# Hierarchical Graph Representation Learning with Differentiable Pooling

Rex Ying, Jure Lescovec et. al

Presented by Eamon Collins

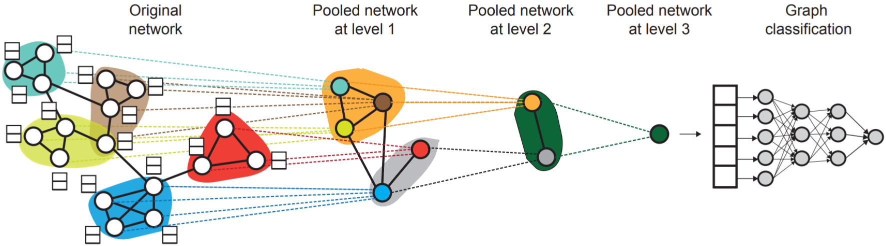
*ec3bd@virginia.edu*

https://qdata.github.io/deep2Read

# Overview

# Motivation

- Whole Graph Classification
  - Protein classification
  - Social networks
- Most current GNN graph classification methods are flat



# Prior Work

- Graph Classification through node embeddings
  - No representation of hierarchical structure
- Hierarchical structure recognized
  - Hierarchy not learned, used deterministic graph clustering algos

# Novel Idea

Learning the Hierarchical Structure of Graphs to Improve Representation

DiffPool enables the construction of deep, multi-layer GNN models by providing a differentiable module to hierarchically pool graph nodes for use with existing GNN techniques.

# Graph Neural Nets

## Message Passing

$$H^{(k)} = M(A, H^{(k-1)}; \Theta^{(k)})$$

M according to GCN:

$$M(A, H^{(k-1)}; \Theta^{(k)}) = ReLU(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(k-1)}W^{(k-1)})$$

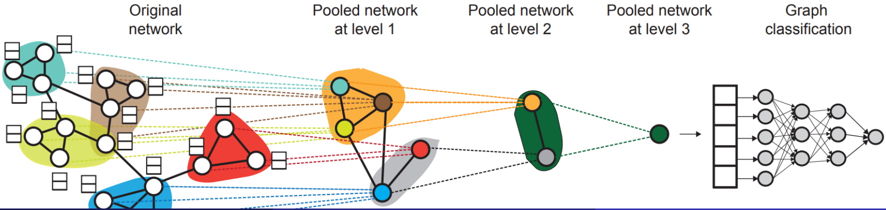
Where $H^k \in \mathbb{R}^{n \times d}$
$\tilde{A} = A + I$
$\tilde{D} = \sum_j \tilde{A}_{ij}$
$W^k \in \mathbb{R}^{d \times d}$ is a trainable weight matrix
For this paper, $Z = H^K = GNN(A, X)$
For some adjacency matrix $A$ and node features $X$

# Stacking GNNs

**Goal:** Define a general, end-to-end differentiable strategy that allows one to stack multiple GNN modules in a hierarchical fashion.

Formally, given $Z = GNN(A, X)$, the output of a GNN module, and a graph adjacency matrix $A \in \mathbb{R}^{n \times n}$,
DiffPool outputs a new coarsened graph containing $m < n$ nodes, with weighted adjacency matrix $A \in \mathbb{R}^{m \times m}$ and node embeddings $Z' \in \mathbb{R}^{m \times d}$

Can be repeated $L$ times to generate a model with L GNN layers that operate on a series of coarser and coarser versions of the input graph.



# Pooling With an Assignment Matrix

**Given $S^{(l)} \in \mathbb{R}^{n_l \times n_{l+1}}$**

$$X^{(l+1)} = S^{(l)\,T} Z^{(l)} \in \mathbb{R}^{n_{l+1} \times d}$$

$$A^{(l+1)} = S^{(l)\,T} A^{(l)} S^{(l)} \in \mathbb{R}^{n_{l+1} \times n_{l+1}}$$

Each row of $S^{(l)}$ corresponds to one of the $n_l$ nodes (or clusters) at layer $l$, and each column of $S^{(l)}$ corresponds to one of the $n_{l+1}$ clusters at the next layer $l+1$. Intuitively, $S^{(l)}$ provides a soft assignment of each node at layer $l$ to a cluster in the next coarsened layer $l+1$.

# Learning the Assignment Matrix

Learn $S^{(l)}$ and $Z^{(l)}$ with two separate GNNs applied over the same input

**$Z^{(l)}$**

$$Z^{(l)} = GNN_{l,embed}(A^{(l)}, X^{(l)})$$

$Z^{(l)}$ represents new embeddings for the input nodes at this layer.

