Graph Neural Networks and “Learning Graphical State Transitions”

Zafarali Ahmed

Deep Learning + Linguistics Reading Group,

March 1st 2018

RLLab
McGill University

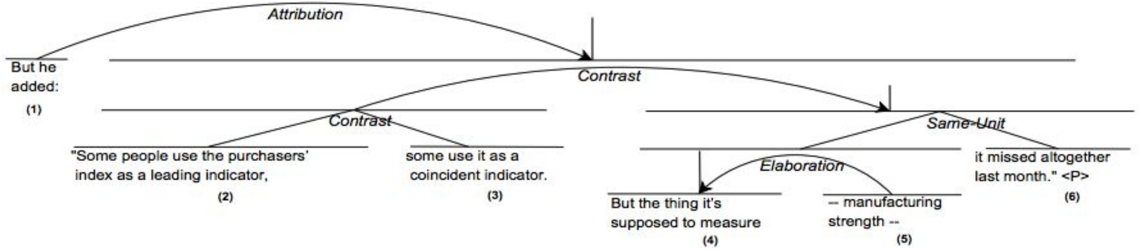
Overview

1. Motivation
2. Task (bAbI)
3. Background
    a. Gated Recurrent Unit
    b. Graph Neural Networks
    c. Gated Graph Neural Networks
4. Learning Graphical State Transitions

Motivation

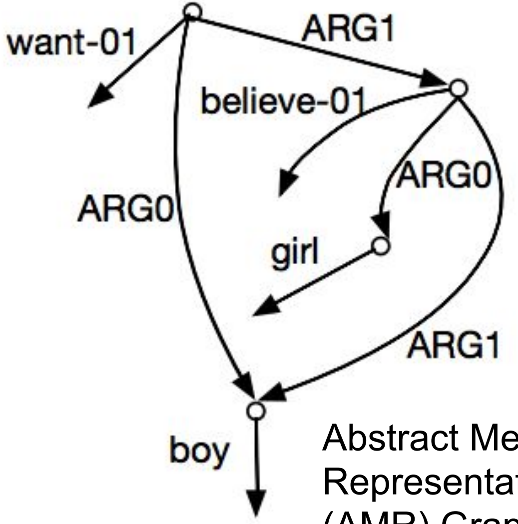
Motivation

- Rich and beautiful linguistic theory on representing sentences and tasks as trees and graphs
- **Can we leverage *structure* of these trees/graphs to improve deep learning systems?**



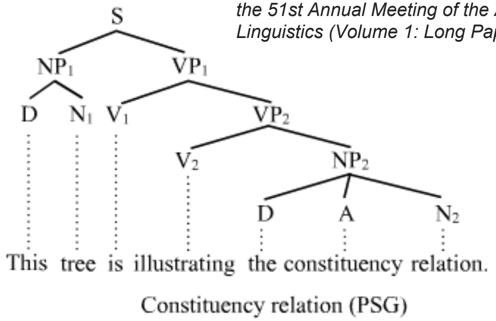
Discourse Tree

Joty, Shafiq, et al. "Combining intra-and multi-sentential rhetorical parsing for document-level discourse analysis." *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vol. 1. 2013.

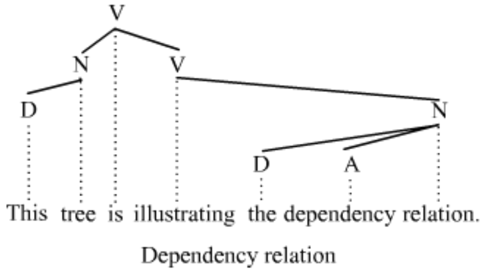


Abstract Meaning Representation (AMR) Graph

Peng, Xiaochang, Linfeng Song, and Daniel Gildea. "A synchronous hyperedge replacement grammar based approach for AMR parsing." *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*. 2015.



Constituency Parse



Dependency Parse

https://en.wikipedia.org/wiki/Phrase_structure_grammar

Motivation

- Does adding such structure really help? Seq2Seq forever?
- Yes, leveraging the tree structure and adding concepts from linguistics helps in learning to do propositional logic

Table 2: Propositional Logic Model Accuracy.

