Recent Progress in RNN and NLP

Tohoku University
Inui and Okazaki Lab.
Sosuke Kobayashi
小林 颯介

Note

- Revised presentation slides from 2016/6/22 NLP-DL MTG@Preferred Networks and 2016/6/30 Inui and Okazaki Lab. Talk
- Overview of basic progress in RNN from late 2014
  - *Attention* is not included.
    c.f. http://www.slideshare.net/yutakikuchi927/deep-learning-nlp-attention
  - Not published arXiv papers are marked with "  "
  - Reference:
    https://docs.google.com/document/d/1nmkidNi_MsRPbB65kHsmyMfGqmaQ0r5dW518J8k_aeI/edit?usp=sharing
    (https://goo.gl/kE6GCM)

Agenda

- Basic RNN
- RNN's Unit
- Benchmarking various RNNs
- Connections in RNNs
- RNN and Tree
- Regularizations and Tricks for RNN's Learning
- Decoding

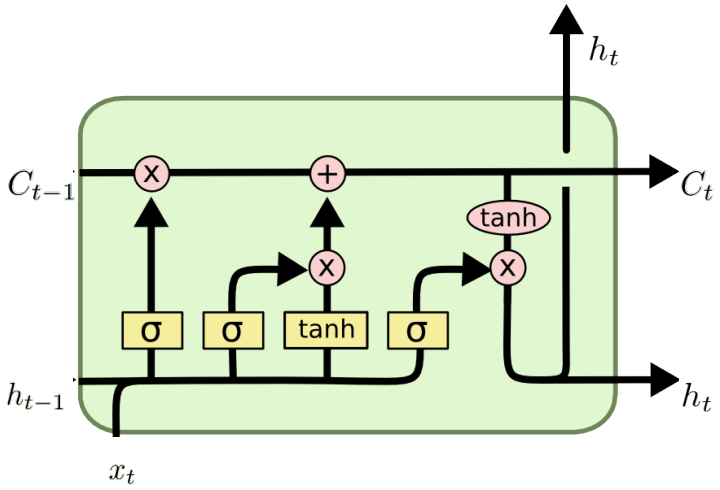
1. Unit and Benchmark

- Benchmarking RNN's various units
  - Variants of LSTM or GRU
  - Examinations of gates in LSTM
  - Initialization trick of LSTM
  - High performance by simple units

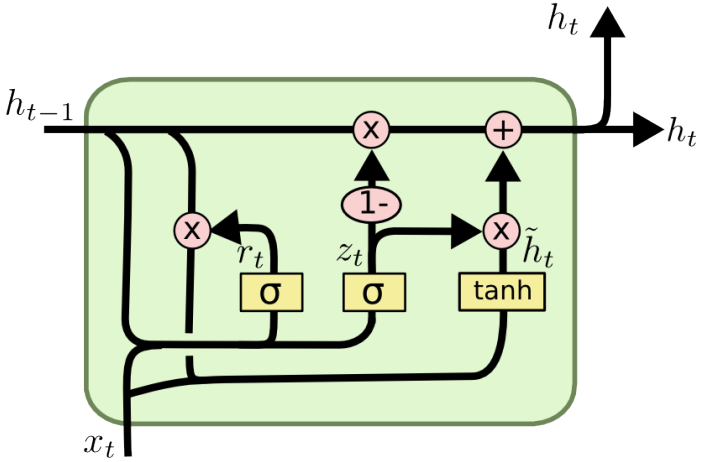
- Visualization and analysis

LSTM and GRU

- LSTM [Hochreiter&Schmidhuber97]



- GRU [Cho+14]



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

(Biases are omitted.)

