Ordered Neurons: Integrating Tree Structures Into Recurrent Neural Networks

Best paper at ICLR 2019

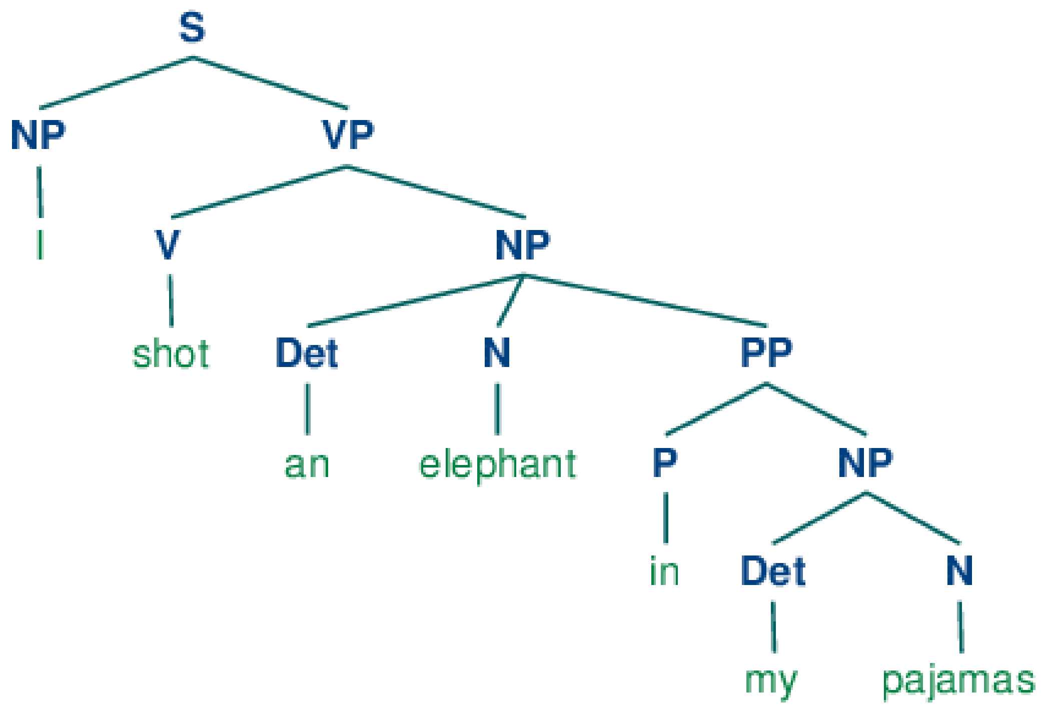
Mohammadali(Sobhan) Niknamian

CS886: Deep Learning and Natural Language Processing

Winter 2020

Motivation

- Underlying structure of language is usually tree-like
- Single words are composed to form meaningful larger units called “constituents”.
- Standard LSTM architecture does not have an explicit bias towards modeling a hierarchy of constituents.



How to predict the latent tree structure?

- Supervised Syntactic parser
  - This solution is limiting for several reasons:
    1) Few languages have annotated data for training such a parser.
    2) In some situations, syntactic rules tend to be broken (e.g. in tweets).
    3) Languages change over time, So syntax rules may evolve.

- Grammar induction: The task of learning the syntactic structure of language from raw corpora without access to expert-labeled data.
  - This is an open problem.

How to predict the latent tree structure?

- Recurrent Neural Networks (RNNs)
  - RNNs impose a chain structure on the data.
  - This assumption is in conflict with the latent non-sequential structure of language.
  - This gives rise to problems such as:
    - Capturing long-term dependencies
    - Achieving good generalization
    - Handling negation

  - However, some evidence exist that traditional LSTMs with sufficient capacity may encode the tree structure implicitly.

How to predict the latent tree structure?