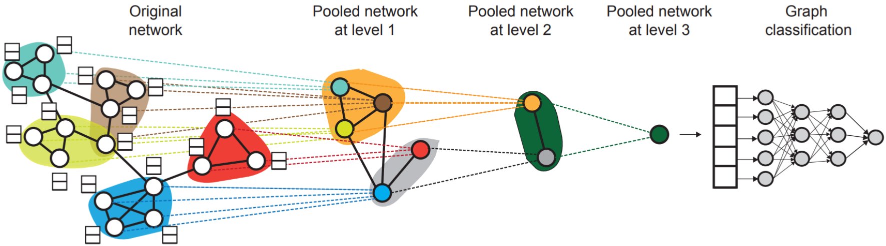
**$S^{(l)}$**

$$S^{(l)} = softmax(GNN_{l,pool}(A^{(l)}, X^{(l)}))$$

$S^{(l)}$ represents probabalistic assignments of the input nodes to $n_{l+1}$ clusters.
Output dimension $n_{l+1}$ is a hyperparameter

# Learning the Assignment Matrix

**Full graph representation:** At the penultimate layer $L-1$ of a deep GNN model using DIFFPOOL, the assignment matrix $S^{(L-1)}$ is set to be a vector of 1's, such that all nodes at the final layer $L$ are assigned to a single cluster, generating a final embedding vector corresponding to the entire graph.



# Goals

- Q1: How does DIFFPOOL compare to other pooling methods proposed for GNNs?
- Q2: How does DIFFPOOL combined with GNNs compare to the state-of-the-art for graph classification task, including both GNNs and kernel-based methods?
- Q3: Does DIFFPOOL compute meaningful and interpretable clusters on the input graphs?

# Experiments

**Datasets**

1. Proteins
2. Enzymes
3. D&D (another protein identification)
4. Reddit
5. Collab (Scientific collaboration set)

**Model Details**

1. GraphSage used as GNN model integrated with DiffPool
   - DiffPool layer after every 2 GraphSage layers, and only 2 DiffPool layers total
2. Every DiffPool layer sets the number of clusters to 25% of the incoming nodes.

# Results

| | Method | Data Set | | | | | |
|---|---|---|---|---|---|---|---|
| | | ENZYMES | D&D | REDDIT-MULTI-12K | COLLAB | PROTEINS | Gain |
| Kernel | GRAPHLET | 41.03 | 74.85 | 21.73 | 64.66 | 72.91 | |
| | SHORTEST-PATH | 42.32 | 78.86 | 36.93 | 59.10 | 76.43 | |
| | 1-WL | 53.43 | 74.02 | 39.03 | 78.61 | 73.76 | |
| | WL-OA | 60.13 | 79.04 | 44.38 | 80.74 | 75.26 | |
| GNN | PATCHYSAN | – | 76.27 | 41.32 | 72.60 | 75.00 | 4.17 |
| | GRAPHSAGE | 54.25 | 75.42 | 42.24 | 68.25 | 70.48 | – |
| | ECC | 53.50 | 74.10 | 41.73 | 67.79 | 72.65 | 0.11 |
| | SET2SET | 60.15 | 78.12 | 43.49 | 71.75 | 74.29 | 3.32 |
| | SORTPOOL | 57.12 | 79.37 | 41.82 | 73.76 | 75.54 | 3.39 |
| | DIFFPOOL-DET | 58.33 | 75.47 | 46.18 | **82.13** | 75.62 | 5.42 |
| | DIFFPOOL-NOLP | 61.95 | 79.98 | 46.65 | 75.58 | 76.22 | 5.95 |
| | DIFFPOOL | **62.53** | **80.64** | **47.08** | 75.48 | **76.25** | **6.27** |

# Conclusions

- Allowing the GNN to learn embeddings with hierarchical information can greatly improve results on graph classification tasks
- Can be easily used to augment existing "flat" GNN techniques
- Invariant under node permutations as long as the component GNN is

| | Method | Data Set | | | | | |
|---|---|---|---|---|---|---|---|
| | | ENZYMES | D&D | REDDIT-MULTI-12K | COLLAB | PROTEINS | Gain |
| Kernel | GRAPHLET | 41.03 | 74.85 | 21.73 | 64.66 | 72.91 | |
| | SHORTEST-PATH | 42.32 | 78.86 | 36.93 | 59.10 | 76.43 | |
| | 1-WL | 53.43 | 74.02 | 39.03 | 78.61 | 73.76 | |
| | WL-OA | 60.13 | 79.04 | 44.38 | 80.74 | 75.26 | |
| GNN | PATCHYSAN | – | 76.27 | 41.32 | 72.60 | 75.00 | 4.17 |
| | GRAPHSAGE | 54.25 | 75.42 | 42.24 | 68.25 | 70.48 | – |
| | ECC | 53.50 | 74.10 | 41.73 | 67.79 | 72.65 | 0.11 |
| | SET2SET | 60.15 | 78.12 | 43.49 | 71.75 | 74.29 | 3.32 |
| | SORTPOOL | 57.12 | 79.37 | 41.82 | 73.76 | 75.54 | 3.39 |
| | DIFFPOOL-DET | 58.33 | 75.47 | 46.18 | **82.13** | 75.62 | 5.42 |
| | DIFFPOOL-NOLP | 61.95 | 79.98 | 46.65 | 75.58 | 76.22 | 5.95 |
| | DIFFPOOL | **62.53** | **80.64** | **47.08** | 75.48 | **76.25** | **6.27** |

# References


. N. Kipf and M. Welling.
Semi-supervised classification with graph convolutional networks.
In International Conference on Learning Representations, 2017.

. Li, D. Tarlow, M. Brockschmidt, and R. Zemel.
Gated graph sequence neural networks.
International Conference on Learning Representations, 2016.

# The End