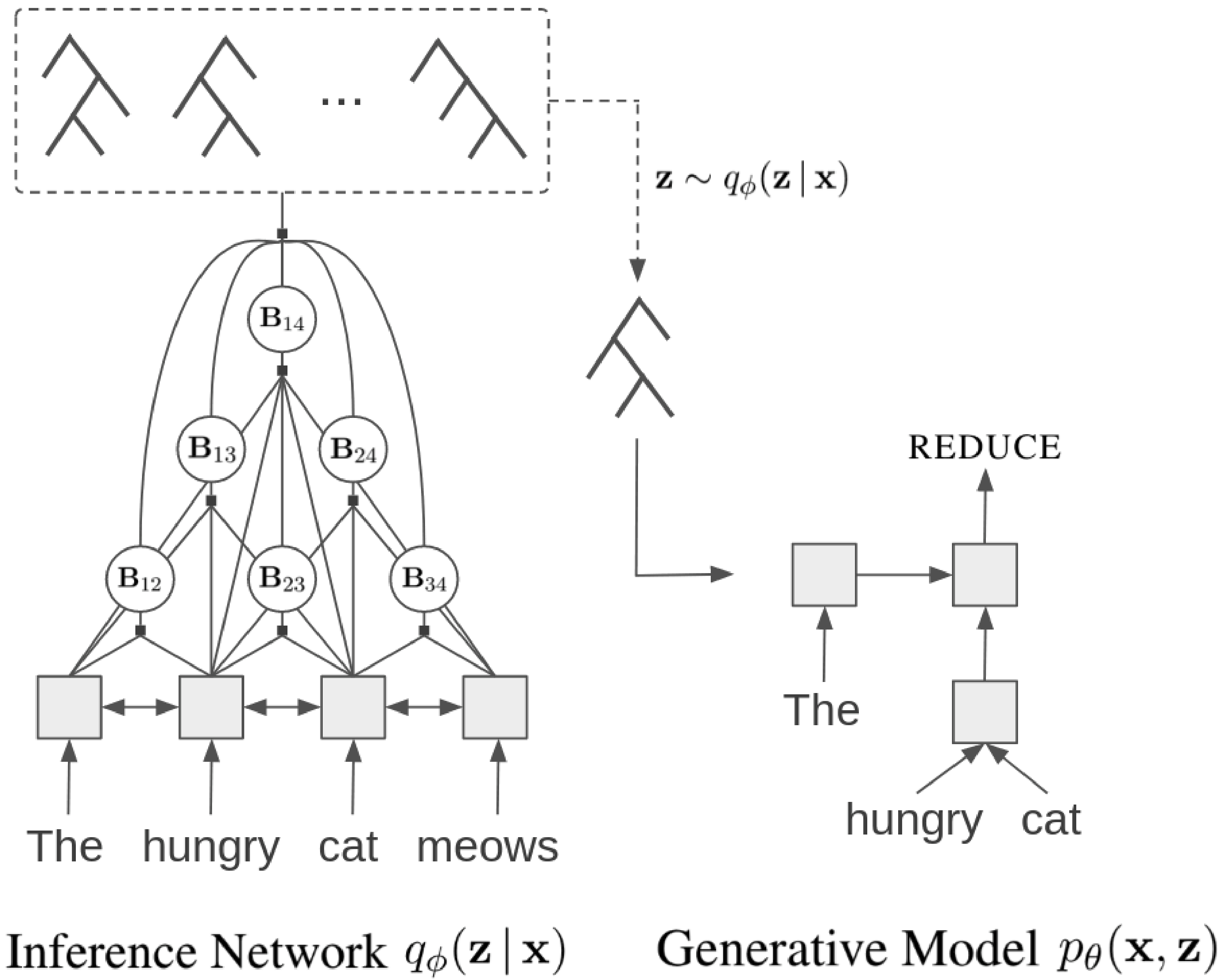
[NAACL19] Unsupervised Recurrent Neural Network Grammars

