Paper Reading

Xinwei Geng

2018-07-09

Reinforced Self-Attention Network: a Hybrid of Hard and Soft Attention for Sequence Modeling

- Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Sen Wang, Chengqi Zhang
- PhD Candidate, University of Technology Sydney
- IJCAI 2018
- Code: https://github.com/taoshen58/ReSAN

Soft Attention vs. Hard Attention

- Soft attention
  - a categorical distribution is calculated over a sequence of element
  - only requires a small number of parameters and less computation time
  - differentiable and thus can be easily trained by end-to-end back-propagation
  - assigns small but non-zero probabilities to trivial elements

- Hard attention
  - concentrate solely on the important elements, entirely discarding the others
  - overcomes the weaknesses associated with soft attention
  - time-inefficient with sequential sampling and non-differentiable
- Soft and hard attention mechanisms might be integrated into a single model to benefit each other

Reinforced Sequence Sampling (RSS)

- Select a subset of critical tokens that provides sufficient information to complete downstream tasks
  - RSS generates an equal-length sequence of binary random variables Z to select or discard the input X

$$p(\boldsymbol{z}|\boldsymbol{x};\theta_r) = \prod_{i=1}^{n} p(z_i|\boldsymbol{x};\theta_r),$$
$$\text{where } p(z_i|\boldsymbol{x};\theta_r) = g(f(\boldsymbol{x};\theta_f)_i;\theta_g).$$

$$f(\boldsymbol{x};\theta_f)_i = [x_i; \text{pooling}(\boldsymbol{x}); x_i \odot \text{pooling}(\boldsymbol{x})],$$
$$g(h_i;\theta_g) = \text{sigmoid}(w^T \sigma(W^{(R)} h_i + b^{(R)}) + b),$$

Reinforced Self-Attention (ReSA)

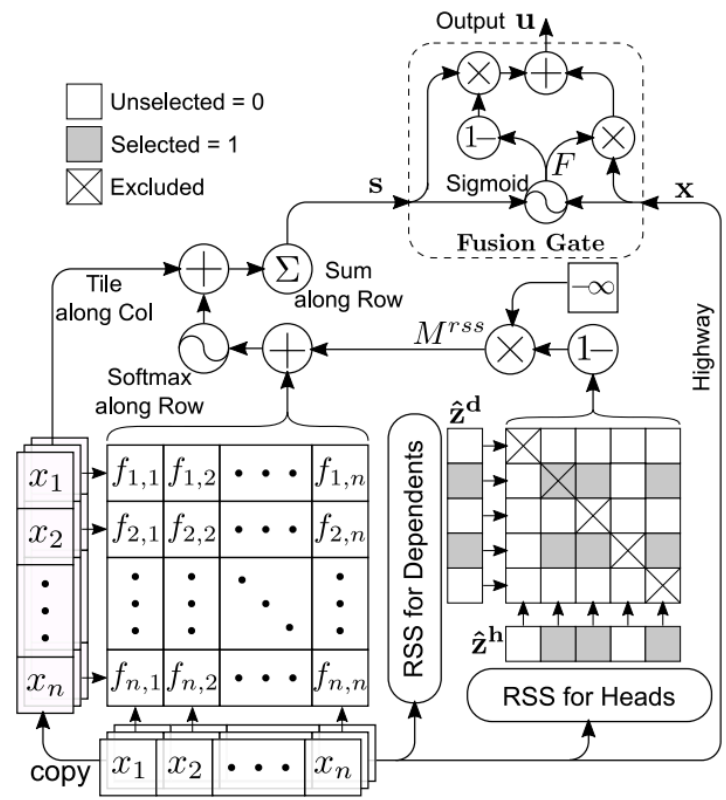
- The proposed RSS provides a sparse mask to a self-attention module that only needs to model the dependencies for the selected token pairs

$$\hat{\boldsymbol{z}}^{\boldsymbol{h}} = [\hat{z}_1^h, \dots, \hat{z}_n^h] \sim \mathrm{RSS}(\boldsymbol{x}; \theta_{rh}),$$
$$\hat{\boldsymbol{z}}^{\boldsymbol{d}} = [\hat{z}_1^d, \dots, \hat{z}_n^d] \sim \mathrm{RSS}(\boldsymbol{x}; \theta_{rd}),$$