	model	valid	test (easy)	test (hard)	test (big)	test (massive)	test (exam)
baselines	Linear BoW	52.6	51.4	50.0	49.7	50.0	52.0
	MLP BoW	57.8	57.1	51.0	55.8	49.9	56.0
benchmark models	Transformer	57.1	56.8	50.8	51.2	50.3	46.9
	ConvNet Encoders	59.3	59.7	52.6	54.9	50.4	54.0
	LSTM Encoders	68.3	68.3	58.1	61.1	52.7	70.0
	BiDirLSTM Encoders	66.6	65.8	58.2	61.5	51.6	78.0
	TreeNet Encoders	72.7	72.2	69.7	67.9	56.6	85.0
	TreeLSTM Encoders	79.1	77.8	74.2	74.2	59.3	75.0
	LSTM Traversal	62.5	61.8	56.2	57.3	50.6	61.0
	BiDirLSTM Traversal	63.3	64.0	55.0	57.9	50.5	66.0
new model	**PossibleWorldNet**	98.7	**98.6**	**96.7**	**93.9**	**73.4**	**96.0**

Recall First Order Logic

[Richard Evans, David Saxton, David Amos, Pushmeet Kohli, Edward Grefenstette, *Can Neural Networks Understand Logical Entailment?* (ICLR 2018)]

The bAbI Task

Motivation: A set of tasks that demonstrates the utility/learning process of the algorithm.

Idea: A sequence of facts (the story) followed by a question. Used to test different aspects of reasoning.

Weston, Jason, et al. "Towards ai-complete question answering: A set of prerequisite toy tasks." *arXiv preprint arXiv:1502.05698* (2015).

Visual Introduction: http://www.thespermwhale.com/jaseweston/babi/abordes-ICLR.pdf

Examples

Task 1: Single Supporting Fact
Mary went to the bathroom.
John moved to the hallway.
Mary travelled to the office.
Where is Mary? A:office

Task 2: Two Supporting Facts
John is in the playground.
John picked up the football.
Bob went to the kitchen.
Where is the football? A:playground

Task 3: Three Supporting Facts
John picked up the apple.
John went to the office.
John went to the kitchen.
John dropped the apple.
Where was the apple before the kitchen? A:office

Task 4: Two Argument Relations
The office is north of the bedroom.
The bedroom is north of the bathroom.
The kitchen is west of the garden.
What is north of the bedroom? A: office
What is the bedroom north of? A: bathroom

Task 5: Three Argument Relations
Mary gave the cake to Fred.
Fred gave the cake to Bill.
Jeff was given the milk by Bill.
Who gave the cake to Fred? A: Mary
Who did Fred give the cake to? A: Bill

Task 6: Yes/No Questions
John moved to the playground.
Daniel went to the bathroom.
John went back to the hallway.
Is John in the playground? A:no
Is Daniel in the bathroom? A:yes

Task 7: Counting
Daniel picked up the football.
Daniel dropped the football.
Daniel got the milk.
Daniel took the apple.
How many objects is Daniel holding? A: two

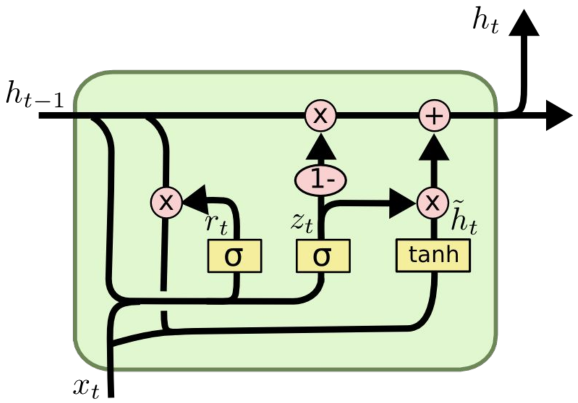
Task 8: Lists/Sets
Daniel picks up the football.
Daniel drops the newspaper.
Daniel picks up the milk.
John took the apple.
What is Daniel holding? milk, football

Task 9: Simple Negation
Sandra travelled to the office.
Fred is no longer in the office.
Is Fred in the office? A:no
Is Sandra in the office? A:yes

Task 10: Indefinite Knowledge
John is either in the classroom or the playground.
Sandra is in the garden.
Is John in the classroom? A:maybe
Is John in the office? A:no

Background

Gated Recurrent Units



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$ ← Update gate

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$ ← Reset gate

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$ ← Proposal

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$ ← Propagated

Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv:1406.1078* (2014).