WeiYang ❤ 51184506043
weiyang@godweiyang.com
https://godweiyang.com
2019.05.09

Model Architecture



Inference Network: $q_\phi(z|x)$

Encoder:

$$f_i, b_i = BiLSTM([e_i, p_i])$$

Span scores:

$$s_{ij} = MLP([f_{j+1} - f_i; b_{i-1} - b_j])$$

Probability of a tree:

$$q_\phi(B|x) = \frac{1}{Z_T(x)} \exp(\sum_{i \leq j} B_{ij} s_{ij})$$

Partition function:

$$Z_T(x) = \sum_{B' \in \mathcal{B}_T} \exp(\sum_{i \leq j} B'_{ij} s_{ij})$$

Inside Algorithm for Calculating $Z_T(x)$

Algorithm 1 Inside algorithm for calculating $Z_T(\mathbf{x})$

procedure INSIDE(s) ▷ scores s_{ij} for $i \leq j$
  for $i := 1$ **to** T **do** ▷ length-1 spans
    $\beta[i, i] = s_{ii}$
  for $\ell := 1$ **to** $T - 1$ **do** ▷ span length
    for $i := 1$ **to** $T - \ell$ **do** ▷ span start
      $j = i + \ell$ ▷ span end
      $\beta[i, j] = \sum_{k=i}^{j-1} s_{ij} \cdot \beta[i, k] \cdot \beta[k + 1, j]$
  return $\beta[1, T]$ ▷ return partition function $Z_T(\mathbf{x})$

Some mistakes here.

Sampling from $q_\phi(z|x)$

Algorithm 2 Top-down sampling a tree from $q_\phi(\mathbf{z} \mid \mathbf{x})$

procedure SAMPLE(β) ▷ β from running INSIDE(s)
  $\mathbf{B} = \mathbf{0}$ ▷ binary matrix representation of tree
  $Q = [(1, T)]$ ▷ queue of constituents
  while Q is not empty **do**
    $(i, j) = \text{pop}(Q)$
    $\tau = \sum_{k=i}^{j-1} \beta[i, k] \cdot \beta[k + 1, j]$
    for $k := i$ to $j - 1$ **do** ▷ get distribution over splits
      $w_k = (\beta[i, k] \cdot \beta[k + 1, j])/\tau$
    $k \sim \text{Cat}([w_i, \dots, w_{j-1}])$ ▷ sample a split point
    $\mathbf{B}_{i,k} = 1, \ \mathbf{B}_{k+1,j} = 1$ ▷ update $\mathbf{B}$
    if $k > i$ **then** ▷ if left child has width > 1
      $\text{push}(Q, (i, k))$ ▷ add to queue
    if $k + 1 < j$ **then** ▷ if right child has width > 1
      $\text{push}(Q, (k + 1, j))$ ▷ add to queue
  $\mathbf{z} = f(\mathbf{B})$ ▷ $f : \mathcal{B}_T \to \mathcal{Z}_T$ maps matrix representation of tree to sequence of actions.
  return $\mathbf{z}$

Generative Model: $p_\theta(x, z)$

Action prediction:

$$p_t = \sigma(w^T h_{prev} + b)$$

SHIFT:

$$x \sim softmax(W h_{prev} + b)$$
$$h_{next} = LSTM(e_x, h_{prev})$$
$$Stack.push((h_{next}, e_x))$$

REDUCE:

$$g_{new} = TreeLSTM(g_l, g_r)$$
$$h_{new} = LSTM(g_{new}, h_{prev})$$
$$Stack.push((h_{new}, g_{new}))$$

Generative Model: $p_\theta(x, z)$

$$\log p_\theta(x, z) = \sum_{t=1}^{T} \log p_\theta(x_t | x_{<t}, z_{<n(t)})$$

$$+ \sum_{j=1}^{2T-1} \log p_\theta(z_j | x_{<m(j)}, z_{<j})$$

Supervised Learning: OK.

Unsupervised Learning: $\log p_\theta(x)$

Variational Inference

Language modeling:

$$\log p_\theta(x) = \log \sum_{z \in \mathcal{Z}_T} p_\theta(x, z)$$

Evidence Lower Bound (ELBO):

$$\log p_\theta(x) = \log \sum_{z \in \mathcal{Z}_T} p_\theta(x, z)$$

$$= \log \sum_{z \in \mathcal{Z}_T} q_\phi(z|x) \frac{p_\theta(x,z)}{q_\phi(z|x)}$$

$$= \log \mathbb{E}_{q_\phi(z|x)} \left[\frac{p_\theta(x,z)}{q_\phi(z|x)} \right]$$

$$\geq \mathbb{E}_{q_\phi(z|x)} \left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)} \right]$$

Evidence Lower Bound (ELBO)

$$ELBO = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right]$$

$$= \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(z|x)p_\theta(x)}{q_\phi(z|x)}\right]$$

$$= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x)\right] - \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right]$$

$$= \log p_\theta(x) - KL(q_\phi(z|x) \parallel p_\theta(z|x))$$

Evidence Lower Bound (ELBO)

$$ELBO = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x, z)\right] - \mathbb{H}\left[q_\phi(z|x)\right]$$

Gradient of θ:

$$\nabla_\theta ELBO = \mathbb{E}_{q_\phi(z|x)}\left[\nabla_\theta \log p_\theta(x, z)\right]$$

$$\nabla_\theta ELBO \approx \frac{1}{K}\sum_{k=1}^{K} \nabla_\theta \log p_\theta(x, z_k)$$

Gradient of ϕ:

Two parts.

