Gated Graph Sequence Neural Networks

School of Industrial and Management Engineering, Korea University

Hansam Cho

KOREA UNIVERSITY

DMQA hcai Human-Centered Artificial Intelligence Center

Contents

Research Purpose

❖ Gated Graph Sequence Neural Networks (2016, ICLR)

- 토론토 대학교에서 연구하였고 2022년 02월 20일 기준으로 2132회 인용

GATED GRAPH SEQUENCE NEURAL NETWORKS

Yujia Li* **& Richard Zemel**
Department of Computer Science, University of Toronto
Toronto, Canada
{yujiali,zemel}@cs.toronto.edu

Marc Brockschmidt & Daniel Tarlow
Microsoft Research
Cambridge, UK
{mabrocks,dtarlow}@microsoft.com

ABSTRACT

Graph-structured data appears frequently in domains including chemistry, natural language semantics, social networks, and knowledge bases. In this work, we study feature learning techniques for graph-structured inputs. Our starting point is previous work on Graph Neural Networks (Scarselli et al., 2009), which we modify to use gated recurrent units and modern optimization techniques and then extend to output sequences. The result is a flexible and broadly useful class of neural network models that has favorable inductive biases relative to purely sequence-based models (e.g., LSTMs) when the problem is graph-structured. We demonstrate the capabilities on some simple AI (bAbI) and graph algorithm learning tasks. We then show it achieves state-of-the-art performance on a problem from program verification, in which subgraphs need to be described as abstract data structures.

Research Purpose

❖ Gated Graph Sequence Neural Networks (2016, ICLR)

- Graph 구조 데이터를 위한 feature learning을 위한 방법론 제안 (기존GNN 개선)
- Graph Neural Networks를 sequential output에도 활용할 수 있도록 개선
- 해당 논문에서는 directed graph에 집중해서 설명, 하지만 undirected graph로 쉽게 확장 가능

Graph Neural Networks (Scarselli et al., 2009)

❖ Propagation Model

- 노드 사이의 정보 교환을 통한 node representation 생성 과정
- 아래 식 수렴할 때까지 학습 진행

$$\mathbf{h}_v^{(t)} = f^*(l_v, l_{\mathrm{CO}(v)}, l_{\mathrm{NBR}(v)}, \mathbf{h}_{\mathrm{NBR}(v)}^{(t-1)})$$

$h_v^{(t)}$: t시점 노드v representation
f^*: neural network
l_v: 노드v 레이블
$l_{CO(v)}$: 노드v에 연결되어 있는 edge 레이블
$l_{NBR(v)}$: 노드v의 인접 노드 레이블
$h_{NBR(v)}^{(t-1)}$: t-1시점 노드v 인접 노드 representation

Graph Neural Networks (Scarselli et al., 2009)

❖ Output Model and Learning

- 마지막 T시점의 node representation 기반으로 출력값 계산 (node 단위)
- Graph 단위의 문제 수행 시 "super node"를 생성
- Super node는 모든 node와 연결된 가상의 node로 super node의 representation이 전체 graph의 representation이라고 가정
- 학습은 Almeida-Pineda 알고리즘을 통해 학습 (RNN 학습 방법 중 하나)
- Almeida-Pineda 알고리즘은 파라미터의 특별한 제약이 있어야 수렴성이 보장된다는 단점 존재

$$o_v = \mathrm{g}\left(h_v^{(T)}, l_v\right)$$

o_v: node v의 출력
g: neural network
$h_v^{(T)}$: 마지막 T시점 node v의 representation
l_v: 노드v 레이블

Gated Graph Neural Networks (GG-NNs)

❖ Gated Graph Neural Networks (GG-NNs)

- GNN을 개선한 모델로 non-sequential output에 활용 가능
- GRU의 gate 시스템을 활용
- 학습 알고리즘을 BPTT(Backpropagation Through Time)로 바꿈
- Node annotation 활용
  - ✓ 주어진 graph에서 node s와 t가 연결 여부 판단하는 task 예시
  - ✓ 시작점 s를 [1, 0], 마지막 지점 t를 [0, 1], 나머지 노드를 [0, 0]으로 지정
  - ✓ Task에 따라서 다르게 정의될 수 있음