Image: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Graph Neural Networks

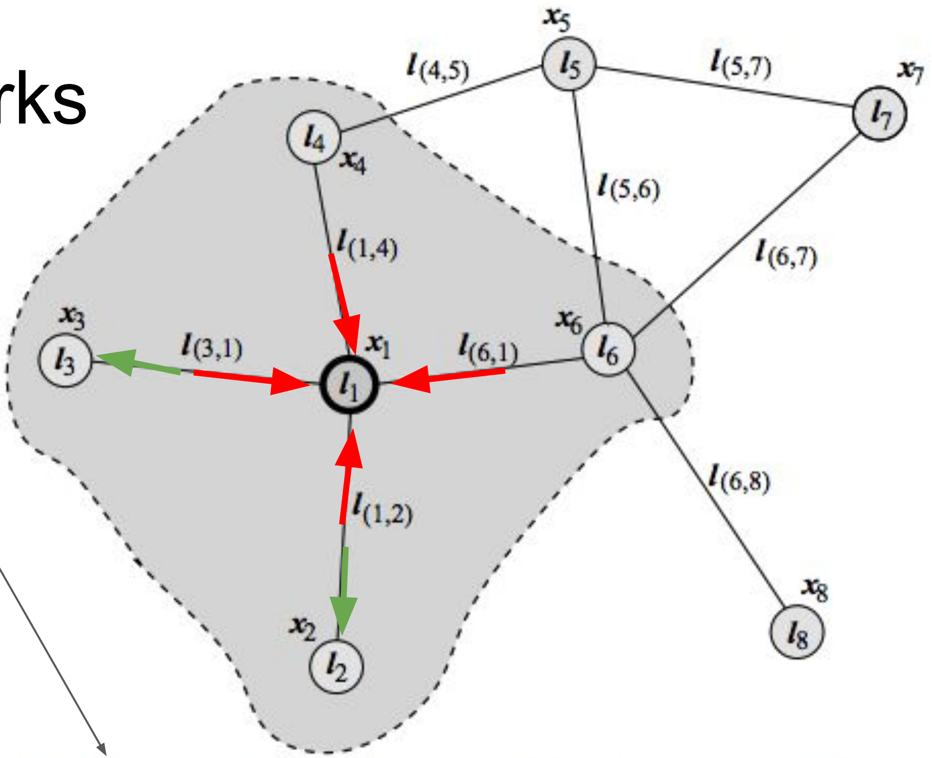
General Idea:

- Each node has a label *l*
- Each node has a hidden representation *x*
- We compute *x* based on a function *f* of the neighbours of it.
- Apply *f* multiple times (i.e. obtain a fixed point of *x*)
- Compute *g* for each node to get a node-level classification

Trained using Almeida–Pineda algorithm to find *x* followed by MSE loss and BPTT

$$\boldsymbol{x}_n(t+1) = f_{\boldsymbol{w}}(\boldsymbol{l}_n, \boldsymbol{l}_{\mathrm{co}[n]}, \boldsymbol{x}_{\mathrm{ne}[n]}(t), \boldsymbol{l}_{\mathrm{ne}[n]})$$

$$\boldsymbol{o}_n(t) = g_{\boldsymbol{w}}(\boldsymbol{x}_n(t), \boldsymbol{l}_n), \qquad n \in \boldsymbol{N}.$$



$$x_1 = f_w(l_1, \underbrace{l_{(1,2)}, l_{(3,1)}, l_{(1,4)}, l_{(6,1)}}_{l_{co[1]}}, \underbrace{x_2, x_3, x_4, x_6}_{x_{ne[1]}}, \underbrace{l_2, l_3, l_4, l_6}_{l_{ne[n]}})$$

Scarselli, Franco, et al. "The graph neural network model." *IEEE Transactions on Neural Networks* 20.1 (2009): 61-80.

Gated Graph Neural Networks

Key Idea: Replace iterative calculation of *x* by using a GRU applied for *T* timesteps!