Discovered Units

[Jozefowicz+15]

- Search better unit structures from LSTM and GRU by mutating computation graphs
  - Arith.: Calculation with noise tokens

3e36d9-h1h39f94eeh43keg3c= -13991064,

  - XML: Character-based prediction of XML tags
  - PTB: Language modeling

Arch.	Arith.	XML	PTB
Tanh	0.29493	0.32050	0.08782
LSTM	0.89228	0.42470	0.08912
LSTM-f	0.29292	0.23356	0.08808
LSTM-i	0.75109	0.41371	0.08662
LSTM-o	0.86747	0.42117	0.08933
LSTM-b	0.90163	0.44434	0.08952
GRU	0.89565	0.45963	0.09069
MUT1	**0.92135**	**0.47483**	0.08968
MUT2	0.89735	**0.47324**	0.09036
MUT3	0.90728	0.46478	**0.09161**

Discovered Units

[Jozefowicz+15]

- Better units are similar to GRU
- Due to bias from search algorighm?

GRU:

$$
\begin{aligned}
r_t &= \mathrm{sigm}\left(W_{\mathrm{xr}}x_t + W_{\mathrm{hr}}h_{t-1} + b_{\mathrm{r}}\right) \\
z_t &= \mathrm{sigm}(W_{\mathrm{xz}}x_t + W_{\mathrm{hz}}h_{t-1} + b_{\mathrm{z}}) \\
\tilde{h}_t &= \tanh(W_{\mathrm{xh}}x_t + W_{\mathrm{hh}}(r_t \odot h_{t-1}) + b_{\mathrm{h}}) \\
h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t
\end{aligned}
$$

- MUT1: Update gate is controlled by only x (not h). It looks reasonable for Arith.

MUT1:

h_tをh_(t_1)にずらしてください

$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}}x_t + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(W_{\mathrm{xr}}x_t + W_{\mathrm{hr}}h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + \tanh(x_t) + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

MUT2:

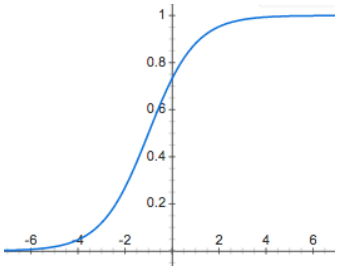
$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}}x_t + W_{\mathrm{hz}}h_t + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(x_t + W_{\mathrm{hr}}h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + W_{xh}x_t + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

MUT3:

$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}}x_t + W_{\mathrm{hz}}\tanh(h_t) + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(W_{\mathrm{xr}}x_t + W_{\mathrm{hr}}h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + W_{xh}x_t + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

Examination of LSTM [Jozefowicz+15]

- Remove LSTM's input, forget, output gates?
- LSTM's forget gate's bias is initialized to +1?
  ≒ Keeping 73% cell's values initially?



  - Initializing forget gate with positive bias is good
    ([Gers+2000] also said so.)
  - Dropout improves LSTM, not GRU, in language modeling
  - Gates' importance are f >> i > o.

Examination of LSTM [Greff+15]

- “LSTM: A Search Space Odyssey.” Cool title.
- Good examinations on LSTM
  - Gates, peephole, tanh before output,
    forget gate = 1 – input gate (like GRU)
  - Full gate recurrence; gates are also controlled by gates’ values at previous step
- Peephole is not important,
  forget gate is important,
  f=1-i is good and can save the # of parameters
- You are recommended to use a common LSTM

Other Devised Units

- Structurally Constrained Recurrent Network (SCRN) [Mikolov+15]
  - RNN with a simple cell by weighted sum

Q is diagonal

$$
\begin{aligned}
s_t &= (I - Q)Bx_t + Qs_{t-1}, \\
h_t &= \sigma\left(Ps_t + Ax_t + Rh_{t-1}\right), \\
y_t &= f\left(Uh_t + Vs_t\right)
\end{aligned}
$$

- IRNN [Le+15]
  - Simple RNN with recurrent matrix initialized with identity matrix and ReLU instead of tanh

$$h_t = \sigma\left(Ax_t + Rh_{t-1}\right)$$

- Effects of diagonal and orthogonal matrix in RNN [Henaff+16]