$$M_{ij}^{rss} = \begin{cases} 0, & \hat{z}_i^d = \hat{z}_j^h = 1 \ \& \ i \neq j \\ -\infty, & \text{otherwise.} \end{cases}$$

$$f^{rss}(x_i, x_j) = f(x_i, x_j) + M_{ij}^{rss}$$

$$F = \mathrm{sigmoid}\left(W^{(f)}[\boldsymbol{x}; \boldsymbol{s}] + b^{(f)}\right),$$
$$\boldsymbol{u} = F \odot \boldsymbol{x} + (1 - F) \odot \boldsymbol{s},$$



Training

- The parameters in ReSAN can be divided into two parts, θ_r for the RSS modules and θ_s for the rest parts
- Use the cross entropy loss plus L2 regularization penalty as the loss to optimize the θ_s

$$J_s(\theta_s) = \mathbb{E}_{(\boldsymbol{x}^*, y^*) \sim \mathcal{D}}[-\log p(y = y^* | \boldsymbol{x}^*; \theta_{s,r})] + \gamma \|\theta_s\|^2,$$

- Policy gradient
  - Use the cross-entropy loss as reward
  - A penalty limiting the number of selected tokens

$$\mathcal{R} = \log p(y = y^* | \boldsymbol{x}^*; \theta_s, \theta_r) - \lambda \sum \hat{z}_i / len(\boldsymbol{x}^*),$$

