Top-down Tree Long Short-Term Memory Networks

WeiYang

weiyang@godweiyang.com

www.godweiyang.com

East China Normal University
Department of Computer Science and Technology

2018.03.08

Outline

Backgrounds

- Sequence structure like LSTMs have been successfully applied to many NLP tasks.
- However, many NLP tasks exploit syntactic information operating over tree structures (e.g., dependency or constituent trees).
- So we combine the advantages of LSTM architecture and syntactic structure.
- Different from recursive neural networks, the model estimates the probability of tree structure using dependency trees.

Dependency Path

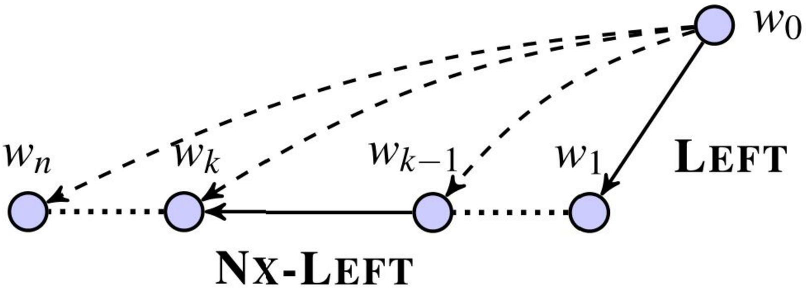

Figure: LEFT and NX-LEFT edges.

Dependency Path

(1) if w is ROOT, then $\mathcal{D}(w) = ()$

(2) if w is a **left dependent** of w^p

(a) if w is the first left dependent, then $\mathcal{D}(w) = \mathcal{D}(w^p) \| (\langle w^p, \text{LEFT} \rangle)$

(b) if w is not the first left dependent and w^s is its right adjacent sibling, then $\mathcal{D}(w) = \mathcal{D}(w^s) \| (\langle w^s, \text{NX-LEFT} \rangle)$

(3) if w is a **right dependent** of w^p

(a) if w is the first right dependent, then $\mathcal{D}(w) = \mathcal{D}(w^p) \| (\langle w^p, \text{RIGHT} \rangle)$

(b) if w is not the first right dependent and w^s is its left adjacent sibling, then $\mathcal{D}(w) = \mathcal{D}(w^s) \| (\langle w^s, \text{NX-RIGHT} \rangle)$

Figure: Algorithm to calculate dependency path.

Dependency Path

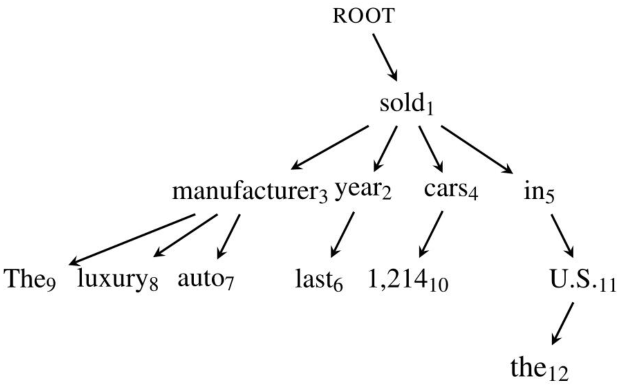

Figure: Dependency tree of the example sentence.

Tree Probability

Probability of a sentence S

$$P(S|T) = \prod_{w \in BFS(T) \backslash ROOT} P(w|\mathcal{D}(w))$$

Tree LSTMs

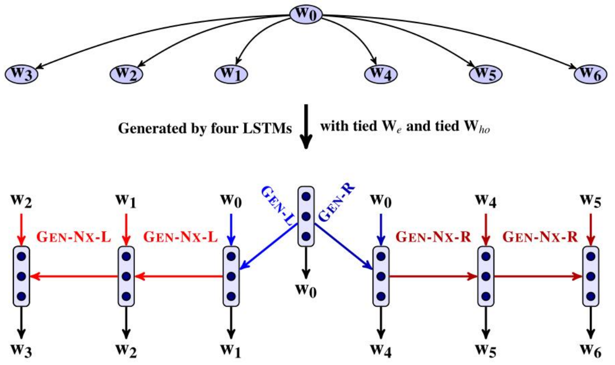

Figure: Generation process.

Tree LSTMs

Four LSTMs

GEN-L, GEN-R, GEN-NX-L, GEN-NX-R.