Other GRU-like Units

- Simple Gated Unit; SGU [Gao+, 16]

$$\vec{x}_g = W_{xh}\vec{x}_t + \vec{b}_g$$
$$\vec{z}_g = \sigma_1(W_{zxh}(\vec{x}_g \cdot \vec{h}_{t-1}))$$
$$\vec{z}_{out} = \sigma_2(\vec{z}_g \cdot \vec{h}_{t-1})$$
$$\vec{z}_t = \sigma_3(W_{xz}\vec{x}_t + \vec{b}_z + W_{hz}\vec{h}_{t-1})$$
$$\vec{h}_t = (1 - \vec{z}_t)\vec{h}_{t-1} + \vec{z}_t \cdot \vec{z}_{out}$$

- Deep SGU; DSGU [Gao+, 16]

$$\vec{x}_g = W_{xh}\vec{x}_t + \vec{b}_g$$
$$\vec{z}_g = \sigma_1(W_{zxh}(\vec{x}_g \cdot \vec{h}_{t-1}))$$
$$\vec{z}_{out} = \sigma_2(W_{go}(\vec{z}_g \cdot \vec{h}_{t-1}))$$
$$\vec{z} = \sigma_3(W_{xz}\vec{x}_t + \vec{b}_z + W_{hz}\vec{h}_{t-1})$$
$$\vec{h}_t = (1 - \vec{z}_t)\vec{h}_{t-1} + \vec{z}_t \cdot \vec{z}_{out}$$

- Minimal Gated Unit; MGU [Zhou+, 16]

GRU (Gated Recurrent Unit)

$$\boldsymbol{z}_t = \sigma\left(W_z\left[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t\right] + \boldsymbol{b}_z\right),$$
$$\boldsymbol{r}_t = \sigma\left(W_r\left[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t\right] + \boldsymbol{b}_r\right),$$
$$\tilde{\boldsymbol{h}}_t = \tanh\left(W_h\left[\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}, \boldsymbol{x}_t\right] + \boldsymbol{b}_h\right),$$
$$\boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t .$$

MGU (Minimal Gated Unit, the proposed me

$$\boldsymbol{f}_t = \sigma\left(W_f\left[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t\right] + \boldsymbol{b}_f\right),$$
$$\tilde{\boldsymbol{h}}_t = \tanh\left(W_h\left[\boldsymbol{f}_t \odot \boldsymbol{h}_{t-1}, \boldsymbol{x}_t\right] + \boldsymbol{b}_h\right),$$
$$\boldsymbol{h}_t = (1 - \boldsymbol{f}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{f}_t \odot \tilde{\boldsymbol{h}}_t .$$

Multiplicative Integration

- Multiplicative Integration [Wu+16]
  - Improvements by using multiplication with addition in RNN, changing