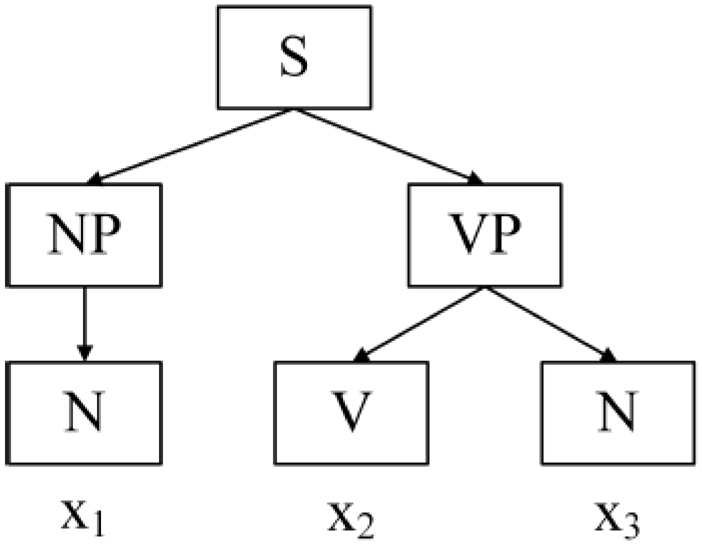
- Proposed method: ON-LSTM

  - Is able to differentiate the life cycle of information stored inside each of the neurons.
    - High ranking neurons will store long-term information which is kept for several steps.
    - Low ranking neurons will store short-term information that can be rapidly forgotten.

  - There is no strict division between high and low ranking neurons.
    - Neurons are actively allocated to store long/short information during each step of processing the input.

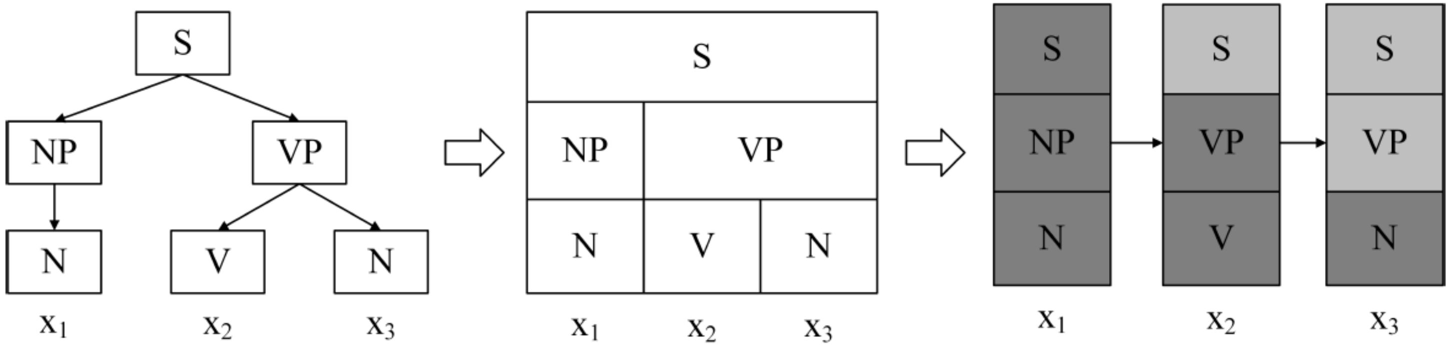
Requirements

- The hidden state h_t of our model would ideally contain information from all nodes in the path between current input x_t and the root S.
- Each node in the tree must be represented by a set of neurons in the hidden state.
- The model should dynamically reallocate the dimensions of the hidden state to each node.



Ordered neurons

- An inductive bias that forces neurons in the cell state of the LSTM to represent information at different time scales.
  - High ranking neurons contain long-term information
  - Low ranking neurons contain short-term information
- To erase (or update) high-ranking neurons, the model should first erase (or update) all lower-ranking neurons.
- The differentiation between low and high ranking neurons in learnt in a data-driven fashion and determined in each time step.