Computation

$$x_t = W_e \cdot e(w_{t'})$$
$$h_t = LSTM^{z_t}(x_t, H[:, t'])$$
$$H[:, t] = h_t$$
$$y_t = W_{ho} \cdot h_t$$

Probability of w_t

$$P(w_t|\mathcal{D}(w_t)) = \frac{\exp(y_{t,w_t})}{\sum_{k'=1}^{|V|} \exp(y_{t,k'})}$$

Tree LSTMs

Deep LSTMs

$$
\begin{aligned}
u_t &= \tanh(W_{ux}^{z,l} \cdot \hat{h}_t^{l-1} + W_{uh}^{z,l} \cdot \hat{h}_{t'}^{l}) \\
i_t &= \sigma(W_{ix}^{z,l} \cdot \hat{h}_t^{l-1} + W_{ih}^{z,l} \cdot \hat{h}_{t'}^{l}) \\
f_t &= \sigma(W_{fx}^{z,l} \cdot \hat{h}_t^{l-1} + W_{fh}^{z,l} \cdot \hat{h}_{t'}^{l}) \\
\hat{c}_t^l &= f_t \odot \hat{c}_{t'}^l + i_t \odot u_t \\
o_t &= \sigma(W_{ox}^{z,l} \cdot \hat{h}_t^{l-1} + W_{oh}^{z,l} \cdot \hat{h}_{t'}^{l}) \\
\hat{h}_t^l &= o_t \odot \tanh(\hat{c}_t^l)
\end{aligned}
$$

Left Dependent Tree LSTMs

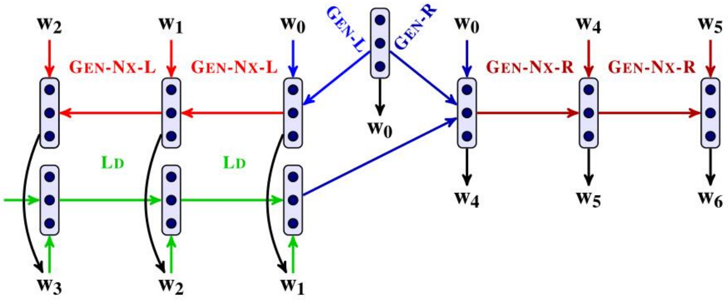

Figure: Generation process according to LDTREELSTM.

Left Dependent Tree LSTMs

Computation

$$
\begin{aligned}
m_k &= W_e \cdot e(v_{t,k}) \\
q_k &= LSTM^{LD}(m_k, q_{k-1}) \\
r_t &= \begin{bmatrix} W_e \cdot e(w_{t'}) \\ q_K \end{bmatrix} \\
h_t &= LSTM^{GEN-R}(r_t, H[:, t'])
\end{aligned}
$$

Training

Small Scale Datasets

$$\mathcal{L}^{NLL}(\theta) = -\frac{1}{|\mathcal{S}|}\sum_{S\in\mathcal{S}} \log P(S|T)$$

Large Scale Datasets

$$\mathcal{L}^{NCE}(\theta) = -\frac{1}{|\mathcal{S}|}\sum_{T\in\mathcal{S}}\sum_{t=1}^{|T|}(\log P_d(w_t, \mathcal{D}(w_t)) + \sum_{j=1}^{k}\log[1 - P_d(\tilde{w}_{t,j}, \mathcal{D}(w_t))])$$

where

$$P_d(w, D(w_t)) = \frac{\hat{P}(w|D(w_t))}{\hat{P}(w|D(w_t))+kP_n(w)}$$

$$\hat{P}(w|D(w_t)) = \frac{\exp(W_{ho}[w,:]\cdot h_t)}{\hat{Z}}$$

Microsoft Sentence Completion Challenge

Model	d	$\|\theta\|$	Accuracy
Word Vector based Models			
LSA	—	—	49.0
Skip-gram	640	102M	48.0
IVLBL	600	96.0M	55.5
Language Models			
KN5	—	—	40.0
UDepNgram	—	—	48.3
LDepNgram	—	—	50.0
RNN	300	48.1M	45.0
RNNME	300	1120M	49.3
depRNN+3gram	100	1014M	53.5
ldepRNN+4gram	200	1029M	50.7
LBL	300	48.0M	54.7
LSTM	300	29.9M	55.00
LSTM	400	40.2M	57.02
LSTM	450	45.3M	55.96
Bidirectional LSTM	200	33.2M	48.46
Bidirectional LSTM	300	50.1M	49.90
Bidirectional LSTM	400	67.3M	48.65
Model Combinations			
RNNMEs	—	—	55.4
Skip-gram + RNNMEs	—	—	58.9
Our Models			
TREELSTM	300	31.6M	55.29
LDTREELSTM	300	32.5M	57.79
TREELSTM	400	43.1M	56.73
LDTREELSTM	400	44.7M	**60.67**

Figure: Model accuracy on the MSR sentence completion task.

Dependency Parsing

Parser	Development		Test	
	UAS	LAS	UAS	LAS
MSTParser-2nd	92.20	88.78	91.63	88.44
TREELSTM	92.51	89.07	91.79	88.53
TREELSTM*	92.64	89.09	91.97	88.69
LDTREELSTM	**92.66**	**89.14**	**91.99**	**88.69**
NN parser*	92.00	89.70	91.80	89.60
S-LSTM*	**93.20**	**90.90**	**93.10**	**90.90**

Figure: Performance on reranking the top dependency trees produced by the 2nd order MSTParser.

Conclusions

Applications

- MSR sentence completion task.
- Dependency parsing.
- Text generation applications such as sentence compression and simplification.

Future

Can be apply to other types of tree structure such as constituent trees or even taxonomies.