    $$\phi(\mathbf{W}\boldsymbol{x} + \mathbf{U}\boldsymbol{z} + \mathbf{b})$$

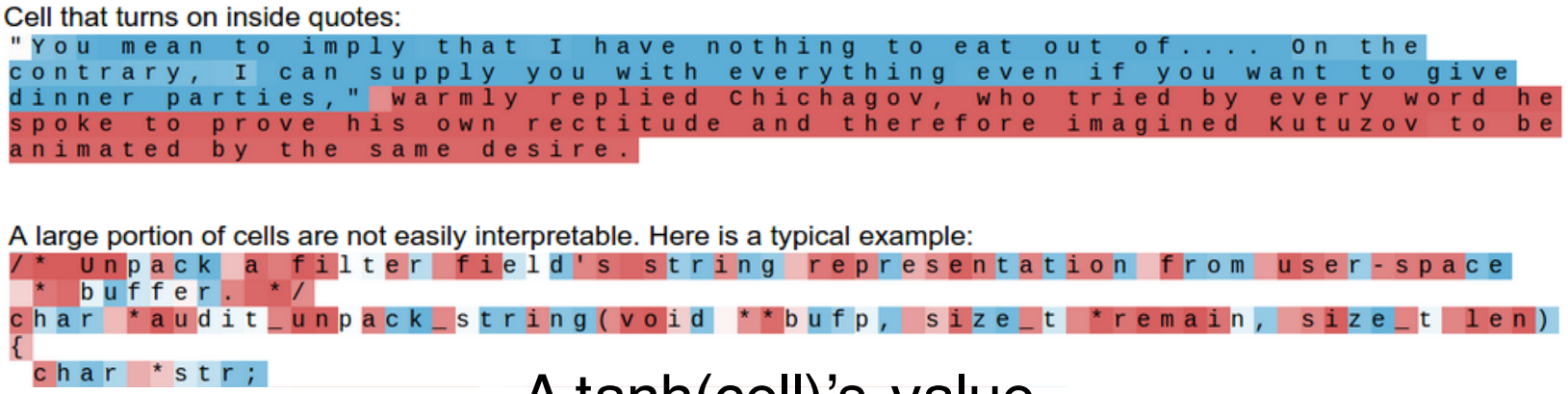
    into

    $$\phi(\boldsymbol{\alpha} \odot \mathbf{W}\boldsymbol{x} \odot \mathbf{U}\boldsymbol{z} + \boldsymbol{\beta}_1 \odot \mathbf{U}\boldsymbol{z} + \boldsymbol{\beta}_2 \odot \mathbf{W}\boldsymbol{x} + \boldsymbol{b})$$

  - Similarly in LSTM and GRU
  - Improve performances of many tasks
  - In the near future, this will get common ...?

Visualization [Karpathy+15]

- Visualization of character-based language model
  - A cell got a function of apostrophes' opening and closing
  - But other cells can not be interpretable

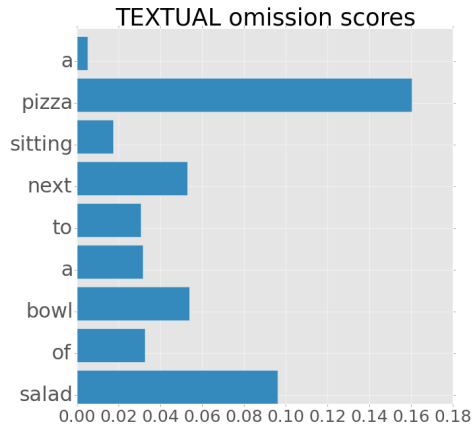


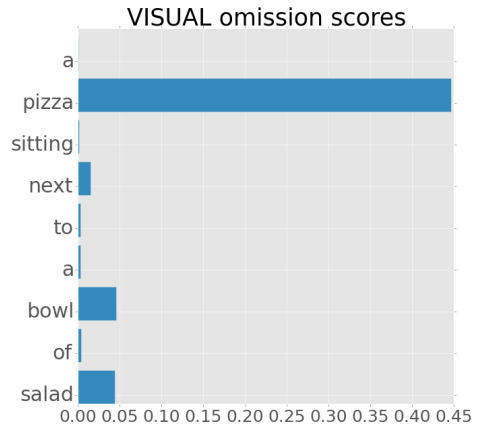
Word Ablation

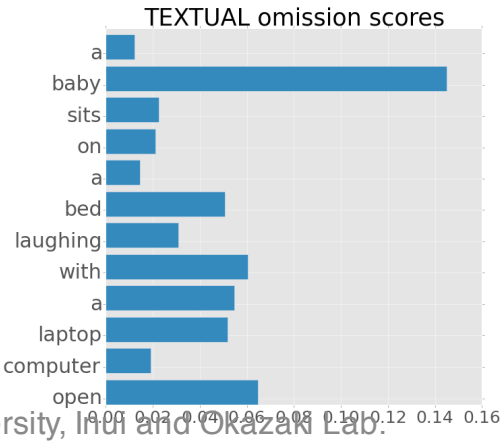
[Kádár+16]

