

Energy Based Graph Neural Networks

John Shin & Prathamesh Dharangutte

NYU

May 11th, 2020

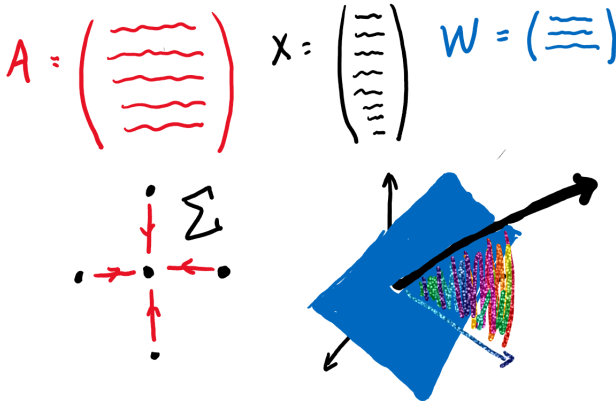
Brief Introduction to Graph NNs

- ▶ Graph Neural Networks are a type of NN that also takes graph data as input.
- ▶ A “vanilla GCN” ^[1] will take the adjacency matrix of a graph, A , feature vectors defined over the nodes, X , and perform message passing on the graph. One way to do this is summing over neighbors and applying a linear transformation W that reduces the dimensionality of the features, $A \cdot X \cdot W$.
- ▶ The propagation rule for GCN is:

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)}) \quad (1)$$

¹Thomas N Kipf and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).

A Simple Visual

Figure 1: $A \cdot X \cdot W$

Classifiers as EBM: Recent Work by W. Grathwohl et. al.

- ▶ Modern ML classifiers generally use a softmax function at the output

$$p_{\theta}(y|x) = \frac{\exp(f_{\theta(x)})[y]}{\sum_{y'} \exp(f_{\theta(x)})[y']} \quad (2)$$

- ▶ if we replace the denominator with the partition function, we get the joint distribution of x, y :

$$p_{\theta}(x, y) = \frac{\exp(f_{\theta(x)})[y]}{Z(\theta)} \quad (3)$$

- ▶ We can interpret this as an EBM with $E_{\theta}(x, y) = -f_{\theta(x)}[y]$ by marginalizing y .

$$p_{\theta}(x) = \sum_y p_{\theta}(x, y) = \frac{\exp(-E_{\theta}(x))}{Z(\theta)} \quad (4)$$

Goal: Energy Based GNNs

We want to create an energy-based GNN following the methodology of W. Grathwohl et. al.²

Issues

- ▶ The original work randomly generates data and minimizes the energy wrt this data using SGLD. A bit tricky with graph structure.
- ▶ We also want the model to be generative over the graph, in addition to the features.

²Will Grathwohl et al. "Your Classifier is Secretly an Energy Based Model and You Should Treat it Like One". In: *arXiv preprint arXiv:1912.03263* (2019).

Goal: Achieved

We want to create an energy-based GNN following the methodology of W. Grathwohl et. al.³

Solutions

- ▶ Sample nodes on graph and randomly generate features, propagate new features on graph and compute energy wrt new nodes only
- ▶ If two nodes are close in energy, add link.

Result: We have a generative model on both the features and the graph.

³Grathwohl et al., “Your Classifier is Secretly an Energy Based Model and You Should Treat it Like One”.

Some Experiments*

Model	CORA	Pubmed
GCN**	81.5	79.0
GCN-JEM	82.19	77.84
GCN-JEMO	83.34	78.17

*Results are average of 5 best performing models with minimal hyperparameter tuning.

**Results taken from paper (Kipf et. al.)⁴

⁴Kipf and Welling, "Semi-supervised classification with graph convolutional networks".

Outlook

Future Work:




The depth of GNNs are limited by the fact that repeated application of the same laplacian operator will result in the data being sent to the null space of that operator (Oono et. al.). A generative model over the graph may allow us to bypass this limitation, resulting in more expressive GNNs.

5

⁵Kenta Oono and Taiji Suzuki. “Graph Neural Networks Exponentially Lose Expressive Power for Node Classification”. In: *arXiv preprint cs.LG/1905.10947* (2019).

Thank You

Thank you for your attention!

-  Grathwohl, Will et al. “Your Classifier is Secretly an Energy Based Model and You Should Treat it Like One”. In: *arXiv preprint arXiv:1912.03263* (2019).
-  Kipf, Thomas N and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).
-  Oono, Kenta and Taiji Suzuki. “Graph Neural Networks Exponentially Lose Expressive Power for Node Classification”. In: *arXiv preprint cs.LG/1905.10947* (2019).