Gradient of $\mathbb{H}\left[q_\phi(z|x)\right]$

Algorithm 3 Calculating the tree entropy $\mathbb{H}[q_\phi(\mathbf{z} \mid \mathbf{x})]$

```
1: procedure ENTROPY(β)                      ▷ β from running INSIDE(s)
2:   for i := 1 to T do                      ▷ initialize entropy table
3:     H[i, i] = 0
4:   for l := 1 to T − 1 do                  ▷ span length
5:     for i := 1 to T − l do                ▷ span start
6:       j = i + l                           ▷ span end
7:       τ = Σ_{u=i}^{j−1} β[i, u] · β[u + 1, j]
8:       for u := i to j − 1 do
9:         w_u = (β[i, u] · β[u + 1, j])/τ
10:      H[i, j] = Σ_{u=i}^{j−1} (H[i, u] + H[u + 1, j]
11:                  − log w_u) · w_u
12:  return H[1, T]       ▷ return tree entropy ℍ[q_φ(z | x)]
```

Gradient of $\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x,z)\right]$

$$
\begin{aligned}
&\nabla_\phi \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x,z)\right] \\
&= \nabla_\phi \sum_z q_\phi(z|x) \log p_\theta(x,z) \\
&= \sum_z \log p_\theta(x,z) \nabla_\phi q_\phi(z|x) \\
&= \sum_z q_\phi(z|x) \log p_\theta(x,z) \nabla_\phi \log q_\phi(z|x) \\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x,z) \nabla_\phi \log q_\phi(z|x)\right] \\
&\approx \frac{1}{K} \sum_{k=1}^{K} \log p_\theta(x,z_k) \nabla_\phi \log q_\phi(z_k|x)
\end{aligned}
$$

Add Baseline

$$\nabla_\phi \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x, z)\right]$$

$$\approx \frac{1}{K}\sum_{k=1}^{K} \log p_\theta(x, z_k) \nabla_\phi \log q_\phi(z_k|x)$$

$$\approx \frac{1}{K}\sum_{k=1}^{K} \left(\log p_\theta(x, z_k) - r_k\right) \nabla_\phi \log q_\phi(z_k|x)$$