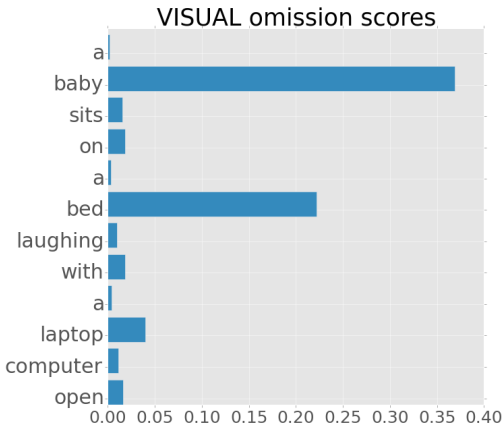
- Analyzing a GRU's output by omission score when encoding a image caption
- Model predicting image's vector (CNN output) focuses on nouns
- Language model focuses more evenly

$$\text{omission}(i, S) = 1 - \text{cosine}(\mathbf{h}_{\text{end}}(S), \mathbf{h}_{\text{end}}(S_{\setminus i}))$$





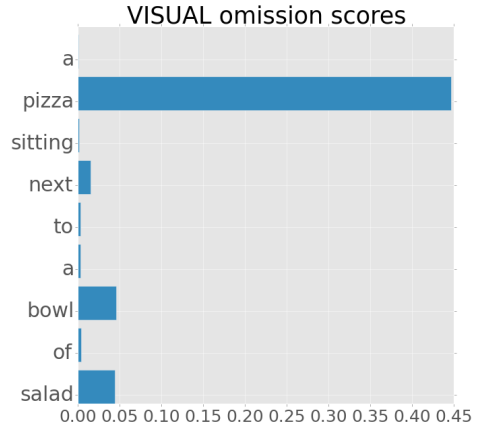


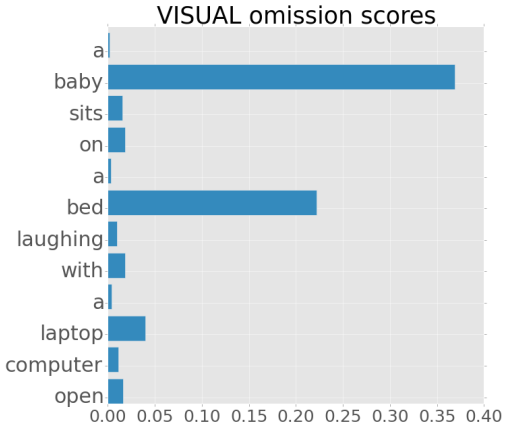


Word Ablation

[Kádár+16]

- Removing word ‘pizza’ removes a just ’pizza’ from the image (searched from dataset)

