Graph Attention Networks

Summary Notes by Max Guo

July 5, 2022

(?) - denotes a lack of familiarity or understanding of a particular concept

1 Information

- **Year**: 2018
- **Conference**: ICLR
- **Authors**:

Name	Institute
Petar Velickovic	University of Cambridge
Guillem Cucurull	University of Barcelona (UAB)
Arantxa Casanova	UAB
Adriana Romero	Montreal Institute
Pietro Lio	University of Cambridge
Yoshua Bengio	Montreal Institute

2 Research Problem

How can we extend Neural Network architectures to arbitrary graphs?

3 Existing Approaches and Shortcomings

- **Graph Neural Networks** (2005, 2009)
  - Generalize RNNs for general graphs.
  - Iteratively propagate node states until equilibrium, then apply NN to node states to get output per node.
- How can we *generalize convolutions* for graphs?
  - **Spectral Approaches**
    - Deal with spectral representation of a graph (graph Laplacian)
    - **Problem**: Trained models are not generalizable! Learned filters depend on the graph structure through the Laplacian eigenbasis (?)
  - **Non-spectral Approaches**
    - Define convolutions on the graph, directly
    - **Challenges**: How can you define an operator with weight-sharing (like CNNs) that works with different sized neighborhoods?

    - Noteworthy approach: GraphSAGE (Hamilton et al, 2017). (?)
        - For each node, sample a fixed-size neighborhood and feed into an aggregator. Works well!

- **Attention Mechanisms** (2015, 2016)
  - A de facto standard in sequence tasks
  - **Benefits**: Deal with variable-sized inputs
  - *Self-attention* - Attention mechanism computes a representation of a single sequence
    - (Vaswani et al, 2017) - self-attention is *sufficient* for constructing a powerful model for state-of-the-art machine translation task performance.

4 High Level Contributions

The authors perform node-classification on graph data using an attention mechanism.

5 Technical Contributions

5.1 GAT Architecture

- **Graph Attention Layer**
  - Takes node features and returns node features

$$\mathbf{h} = \{\vec{h}_1, \ldots, \vec{h}_N\} \longrightarrow \mathbf{h}' = \{\vec{h}'_1, \ldots, \vec{h}'_N\}$$

$$\vec{h}_i \in \mathbb{R}^F, \vec{h}'_i \in \mathbb{R}^{F'}$$

  - Attention coefficients (importance of node j's features to node i):

$$e_{ij} = a(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j) \tag{1}$$

Weight matrix $\mathbf{W} \in \mathbb{R}^{F' \times F}$

attention mechanism a is single-layer feedforward LeakyReLU NN

  - Normalization of attention coefficients, masking attention to be within a neighborhood:

$$\alpha_{ij} = \text{softmax}_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(e_{ik})} \tag{2}$$

  - Obtain the final node features:

$$\vec{h}'_i = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W}\vec{h}_j\right) \tag{3}$$

  - Multi-headed attention (concatenate results from K independent attention mechanisms a)

$$\vec{h}'_i = \Big|\Big|_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right) \tag{4}$$

  - Final layer employs averaging

- **Advantages over Related Works**
  - Highly parallelizable across edges and nodes. Complexity is on par with GCNs

  - Difference in importances assigned to nodes of a same neighborhood (differs from GCNs)
  - Applicable to inductive learning (supervised learning)
  - (Hamilton et al, 2017) samples fixed-size neighborhoods and requires consistent node-ordering in neighborhoods, whereas GATs do not assume any ordering.
  - No assumption on node's structural properties

6 Experimentation

- **Datasets**: CORA, Citeseer, Pubmed (all citation networks), protein-protein interaction (PPI)
- **Transductive learning** (citation networks) - label remainder of training dataset, vs **inductive learning** (PPI)
- (Skipping most experimental details...)
- **Metrics**: Accuracy for citation networks, micro-averaged F1 score for PPI
- **Results**: Improves upon GCNs by 1.5% on CORA, 1.6% on Citeseer, and 20.5% on GraghSAGE
- Perform a t-SNE visualization on the feature representations, showing clustering of the different node classes.

7 Further Work

- Practical problems (e.g. larger batch sizes)
- Model interpretability in attention mechanisms
- Graph classification, not just node classification
- Incorporate edge features

8 My Questions and Thoughts

- How robust is the GAT architecture to different hyperparameters?
- What happens when you utilize more complicated attention mechanisms, or more complicated "single-layers"?
- Does this generalize to hyper-graphs?

9 Appendix

(Things I learned in the process of searching things up while reading this paper)

- **Difference between semi-supervised and transductive learning**:
  - Semi-supervised learning has the goal of generalizing based on a dataset with some labels given
  - Transductive learning has the goal of labeling the remainder of the dataset with some labels given
- **Micro-averaged F1 Score** (source: towardsdatascience):

– Recall the following definitions:

$$\begin{aligned} \text{Precision} &= \frac{TP}{TP + FP} \\ \text{Recall} &= \frac{TP}{TP + FN} \\ \text{F1 Score} &= \frac{2(\text{Precision} * \text{Recall})}{\text{Precision} + \text{Recall}} \end{aligned}$$

– Multi-class approach: use one-vs-all approach. Different averaging techniques:

  * Macro-average: unweighted average
  * Weighted average: weight according to proportion of examples for each class
  * Micro-average: Same across micro-F1, micro-precision, micro-recall, and accuracy. Simply the proportion of observations classified correctly.