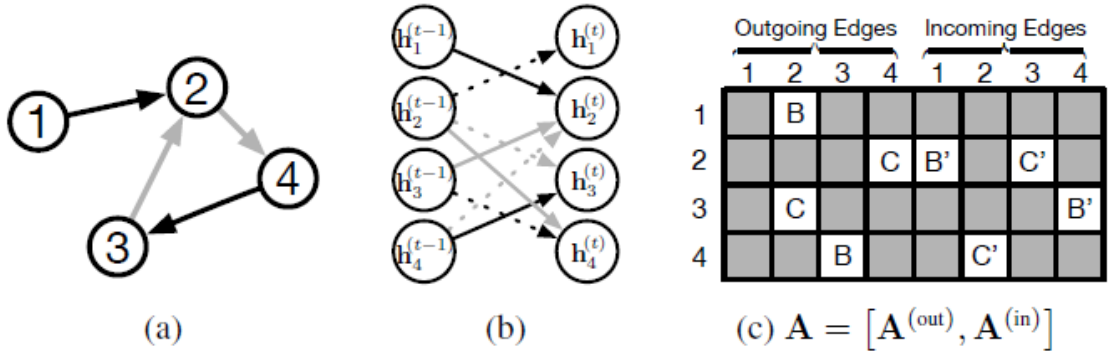
Gated Graph Neural Networks (GG-NNs)

❖ Propagation Model

- (1) : node initialize
- (2) : t-1 시점 → t 시점, node 사이의 정보 교환 식
  - (a) : 그래프 예시, edge 색에 따라 edge type 구분
  - (b) : t-1시점에서 t시점으로 넘어가는 정보교환 예시, 점선은 edge의 역방향
  - (c) : (2)번 식에서 사용되는 A matrix 예시
    - 연결된 edge만 학습가능한 parameter, 나머지는 0
    - 역방향 edge는 {edge}'으로 표현

$$\mathbf{h}_v^{(1)} = [\boldsymbol{x}_v^\top, \mathbf{0}]^\top \quad (1)$$

$$\mathbf{a}_v^{(t)} = \mathbf{A}_{v:}^\top \left[\mathbf{h}_1^{(t-1)\top} \ldots \mathbf{h}_{|\mathcal{V}|}^{(t-1)\top}\right]^\top + \mathbf{b} \quad (2)$$



(a) (b) (c) $\mathbf{A} = [\mathbf{A}^{(\text{out})}, \mathbf{A}^{(\text{in})}]$

Gated Graph Neural Networks (GG-NNs)

❖ Propagation Model

(1) : node initialize

(2) : t-1 시점 → t 시점, node 사이의 정보 교환 식

(3) : GRU의 update gate

(4) : GRU의 reset gate

(5) : reset gate를 통해 과거 정보를 현재 시점에 반영할 정도 선택

(6) : update gate를 통해 현재 시점의 정보 업데이트

$$\mathbf{h}_v^{(1)} = [\boldsymbol{x}_v^\top, \mathbf{0}]^\top \quad (1)$$

$$\mathbf{a}_v^{(t)} = \mathbf{A}_{v:}^\top \left[\mathbf{h}_1^{(t-1)\top} \dots \mathbf{h}_{|\mathcal{V}|}^{(t-1)\top}\right]^\top + \mathbf{b} \quad (2)$$

$$\mathbf{z}_v^t = \sigma\left(\mathbf{W}^z \mathbf{a}_v^{(t)} + \mathbf{U}^z \mathbf{h}_v^{(t-1)}\right) \quad (3)$$

$$\mathbf{r}_v^t = \sigma\left(\mathbf{W}^r \mathbf{a}_v^{(t)} + \mathbf{U}^r \mathbf{h}_v^{(t-1)}\right) \quad (4)$$

$$\widetilde{\mathbf{h}_v^{(t)}} = \tanh\left(\mathbf{W}\mathbf{a}_v^{(t)} + \mathbf{U}\left(\mathbf{r}_v^t \odot \mathbf{h}_v^{(t-1)}\right)\right) \quad (5)$$

$$\mathbf{h}_v^{(t)} = (1 - \mathbf{z}_v^t) \odot \mathbf{h}_v^{(t-1)} + \mathbf{z}_v^t \odot \widetilde{\mathbf{h}_v^{(t)}}. \quad (6)$$

Gated Graph Neural Networks (GG-NNs)

❖ Output Models

- 노드 단위 output
  - ✓ 마지막 T시점 노드 v의 representation과 node annotation을 입력
  - ✓ g: neural network

$$o_v = \mathrm{g}\left(h_v^{(T)}, x_v\right)$$

- 그래프 단위 output
  - ✓ i: 노드 별 가중치를 계산 (neural network)
  - ✓ j: 노드 별 출력 계산 (neural network)

$$\mathbf{h}_{\mathcal{G}} = \tanh\left(\sum_{v\in\mathcal{V}} \sigma\left(i(\mathbf{h}_v^{(T)}, \boldsymbol{x}_v)\right) \odot \tanh\left(j(\mathbf{h}_v^{(T)}, \boldsymbol{x}_v)\right)\right), \tag{7}$$