(a) Constituency tree (b) Block view (c) ON-LSTM cell states

ON-LSTM: general architecture

- The new model uses an architecture similar to the standard LSTM.
- The only difference is in the update function of cell state c_t.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$
$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$
$$\hat{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c)$$

$$h_t = o_t \circ \tanh(c_t)$$

$$\tilde{f}_t = \mathrm{cumax}(W_{\tilde{f}} x_t + U_{\tilde{f}} h_{t-1} + b_{\tilde{f}})$$
$$\tilde{i}_t = 1 - \mathrm{cumax}(W_{\tilde{i}} x_t + U_{\tilde{i}} h_{t-1} + b_{\tilde{i}})$$
$$\omega_t = \tilde{f}_t \circ \tilde{i}_t$$
$$\hat{f}_t = f_t \circ \omega_t + (\tilde{f}_t - \omega_t) = \tilde{f}_t \circ (f_t \circ \tilde{i}_t + 1 - \tilde{i}_t)$$
$$\hat{i}_t = i_t \circ \omega_t + (\tilde{i}_t - \omega_t) = \tilde{i}_t \circ (i_t \circ \tilde{f}_t + 1 - \tilde{f}_t)$$
$$c_t = \hat{f}_t \circ c_{t-1} + \hat{i}_t \circ \hat{c}_t$$

Activation function: cumax()

- Input is a numerical vector
- It is the cumulative sum of the softmax of the input vector

$$\hat{g} = \text{cumax}(\ldots) = \text{cumsum}(\text{softmax}(\ldots))$$

- We could approximate this as a binary gate $g = (0, \ldots, 0, 1, \ldots, 1)$.
- This binary gate splits the cell state into two segments: 0-segment and 1-segment.
- The model can apply different update rules on the two segment to differentiate long/short-term information.

Intuition behind new update rules

- We will explain this with an example:

$$c_t = (\underbrace{\quad,\quad,\quad,\quad}_{\text{Short-term info}}\quad,\underbrace{\quad,\quad,\quad,\quad}_{\text{long-term info}})$$

$$\tilde{f}_t = (\underbrace{0,\ 0,\ 0,}_{\substack{\text{We are done with} \\ \text{these neurons}}}\ \underbrace{1,\ 1,\ 1,\ 1,\ 1,\ 1}_{\substack{\text{We want to keep} \\ \text{these neurons}}})$$

$$\tilde{i}_t = (\underbrace{1,\ 1,\ 1,\ 1,\ 1,\ 1,}_{\substack{\text{We need these neurons from the current} \\ \text{state to be passed to future states}}}\ 0,\ 0,\ 0)$$

$$\omega_t = (0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0)$$

$$\hat{f}_t = (0,\ 0,\ 0,\ \underbrace{\quad,\quad,\quad,}_{f_t(i:j)}\ 1,\ 1,\ 1)$$

$$\tilde{f}_t = \mathrm{cumax}(W_{\tilde{f}} x_t + U_{\tilde{f}} h_{t-1} + b_{\tilde{f}})$$

$$\tilde{i}_t = 1 - \mathrm{cumax}(W_{\tilde{i}} x_t + U_{\tilde{i}} h_{t-1} + b_{\tilde{i}})$$

$$\omega_t = \tilde{f}_t \circ \tilde{i}_t$$

$$\hat{f}_t = f_t \circ \omega_t + (\tilde{f}_t - \omega_t) = \tilde{f}_t \circ (f_t \circ \tilde{i}_t + 1 - \tilde{i}_t)$$

$$\hat{i}_t = i_t \circ \omega_t + (\tilde{i}_t - \omega_t) = \tilde{i}_t \circ (i_t \circ \tilde{f}_t + 1 - \tilde{f}_t)$$

$$c_t = \hat{f}_t \circ c_{t-1} + \hat{i}_t \circ \hat{c}_t$$

Experiment: Language Modeling

- Perplexity on the Penn TreeBank (PTB) dataset.
- Perplexity measures the ability of a model in predicting the next word in a sentence (lower is better).