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

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

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

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

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

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

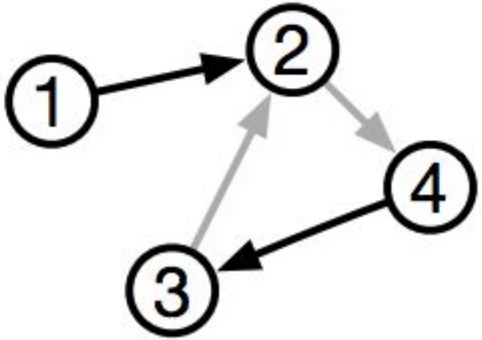
"Adjacency" matrix for node v

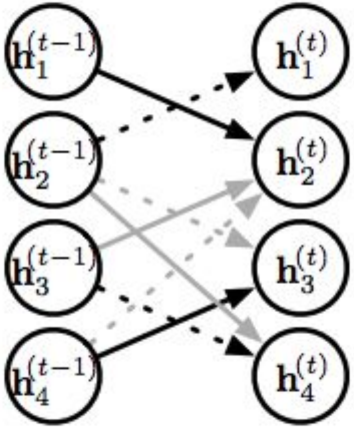
These should look familiar

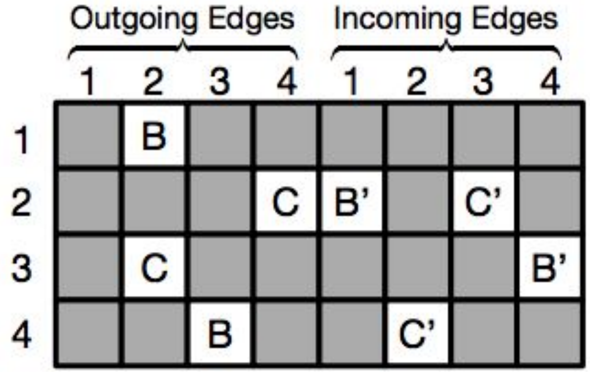
Final Graph Level Prediction: $\mathbf{h}_\mathcal{G} = \tanh\left(\sum_{v \in \mathcal{V}} \sigma\left(i(\mathbf{h}_v^{(T)}, \boldsymbol{x}_v)\right) \odot \tanh\left(j(\mathbf{h}_v^{(T)}, \boldsymbol{x}_v)\right)\right),$

[Li, Yujia, et al. "Gated graph sequence neural networks." *arXiv preprint arXiv:1511.05493* (2015).]

Gated Graph Neural Networks: Example







(Unroll 1 Timestep)

Also called a
propagation step

"Adjacency Matrix"

Gated Graph Sequence Neural Network

A Gated Graph Neural Network to produce the output

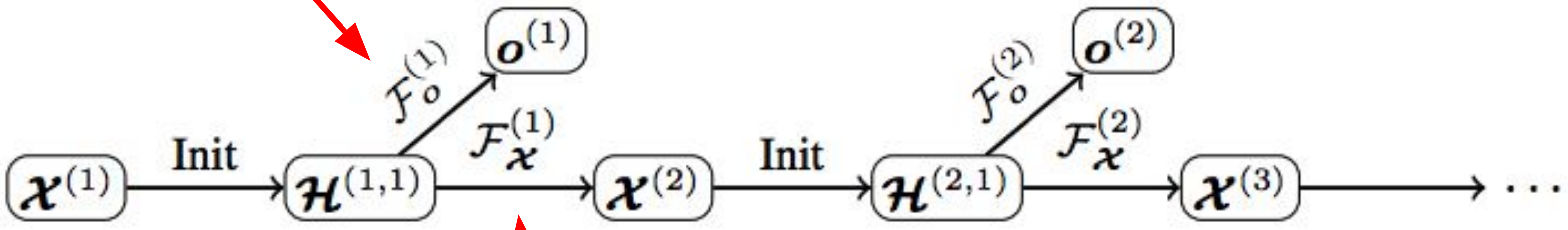


Figure 2: Architecture of GGS-NN models.

A Gated Graph Neural Network to produce the next graph state

Performance

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)

Two argument Relations
Basic Induction,
Basic Deduction
Size Reasoning

Table 1: Accuracy in percentage of different models for different tasks. Number in parentheses is number of training examples required to reach shown accuracy.

Learning Graphical State Transitions

Johnson, Daniel D. "Learning graphical state transitions." ICLR 2017 (2017).

Goal: **incrementally construct graph given natural language input**

In particular:

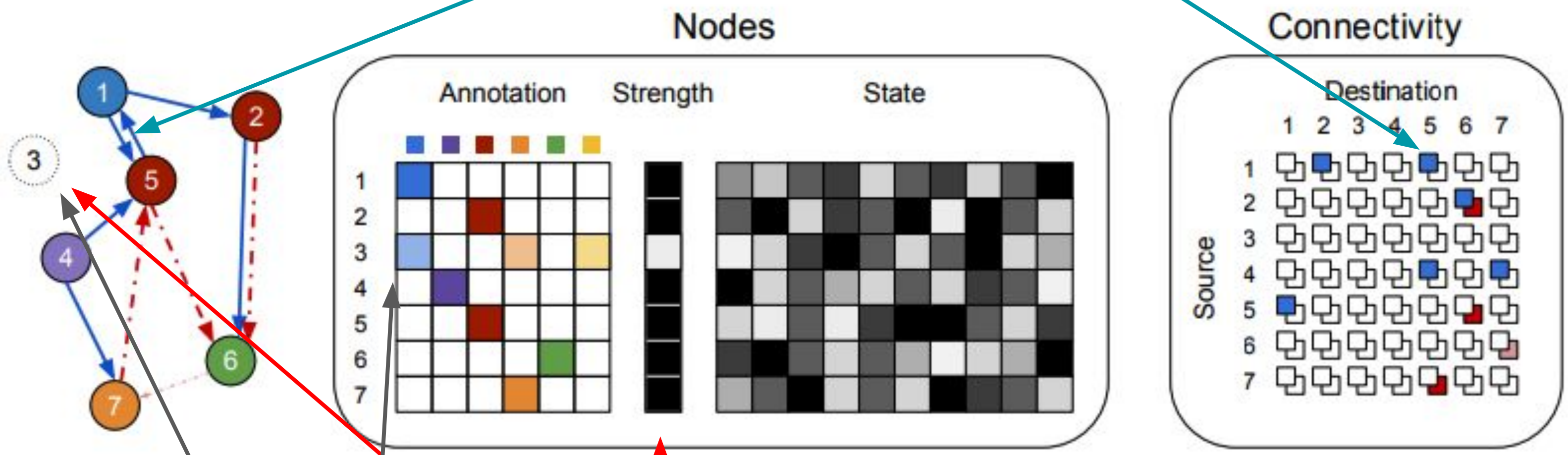
- Internal state is a graph: many tasks have this property
- Recurrent model manipulates the graph hidden state using *transformations*.
- These transformations are *differentiable*