where

$$r_k = \frac{1}{K-1}\sum_{j \neq k} \log p_\theta(x, z_j)$$

Final Objective Function

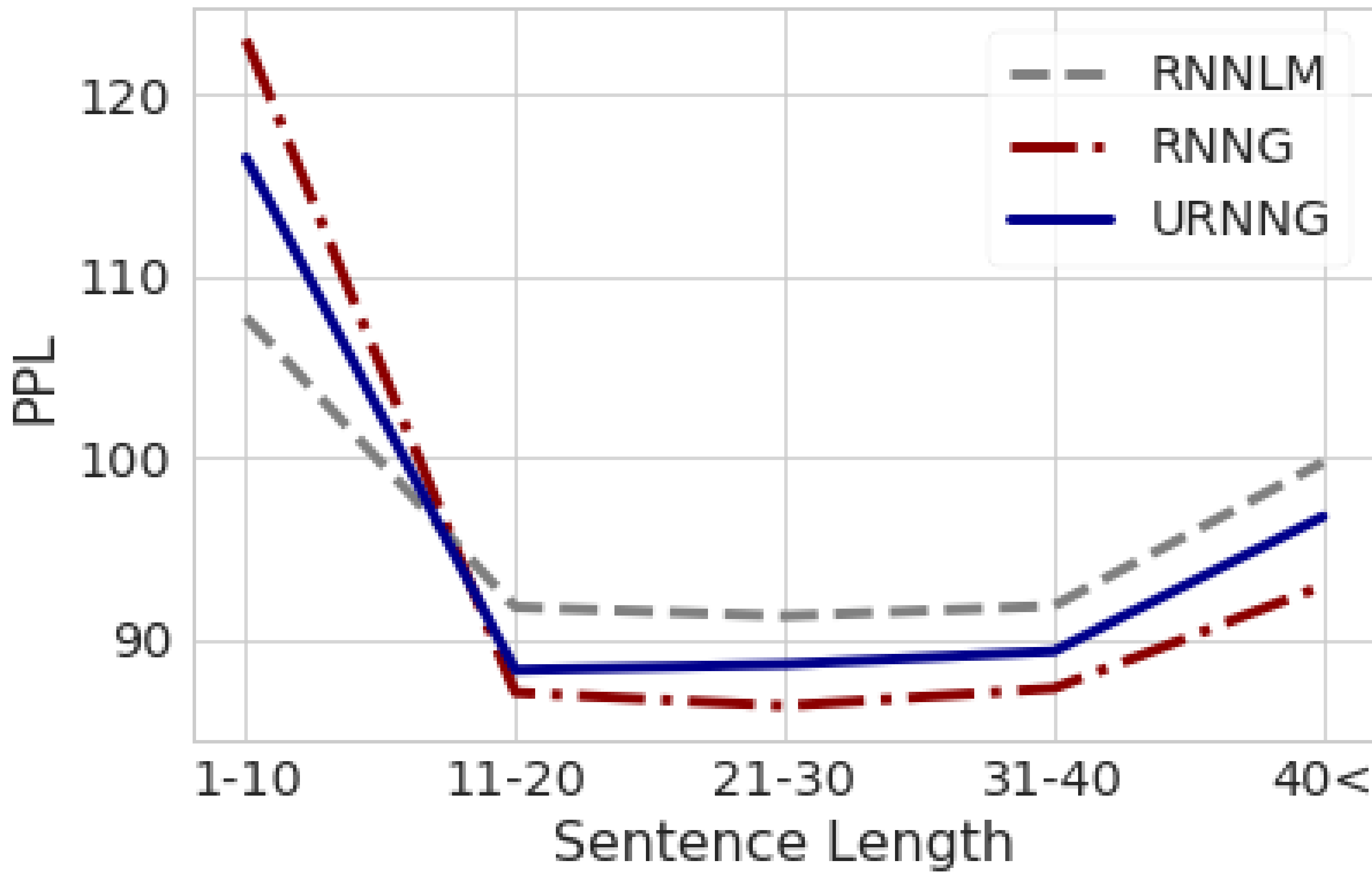
$$\frac{1}{K}\sum_{k=1}^{K}\left[\log p_\theta(x, z_k) + \color{red}{\left(\log p_\theta(x, z_k) - r_k\right)} \log q_\phi(z_k|x)\right]$$

$$-\mathbb{H}\left[q_\phi(z|x)\right]$$

Experiments

Model	PTB		CTB	
	PPL	F_1	PPL	F_1
RNNLM	93.2	–	201.3	–
PRPN (default)	126.2	32.9	290.9	32.9
PRPN (tuned)	96.7	41.2	216.0	36.1
Left Branching Trees	100.9	10.3	223.6	12.4
Right Branching Trees	93.3	34.8	203.5	20.6
Random Trees	113.2	17.0	209.1	17.4
URNNG	90.6	40.7	195.7	29.1
RNNG	88.7	68.1	193.1	52.3
RNNG $\rightarrow$ URNNG	85.9	67.7	181.1	51.9
Oracle Binary Trees	–	82.5	–	88.6

Experiments



Experiments

PTB	PPL
KN 5-gram (Dyer et al., 2016)	169.3
RNNLM (Dyer et al., 2016)	113.4
Original RNNG (Dyer et al., 2016)	102.4
Stack-only RNNG (Kuncoro et al., 2017)	101.2
Gated-Attention RNNG (Kuncoro et al., 2017)	100.9
Generative Dep. Parser (Buys and Blunsom, 2015)	138.6
RNNLM (Buys and Blunsom, 2018)	100.7
Sup. Syntactic NLM (Buys and Blunsom, 2018)	107.6
Unsup. Syntactic NLM (Buys and Blunsom, 2018)	125.2
PRPN† (Shen et al., 2018)	96.7
This work:	
RNNLM	93.2
URNNG	90.6
RNNG	88.7
RNNG → URNNG	85.9
1M Sentences	**PPL**
PRPN† (Shen et al., 2018)	77.7
RNNLM	77.4
URNNG	71.8
RNNG‡	72.9
RNNG‡ → URNNG	72.0

Ordered Neurons: Integrating Tree Structures into Recurrent Neural Networks

ICLR 2019 best paper

Neural Language Modeling by Jointly Learning Syntax and Lexicon

ICLR 2018