Model	Parameters	Validation	Test
Zaremba et al. (2014) - LSTM (large)	66M	82.2	78.4
Gal & Ghahramani (2016) - Variational LSTM (large, MC)	66M	–	73.4
Kim et al. (2016) - CharCNN	19M	–	78.9
Merity et al. (2016) - Pointer Sentinel-LSTM	21M	72.4	70.9
Grave et al. (2016) - LSTM	–	–	82.3
Grave et al. (2016) - LSTM + continuous cache pointer	–	–	72.1
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	51M	71.1	68.5
Zilly et al. (2016) - Variational RHN (tied)	23M	67.9	65.4
Zoph & Le (2016) - NAS Cell (tied)	54M	–	62.4
Shen et al. (2017) - PRPN-LM	–	–	62.0
Melis et al. (2017) - 4-layer skip connection LSTM (tied)	24M	60.9	58.3
Merity et al. (2017) - AWD-LSTM - 3-layer LSTM (tied)	24M	60.0	57.3
ON-LSTM - 3-layer (tied)	25M	**58.29 ± 0.10**	**56.17 ± 0.12**
Yang et al. (2017) - AWD-LSTM-MoS*	22M	56.5	54.4

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i \mid w_1 \ldots w_{i-1})}}$$

Best models based on perplexity

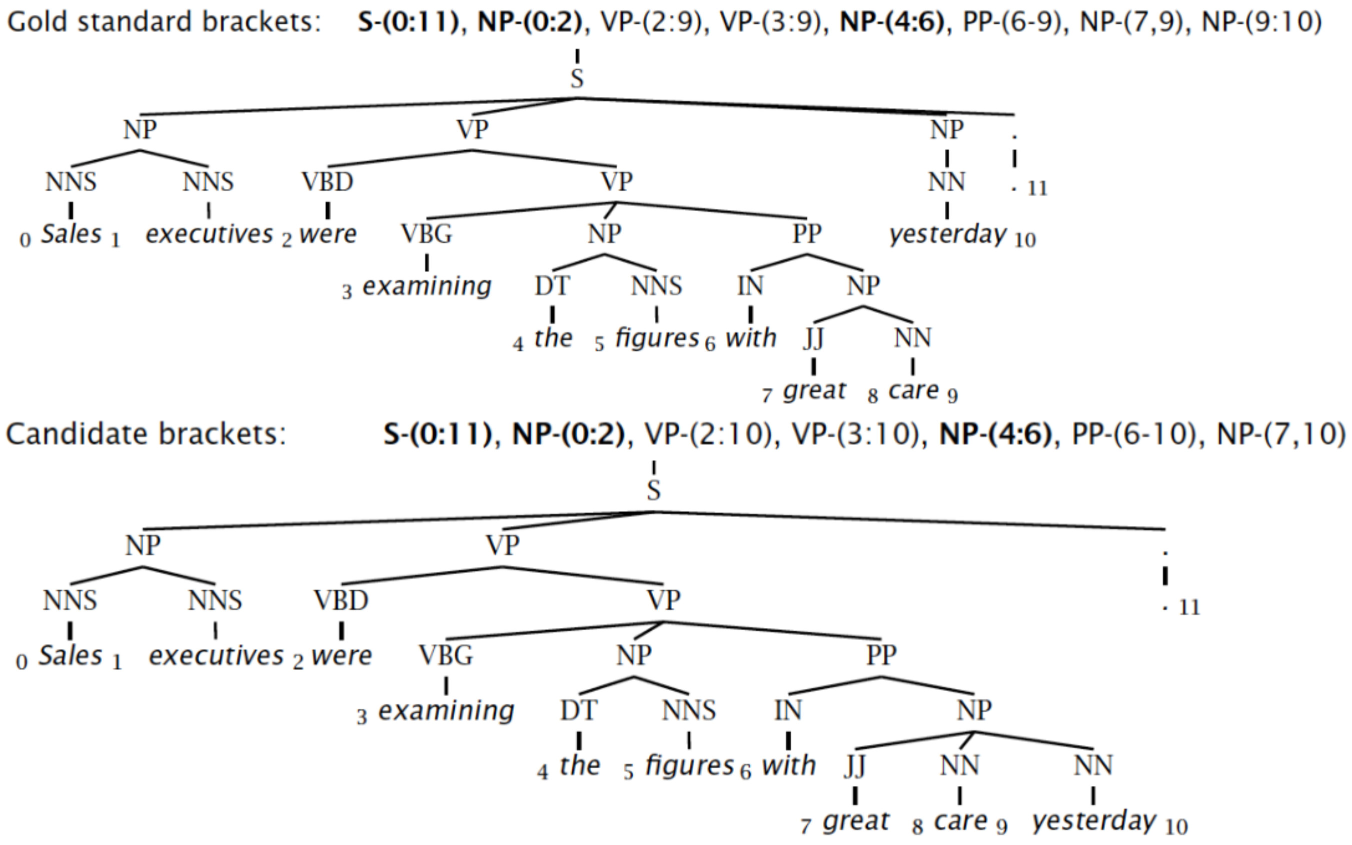
Model	Validation perplexity	Test perplexity	Number of params	Paper / Source	Code
Mogrifier LSTM + dynamic eval (Melis et al., 2019)	44.9	44.8	24M	Mogrifier LSTM	Official
AdvSoft + AWD-LSTM-MoS + dynamic eval (Wang et al., 2019)	46.63	46.01	22M	Improving Neural Language Modeling via Adversarial Training	Official
FRAGE + AWD-LSTM-MoS + dynamic eval (Gong et al., 2018)	47.38	46.54	22M	FRAGE: Frequency-Agnostic Word Representation	Official