Graph	Differentiable Graph
Nodes	Nodes
Adjacency Matrix	Connectivity Matrix Component c_vv'y represents belief that there is an edge from node v to v' of type y
N node types	N node types, s_v belief that node v should exist.
Y edge types	Y edge types
X_v, node annotation, h_v node hidden	X_v such that sum(X_v) = 1 Each component x_vj represent belief of node v being type j

Defines a “soft” graph

Example

Existence of blue edge
between 1 and 5



Node 3 is partially of type blue,
orange and yellow

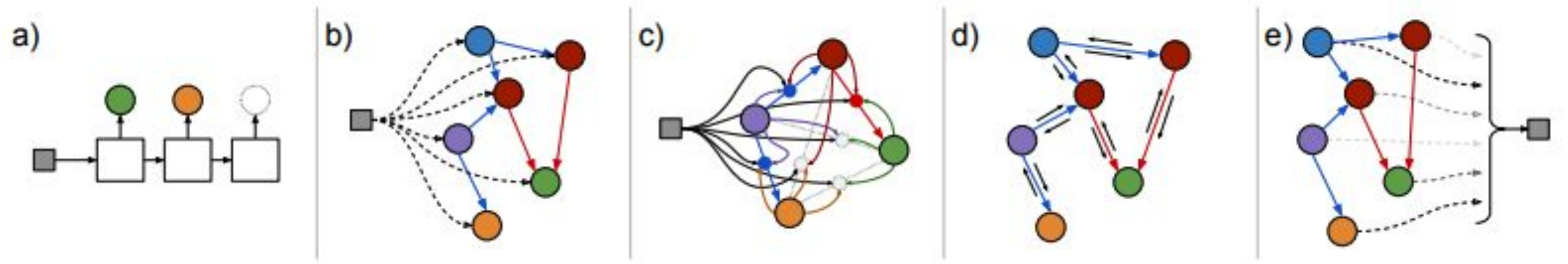
“Existence” strength

Graph Transformations

- Node Addition
  - Add new node and assign strength, annotation
- Node State Update
  - Updates the internal state
- Edge Update
  - Updates edge between pairs of nodes
- Propagate
  - Does a propagation step between all nodes in the graph
- Aggregate
  - Attention mechanism to select nodes and produce an output

“operations act on all nodes and edges in parallel” (Visualizations in the extra slides)

Graph Transformations



Add, RNN that produces nodes and strengths

State Update, uses input to update internal states

Edge Update, edges are added or removed based on input and internal states

Propagation, pass information between nodes of the graph

Aggregation, use attention mechanism to produce an output based on graph and internal states

“operations act on all nodes and edges in parallel” (More visualizations in the extra slides)