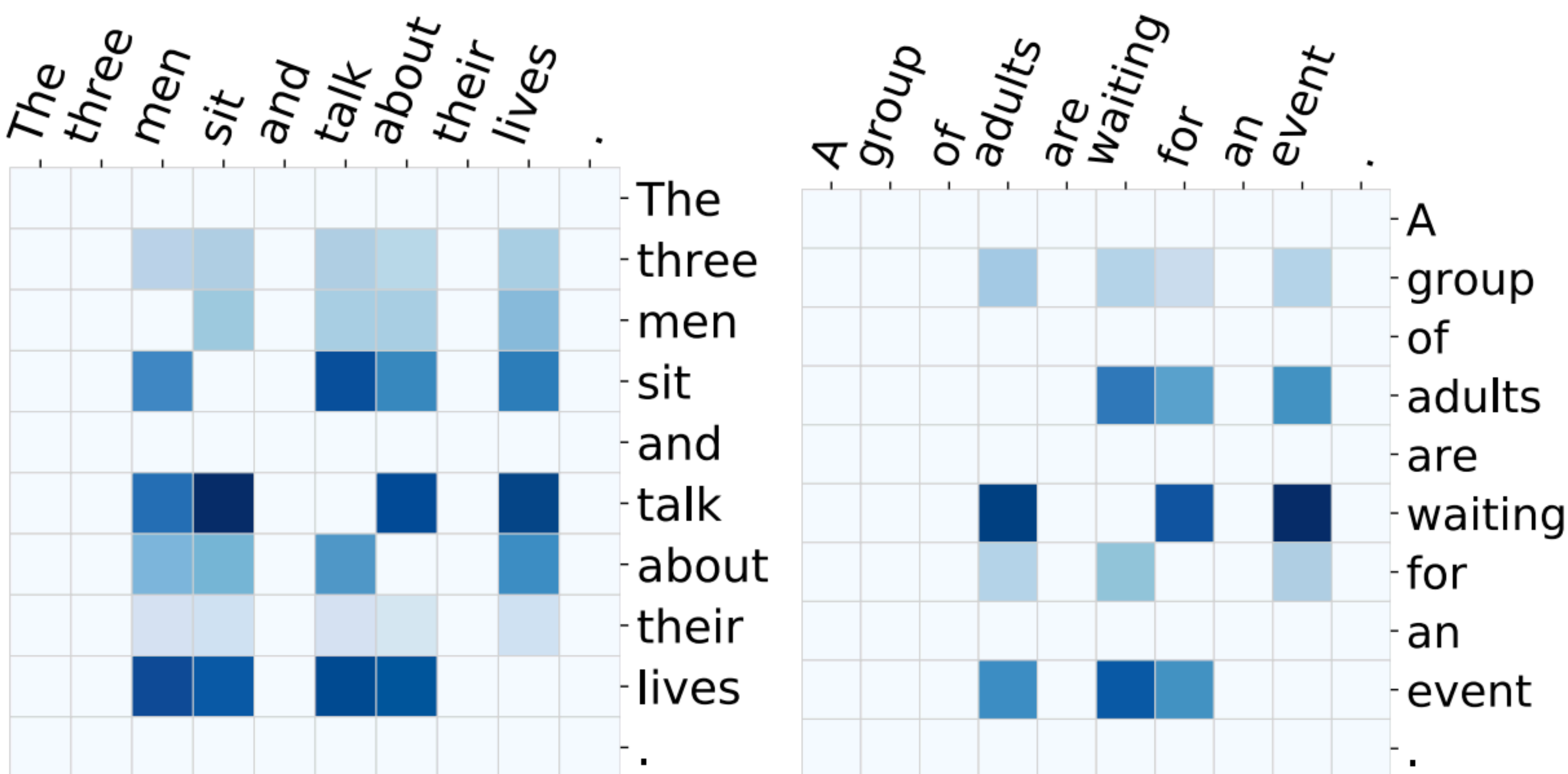
Natural Language Inference

Model	$\|\theta\|$	**T(s)/epoch**	**Inference T(s)**	**Train Accuracy**	**Test Accuracy**
300D LSTM encoders [Bowman *et al.*, 2016]	3.0m			83.9	80.6
300D SPINN-PI encoders [Bowman *et al.*, 2016]	3.7m			89.2	83.2
600D Bi-LSTM encoders [Liu *et al.*, 2016]	2.0m			86.4	83.3
600D Bi-LSTM +intra-attention [Liu *et al.*, 2016]	2.8m			84.5	84.2
300D NSE encoders [Munkhdalai and Yu, 2017]	3.0m			86.2	84.6
600D Deep Gated Attn. [Chen *et al.*, 2017]	11.6m			90.5	85.5
600D Gumbel TreeLSTM encoders [Choi *et al.*, 2017b]	10m			93.1	86.0
600D Residual stacked encoders [Nie and Bansal, 2017]	29m			91.0	86.0
Bi-LSTM [Graves *et al.*, 2013]	2.9m	2080	9.2	90.4	85.0
Bi-GRU [Chung *et al.*, 2014]	2.5m	1728	9.3	91.9	84.9
Multi-window CNN [Kim, 2014]	1.4m	284	2.4	89.3	83.2
Hierarchical CNN [Gehring *et al.*, 2017]	3.4m	343	2.9	91.3	83.9
Multi-head [Vaswani *et al.*, 2017]	2.0m	345	3.0	89.6	84.2
DiSAN [Shen *et al.*, 2018]	2.4m	587	7.0	91.1	85.6
300D ReSAN	3.1m	622	5.5	92.6	**86.3**

Semantic Relatedness

Model	Pearson's r	Spearman's ρ	MSE
Meaning Factory[a]	.8268	.7721	.3224
ECNU[b]	.8414	/	/
DT-RNN[c]	.7923 (.0070)	.7319 (.0071)	.3822 (.0137)
SDT-RNN[c]	.7900 (.0042)	.7304 (.0042)	.3848 (.0042)
Cons. Tree-LSTM[d]	.8582 (.0038)	.7966 (.0053)	.2734 (.0108)
Dep. Tree-LSTM[d]	.8676 (.0030)	.8083 (.0042)	**.2532 (.0052)**
Bi-LSTM	.8473 (.0013)	.7913 (.0019)	.3276 (.0087)
Bi-GRU	.8572 (.0022)	.8026 (.0014)	.3079 (.0069)
Multi-window CNN	.8374 (.0021)	.7793 (.0028)	.3395 (.0086)
Hierarchical CNN	.8436 (.0014)	.7874 (.0022)	.3162 (.0058)
Multi-head	.8521 (.0013)	.7942 (.0050)	.3258 (.0149)
DiSAN	.8695 (.0012)	.8139 (.0012)	.2879 (.0036)
ReSAN	**.8720 (.0014)**	**.8163 (.0018)**	*.2623 (.0053)*

Visualization



(a) Sentence 1 (b) Sentence 2

Thanks & QA