http://nlpprogress.com/english/language_modeling.html

Experiment: Unsupervised Constituency Parsing

- Compares the latent tree structure induced by the model with those annotated by human experts.
- Lets consider d_t^f be the split point in the master forget gate $\tilde{f}_t$ in time step t.
- We sort the $\{d_t^f\}$ in the decreasing order. For the first d_t^f we split the sequence into constituents $((x_{<i}),\left(x_i,(x_{>i})\right))$. Then we repeat this recursively for constituents $(x_{<i})$ and $(x_{>i})$.

Evaluating constituency parsing



Gold standard brackets:

S-(0:11), NP-(0:2), VP-(2:9), VP-(3:9), **NP-(4:6)**, PP-(6-9), NP-(7,9), NP-(9:10)

Candidate brackets:

S-(0:11), **NP-(0:2)**, VP-(2:10), VP-(3:10), **NP-(4:6)**, PP-(6-10), NP-(7,10)

- Precision: 3/7 = 42.9%
- Recall: 3/8 = 37.5%
- F1: 40%

Experiment: Unsupervised Constituency parsing

Model	**Training Data**	**Training Object**	**Vocab Size**	**Parsing F1** WSJ10 $\mu\,(\sigma)$	WSJ10 max	WSJ $\mu\,(\sigma)$	WSJ max	**Depth WSJ**	**Accuracy on WSJ by Tag** ADJP	NP	PP	INTJ
PRPN-UP	AllNLI Train	LM	76k	66.3 (0.8)	68.5	38.3 (0.5)	39.8	5.8	28.7	65.5	32.7	*0.0*
PRPN-LM	AllNLI Train	LM	76k	52.4 (4.9)	58.1	35.0 (5.4)	42.8	6.1	37.8	59.7	**61.5**	**100.0**
PRPN-UP	WSJ Train	LM	15.8k	62.2 (3.9)	70.3	26.0 (2.3)	32.8	5.8	24.8	54.4	17.8	*0.0*
PRPN-LM	WSJ Train	LM	10k	70.5 (0.4)	71.3	37.4 (0.3)	38.1	5.9	26.2	**63.9**	24.4	*0.0*
ON-LSTM 1st-layer	WSJ Train	LM	10k	35.2 (4.1)	42.8	20.0 (2.8)	24.0	5.6	38.1	23.8	18.3	**100.0**
ON-LSTM 2nd-layer	WSJ Train	LM	10k	65.1 (1.7)	66.8	47.7 (1.5)	**49.4**	5.6	**46.2**	61.4	55.4	*0.0*
ON-LSTM 3rd-layer	WSJ Train	LM	10k	54.0 (3.9)	57.6	36.6 (3.3)	40.4	5.3	44.8	57.5	47.2	*0.0*
300D ST-Gumbel	AllNLI Train	NLI	–	–	–	19.0 (1.0)	20.1	–	*15.6*	*18.8*	*9.9*	59.4
w/o Leaf GRU	AllNLI Train	NLI	–	–	–	22.8 (1.6)	25.0	–	18.9	24.1	*14.2*	51.8
300D RL-SPINN	AllNLI Train	NLI	–	–	–	*13.2 (0.0)*	*13.2*	–	*1.7*	*10.8*	*4.6*	50.6
w/o Leaf GRU	AllNLI Train	NLI	–	–	–	*13.1 (0.1)*	*13.2*	–	*1.6*	*10.9*	*4.6*	50.0
CCM	WSJ10 Full	–	–	–	71.9	–	–	–	–	–	–	–
DMV+CCM	WSJ10 Full	–	–	–	77.6	–	–	–	–	–	–	–
UML-DOP	WSJ10 Full	–	–	–	**82.9**	–	–	–	–	–	–	–
Random Trees	–	–	–	31.7 (0.3)	32.2	18.4 (0.1)	18.6	5.3	17.4	22.3	16.0	40.4
Balanced Trees	–	–	–	43.4 (0.0)	43.4	24.5 (0.0)	24.5	4.6	22.1	*20.2*	*9.3*	55.9
Left Branching	–	–	–	*19.6 (0.0)*	*19.6*	*9.0 (0.0)*	*9.0*	12.4	–	–	–	–
Right Branching	–	–	–	56.6 (0.0)	56.6	39.8 (0.0)	39.8	12.4	–	–	–	–