Question Answering Pseudocode

Loop over all sentences in the story

Add new nodes

Sentence representation

Internal state update

Algorithm 1 Graph Transformation Pseudocode

$\mathcal{G} \leftarrow \varnothing$
for k from 1 to K **do**
  $\mathcal{G} \leftarrow \mathcal{T}_{\mathbf{h}}(\mathcal{G}, \mathbf{i}^{(k)})$
  if direct reference enabled **then**
    $\mathcal{G} \leftarrow \mathcal{T}_{\mathbf{h},\text{direct}}(\mathcal{G}, \mathbf{D}^{(k)})$
  end if
  if intermediate propagation enabled **then**
    $\mathcal{G} \leftarrow \mathcal{T}_{\text{prop}}(\mathcal{G})$
  end if
  $\mathbf{h}_{\mathcal{G}}^{\text{add}} \leftarrow \mathcal{T}_{\text{repr}}(\mathcal{G})$
  $\mathcal{G} \leftarrow \mathcal{T}_{\text{add}}(\mathcal{G}, [\mathbf{i}^{(k)}\ \mathbf{h}_{\mathcal{G}}^{\text{add}}])$
  $\mathcal{G} \leftarrow \mathcal{T}_{\mathcal{C}}(\mathcal{G}, \mathbf{i}^{(k)})$
end for
$\mathcal{G} \leftarrow \mathcal{T}_{\mathbf{h}}^{\text{query}}(\mathcal{G}, \mathbf{i}^{\text{query}})$
if direct reference enabled **then**
  $\mathcal{G} \leftarrow \mathcal{T}_{\mathbf{h},\text{direct}}^{\text{query}}(\mathcal{G}, \mathbf{D}^{\text{query}})$
end if
$\mathcal{G} \leftarrow \mathcal{T}_{\text{prop}}^{\text{query}}(\mathcal{G})$
$\mathbf{h}_{\mathcal{G}}^{\text{answer}} \leftarrow \mathcal{T}_{\text{repr}}^{\text{query}}(\mathcal{G})$
return $f_{\text{output}}(\mathbf{h}_{\mathcal{G}}^{\text{answer}})$

Message Passing

Aggregation

Training

- Supervised training to increase the likelihood of producing a correct answer
- HOWEVER:
    - Author was not able to make internal states mean anything useful *for humans*.
- **Strong Supervision**
    - Provide the correct graph at train time and minimize loss between true graph and hidden graph

$$\mathcal{L}_{\text{node}} = -\max_{\pi} \sum_{v=|\mathcal{V}_{\text{old}}|+1}^{|\mathcal{V}_{\text{new}}|} s^*_{\pi(v)} \ln(s_v) + (1 - s^*_{\pi(v)}) \ln(1 - s_v) + \mathbf{x}^*_{\pi(v)} \cdot \ln(\mathbf{x}_v).$$

    - substitute fuzzy graph with true graph

Performance

	Task	GGT-NN + direct ref	GGT-NN	LSTM	MemNN	MemN2N	EntNet
	1	**0**	**0.7**	50.0	**0**	**0**	**0.7**
	2	**0**	5.7	80.0	**0**	8.3	56.4
	3	**1.3**	12.0	80.0	**0**	40.3	69.7
Two argument Relations	4	**1.2**	**2.2**	39.0	**0**	**2.8**	**1.4**
	5	**1.6**	10.9	30.0	**2.0**	13.1	**4.6**
	6	**0**	7.7	52.0	**0**	7.6	30.0
	7	**0**	5.6	51.0	15.0	17.3	22.3
	8	**0**	**3.3**	55.0	9.0	10.0	19.2
	9	**0**	11.6	36.0	**0**	13.2	31.5
	10	**3.4**	28.6	56.0	**2.0**	15.1	15.6
	11	**0**	**0.2**	28.0	**0**	**0.9**	8.0
	12	**0.1**	**0.7**	26.0	**0**	**0.2**	**0.8**
	13	**0**	**0.8**	6.0	**0**	**0.4**	9.0
	14	**2.2**	55.1	73.0	**1.0**	**1.7**	62.9
Basic Induction,	15	**0.9**	**0**	79.0	**0**	**0**	57.8
Basic Deduction	16	**0**	**0**	77.0	**0**	**1.3**	53.2
Positional Reasoning	17	34.5	48.0	49.0	35.0	51.0	46.4
Size Reasoning	18	**2.1**	10.6	48.0	**5.0**	11.1	8.8
Path Finding	19	**0**	70.6	92.0	64.0	82.8	90.4
	20	**0**	**1.0**	9.0	**0**	**0**	**2.6**

Number of training examples needed to get >= 95%

More data efficient?

