datasets:
  - Citeseer
  - Cora
  - Pubmed

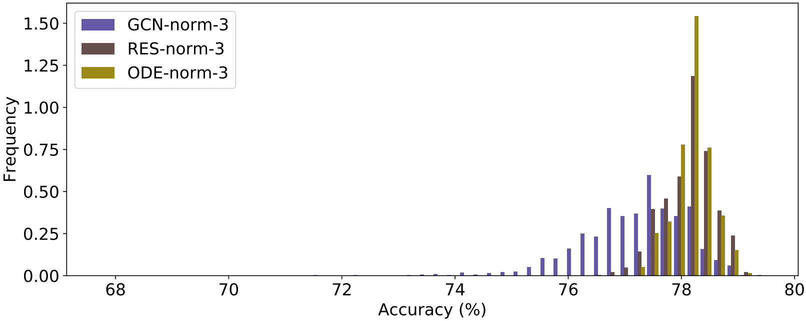

GCN-norm-3
RES-norm-3
ODE-norm-3
Frequency
0.0
0.1
0.2
0.3
0.4
40
45
50
55
60
65
70
Accuracy (%)

RESULTS



GCN-norm-3
RES-norm-3
ODE-norm-3
Frequency
0.0
0.1
0.2
0.3
0.4
0.5
0.6
60
65
70
75
80
Accuracy (%)

RESULTS


GCN-norm-3
RES-norm-3
ODE-norm-3
Frequency
1.50
1.25
1.00
0.75
0.50
0.25
0.00
68
70
72
74
76
78
80
Accuracy (%)

Working under way to extend this to physics models such as n-body orbits, ball collisions and string/spring physics

Suggestions?

Side Projects

MAIN IDEAS

- Computing ethics keywords from arxiv paper abstracts from Human-AI field

MAIN IDEAS

- Computing ethics keywords from arxiv paper abstracts from Human-AI field
- Checking growth in each category

MAIN IDEAS

- Computing ethics keywords from arxiv paper abstracts from Human-AI field
- Checking growth in each category
- Planning to have a rough draft by HAI

MAIN IDEAS

- Computing ethics keywords from arxiv paper abstracts from Human-AI field
- Checking growth in each category
- Planning to have a rough draft by HAI
- Suggestions?

MAIN IDEAS

- Using HLR and attention to predict errors learners may do

MAIN IDEAS

- Using HLR and attention to predict errors learners may do
- HLR will model forgetting

MAIN IDEAS

- Using HLR and attention to predict errors learners may do
- HLR will model forgetting
- Attention will model language structure

MAIN IDEAS

- Using HLR and attention to predict errors learners may do
- HLR will model forgetting
- Attention will model language structure
- Suggestions?

Not Covered Here

NOT COVERED HERE

- Differential Gradient (TBD – Planning to present it at Khipu)
- BRHIM (Waiting for data)

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V. F., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gülçehre, Ç., Song, F., Ballard, A. J., Gilmer, J., Dahl, G. E., Vaswani, A., Allen, K., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R. (2018).
Relational inductive biases, deep learning, and graph networks.
CoRR, abs/1806.01261.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017).
Neural message passing for quantum chemistry.
In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR.

Gori, M., Monfardini, G., and Scarselli, F. (2005).
A new model for learning in graph domains.

In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE.

Kipf, T. N. and Welling, M. (2017).
Semi-supervised classification with graph convolutional networks.
In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009).
The graph neural network model.
IEEE Transactions on Neural Networks, 20(1):61–80.

Selsam, D., Lamm, M., Bünz, B., Liang, P., de Moura, L., and Dill, D. L. (2018).
Learning a SAT solver from single-bit supervision.
CoRR, abs/1802.03685.

Sperduti, A. and Starita, A. (1997).

IEEE Trans. Neural Networks, 8(3):714–735.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018).
Graph attention networks.
In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019).
How powerful are graph neural networks?
In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, United States, May 6 - May 9, 2019, Conference Track Proceedings*. OpenReview.net.