Experiment: Unsupervised Constituency parsing

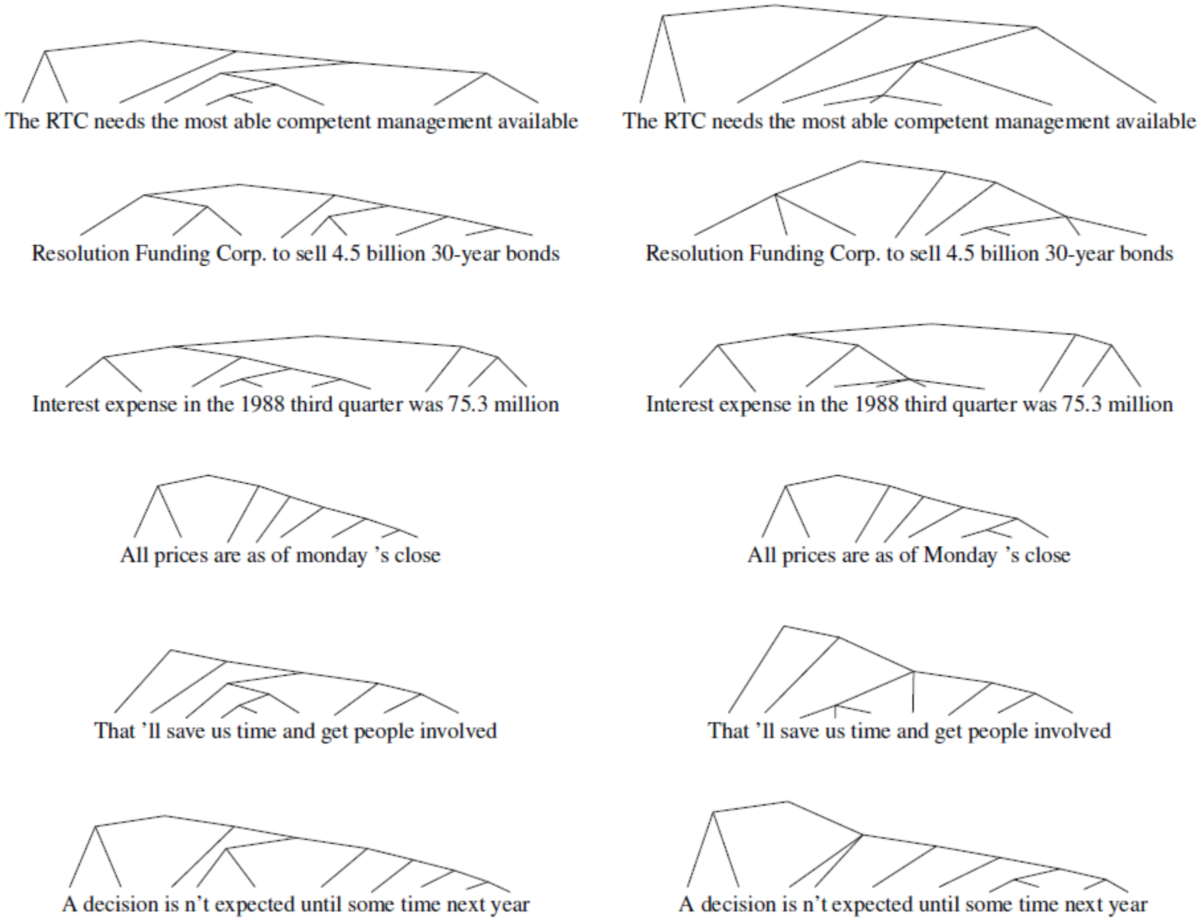


Figure A.1: *Left* parses are from the 2nd layer of the ON-LSTM model, *Right* parses are converted from human expert annotations (removing all punctuations).

Experiment: Targeted Syntactic Evaluation

- A collection of tasks that evaluate language models along three different structure-sensitive linguistic phenomena:
  1) Subject-verb agreement
  2) Reflexive anaphora
  3) Negative polarity items
- Given a large number of minimally different pairs of a grammatical and an ungrammatical sentence, the model should assign higher probability to the grammatical sentence.

a. The bankers knew the officer smiles.
b. *The bankers knew the officer smile.

a. The bankers thought the pilot embarrassed himself.
b. *The bankers thought the pilot embarrassed themselves.

a. No authors that the security guards like have ever been famous.
b. *The authors that no security guards like have ever been famous.

Experiment: Targeted Syntactic Evaluation

- Long-term dependency means that an unrelated phrase exist between the targeted pairs of words.
- The paper states that the reason standard LSTM performs better on short-term dependencies is due to the small number units in the hidden states of the ON-LSTM, which is insufficient to take into account both long and short-term information.

	ON-LSTM	LSTM
Short-Term Dependency		
SUBJECT-VERB AGREEMENT:		