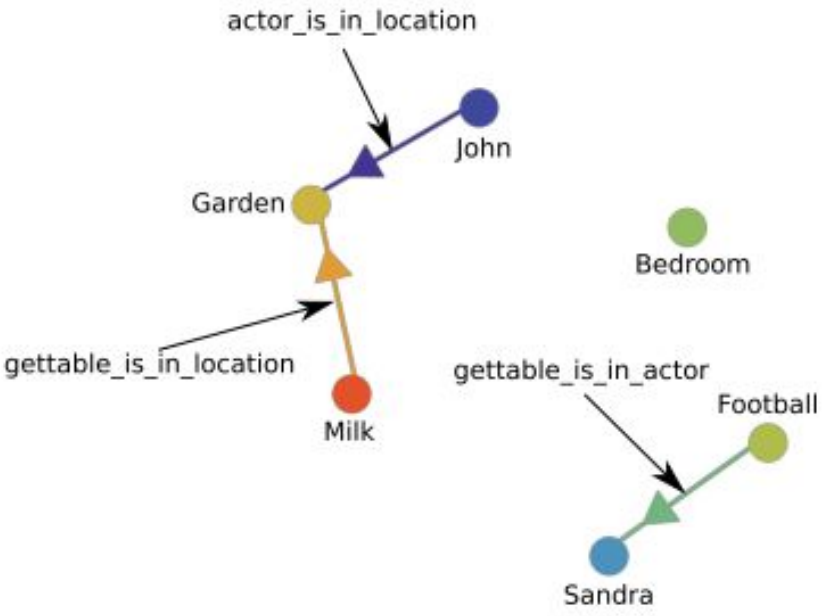
Task	GGT-NN + direct ref	GGT-NN
1 - Single Supporting Fact	100	1000
2 - Two Supporting Facts	250	-
3 - Three Supporting Facts	1000	-
4 - Two Arg. Relations	1000	1000
5 - Three Arg. Relations	500	-
6 - Yes/No Questions	100	-
7 - Counting	250	-
8 - Lists/Sets	250	1000
9 - Simple Negation	250	-
10 - Indefinite Knowledge	1000	-

Task	GGT-NN + direct ref	GGT-NN
11 - Basic Coreference	100	1000
12 - Conjunction	500	1000
13 - Compound Coref.	100	1000
14 - Time Reasoning	1000	-
15 - Basic Deduction	500	500
16 - Basic Induction	100	500
17 - Positional Reasoning	-	-
18 - Size Reasoning	1000	-
19 - Path Finding	500	-
20 - Agent's Motivations	250	250

What did the hidden graphs learn?

1. John grabbed the milk.
2. John travelled to the bedroom.
3. Sandra took the football.
4. John went to the garden.
5. *John let go of the milk.*
6. Sandra let go of the football.
7. John got the football.
8. John grabbed the milk.

Where is the milk?



Discussion Points

- How can we use this?
- What kind of tasks are suitable for this kind of work?
- How can we reduce the need for strong supervision?
- Mixing strong and weak supervision?
- Suboptimal graphs?
- Can we generalize *between* tasks in a few-shot learning sense?
    - Example strong supervision on one task, no supervision on another

References

1. Scarselli, Franco, et al. "The graph neural network model." IEEE Transactions on Neural Networks20.1 (2009): 61-80.

2. Li, Yujia, et al. "Gated graph sequence neural networks." ICLR 2016 (2016).

3. Johnson, Daniel D. "Learning graphical state transitions." ICLR 2017 (2017).

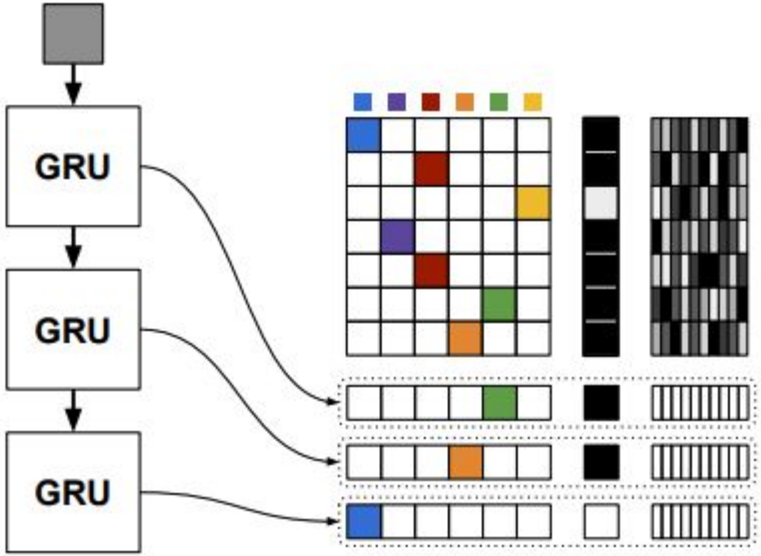
More:

Blog: http://www.hexahedria.com/2016/11/06/introducing-the-ggt-nn.html

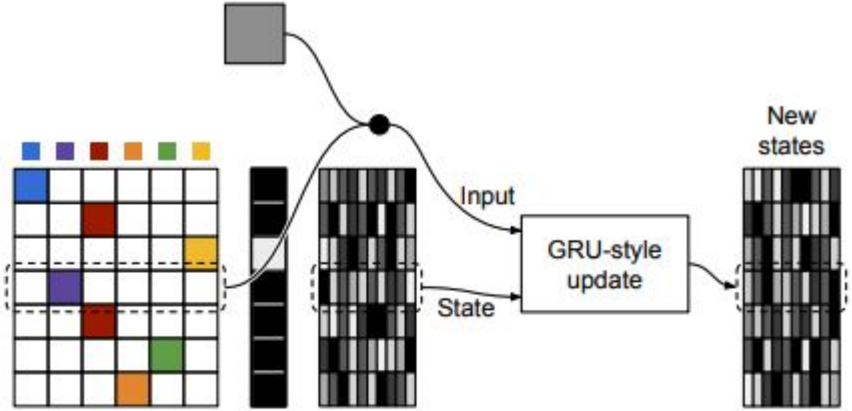
Reviews: https://openreview.net/forum?id=HJ0NvFzxl&noteId=HJ0NvFzxl

Code: https://github.com/hexahedria/gated-graph-transformer-network

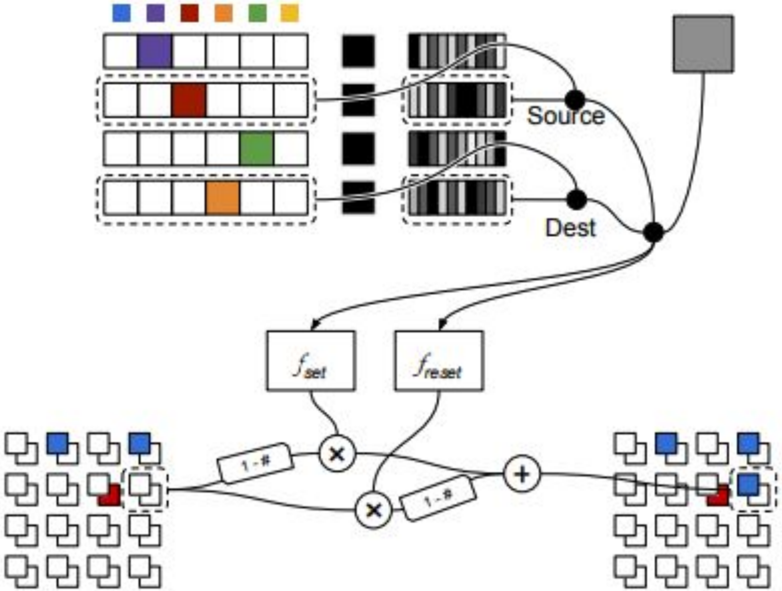
Node addition



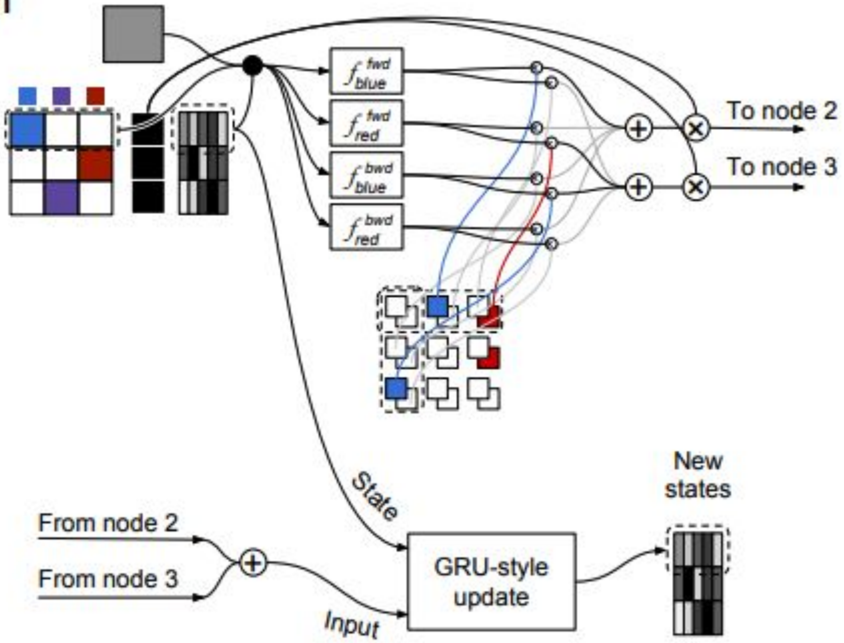
Node state update



Edge update



Propagation



Aggregation