Gated Graph Sequence Neural Networks (GGS-NNs)

❖ Gated Graph Sequence Neural Networks

- 두개의 GG-NN사용
  - ✓ $F_o^{(k)}$: $X^{(k)}$로부터 k번째 출력($o^{(k)}$) 예측
  - ✓ $F_X^{(k)}$: $X^{(k)}$로부터 다음시점 node annotation($X^{(k+1)}$) 예측

k: 전체 데이터에 대한 time index
t: GG-NN 내부의 time index

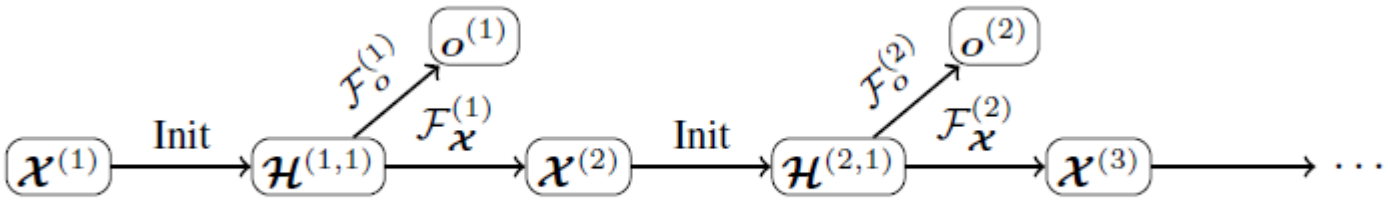


Figure 2: Architecture of GGS-NN models.

$X^{(k)}$: k시점 node annotation
$H^{(k,1)}$: k시점 node annotation으로 부터 initialize된 값(GG-NN 1번식 참고)

$$\mathbf{h}_v^{(1)} = [\boldsymbol{x}_v^\top, \mathbf{0}]^\top \qquad (1)$$

Experiments

❖ bAbI task

- AI 모델의 종합적인 사고 능력을 판단할 수 있는 20가지 task
- Example
  - ✓ Node : 대문자 알파벳 (A, B, C …) / Edge : is, has_fear (edge 연결 타입)
  - ✓ eval 이전까지 활용해 graph 구축 / eval 이후 정보로 학습 진행
  - ✓ eval B has_fear H

→ B를 1 나머지 node annotation 0으로 설정 후 H의 노드 output이 1이 되도록 학습 진행

```
D is A
B is E
A has_fear F
G is F
E has_fear H
F has_fear A
H has_fear A
C is H
eval B has_fear      H
eval G has_fear      A
eval C has_fear      A
eval D has_fear      F
```

Experiments

❖ bAbI task

- AI 모델의 종합적인 사고 능력을 판단할 수 있는 20가지 task
- Example
  - ✓ RNN과 LSTM의 경우 각 문장을 special token으로 변환 후 모델에 입력
  - ✓ n<id>: node 번호 / e<id>: edge 타입 / eol: end of line / q<id>: 질문 타입 / ans: 정답

```
n6 e1 n1 eol n6 e1 n5 eol n1 e1 n2 eol n4 e1 n5 eol n3 e1 n4
eol n3 e1 n5 eol n6 e1 n4 eol q1 n6 n2 ans 1
```

Experiments

❖ bAbI task

- 다양한 task에서 GG-NN이 RNN, LSTM 보다 좋은 성능을 보임

Task	RNN	LSTM	GG-NN
bAbI Task 4	97.3±1.9 (250)	97.4±2.0 (250)	100.0±0.0 (50)
bAbI Task 15	48.6±1.9 (950)	50.3±1.3 (950)	100.0±0.0 (50)
bAbI Task 16	33.0±1.9 (950)	37.5±0.9 (950)	100.0±0.0 (50)
bAbI Task 18	88.9±0.9 (950)	88.9±0.8 (950)	100.0±0.0 (50)

- 순차적 출력이 필요한 bAbI Task 19(path finding)에서도 GGS-NNs이 좋은 성능을 보임

Task	RNN	LSTM	GGS-NNs		
bAbI Task 19	24.7±2.7 (950)	28.2±1.3 (950)	71.1±14.7 (50)	92.5±5.9 (100)	99.0±1.1 (250)
Shortest Path	9.7±1.7 (950)	10.5±1.2 (950)	100.0± 0.0 (50)		
Eulerian Circuit	0.3±0.2 (950)	0.1±0.2 (950)	100.0± 0.0 (50)		


Thank You