Simple	0.99	**1.00**
In a sentential complement	0.95	**0.98**
Short VP coordination	0.89	**0.92**
In an object relative clause	0.84	**0.88**
In an object relative (no *that*)	0.78	**0.81**
REFLEXIVE ANAPHORA:		
Simple	**0.89**	0.82
In a sentential complement	**0.86**	0.80
NEGATIVE POLARITY ITEMS:		
Simple (grammatical vs. intrusive)	0.18	**1.00**
Simple (intrusive vs. ungrammatical)	**0.50**	0.01
Simple (grammatical vs. ungrammatical)	0.07	**0.63**
Long-Term Dependency		
SUBJECT-VERB AGREEMENT:		
Long VP coordination	**0.74**	**0.74**
Across a prepositional phrase	0.67	**0.68**
Across a subject relative clause	**0.66**	0.60
Across an object relative clause	**0.57**	0.52
Across an object relative (no *that*)	**0.54**	0.51
REFLEXIVE ANAPHORA:		
Across a relative clause	0.57	**0.58**
NEGATIVE POLARITY ITEMS:		
Across a relative clause (grammatical vs. intrusive)	0.59	**0.95**
Across a relative clause (intrusive vs. ungrammatical)	**0.20**	0.00
Across a relative clause (grammatical vs. ungrammatical)	**0.11**	0.04

References

- Shen, Yikang, Shawn Tan, Alessandro Sordoni, and Aaron Courville. "Ordered neurons: Integrating tree structures into recurrent neural networks." *arXiv preprint arXiv:1810.09536* (2018).
- Marvin, Rebecca, and Tal Linzen. "Targeted syntactic evaluation of language models." *arXiv preprint arXiv:1808.09031* (2018).
- http://www.cs.cornell.edu/courses/cs5740/2017sp/lectures/13-parsing-const.pdf

Thank you!