Weekly Report 2019.07.15-2019.07.21

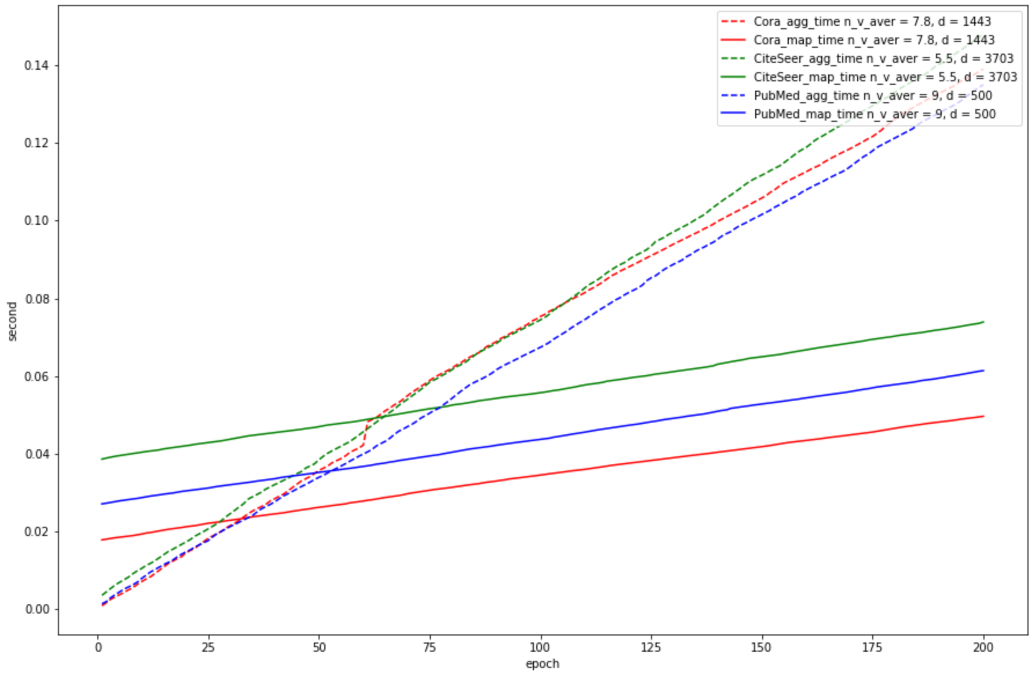
> Jingtun ZHANG

Work and Progress

1. Reimplementation of [HAG](): O(V^3) -> unreasonable

2. Profiling of MPNN model in pyg:

   get data on small graph but not reasonable, maybe wrong timer



3. Overview of GNN and MPNN Model

GNN(Graph Neural Network)

> from: Graph Neural Networks: A Review of Methods and Applications

Why GNN from CNN

1. graphs are the most typical locally connected structure
2. share weights reduce the computational cost compared with traditional spectral graph theory, approximation
3. multilayer structure is the key to deal with hierarchical patterns, which captures the featuofres of various sizes
4. CNNs or RNNs need a specific order, but there is no natural order of nodes in graph, GNNs output is input order invarient

5. human intelligencce is most based on the graph, GNNs can do information propagation guided by the graph structure
6. GNNs explores to generate the graph from non-structural data

Drawback of traditional algorithm

1. no parameter-share cause computational inefficiency
2. lack of generalization

Origin GNN

target: learn a state embedding $\mathbf{h}_v \in \mathbb{R}^s$ for each node

procedure:

$\mathbf{h}_v = f(\mathbf{x}_v, edge\_attr_v, \mathbf{h}_u, \mathbf{x}_u)$

u means neighbors of v, f is local parameterized transition function

$\mathbf{o}_v = g(\mathbf{h}_v, \mathbf{x}_v)$

g is local output function

loss:

$loss = \sum_{i=1}^{p} (target_i - output_i)$

variants:

1. Directed Graphs
2. Heterogeneous Graphs
3. Graphs with edge information
4. Dynamics Graphs
5. Convolution
    1. spectral network
    2. non-spectral metworks
        1. Neural FPs
        2. DCNN (Diffusion-convolutional neural network)
        3. DGCN (Dual graph convolutional network)
        4. PATCHY-SAN
        5. LGCN
        6. MoNet
        7. GraphSAGE
        8. Gate
            1. Child-sum Tree-LSTM
            2. N-ary Tree-LSTM

            3. Sentence LSTM

        9. GAT (Graph Attention Network)

        10. Ship Connection

        11. Hierarchical Pooling

General Framework

1. Message Passing NN
2. Non-Local NN
3. GN (Graph networks)

Applications

1. Structural Scenarios

    1. Physics: CommNet

    2. Chemistry and Biology

        1. Molecular Fingerprints
        2. Protein Interface Prediction

    3. Knowledge Graph

2. Non-structural Scenarios

    1. Image Classification
    2. Visual Reasoning
    3. Semantic Segmentation
    4. Machine translation
    5. Sequence labeling
    6. Text Classification
    7. Relation & Event extraction

3. Others

    1. Generative Model
    2. Combinational Optimization

Open Problems

1. Hard to build Deep GNN: over-smoothing
2. Dynamic Graphs
3. Non-structral Scenarios
4. Scalablity

Datasets

Websites:

1. tu-dortmund
2. networkrepository

group by raw data size:

1. small graph ($\leq 300MB$)

2. big graph (500MB~5GB)
3. huge graph ($\geq 5GB$)

Message Passing Model

Message passing phase (T times propagation step)

M_t: message function

U_t: vertex update function

$\mathbf{m}_v^{t+1} = \sum_{w \in \mathcal{N}_v} M_t(\mathbf{h}_v^t, \mathbf{h}_w^t, \mathbf{e}_{vw})$

$\mathbf{h}_v^{t+1} = U_t(\mathbf{h}_v^t, \mathbf{m}_v^{t+1})$

Readout phase

R: readout function

$\hat{\mathbf{y}} = R(\mathbf{h}_v^T | v \in G)$

This week plan

1. understand MPNN model and change PyG source code to get insight profiling of MPNN model
2. Implementing HAG to reasonable time complexity