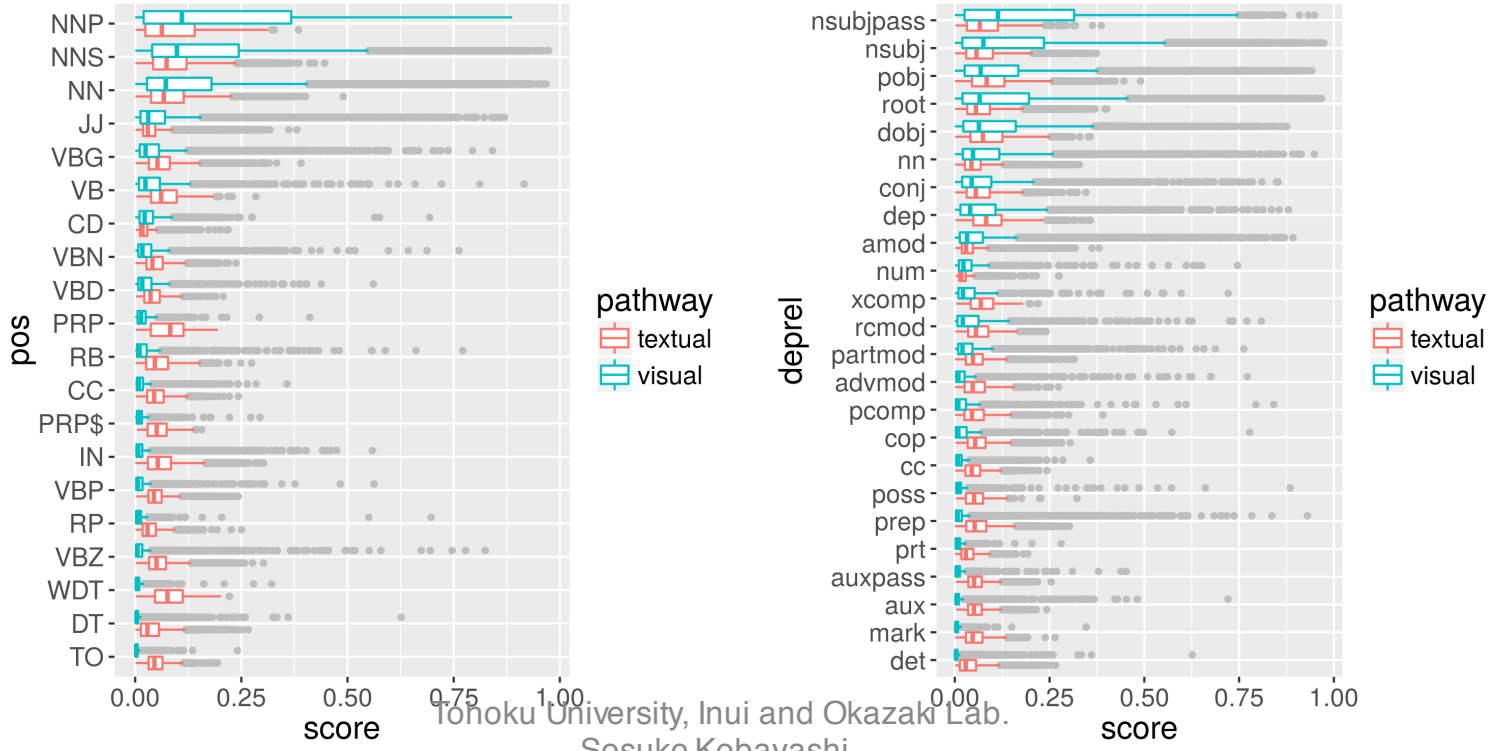




Word Ablation

[Kádár+16]

- Analyzing mean omission scores in dataset on pos-tag, model for image focuses on
  NN > JJ > VB, CD > ...

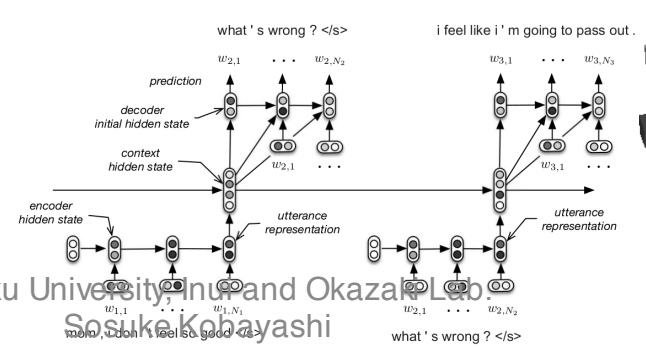


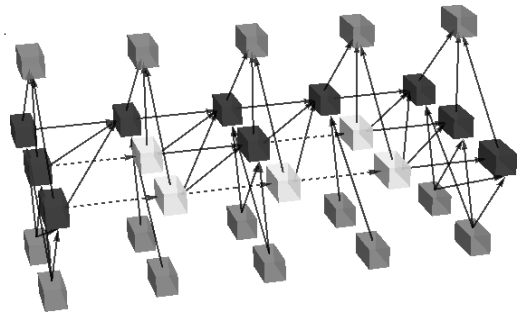
2. Connections and Trees

- Connections of RNNs
- Tree structure and RNN (LSTM)
- Tree-based Composition by Shift-reduce

Connections in Multi-RNNs

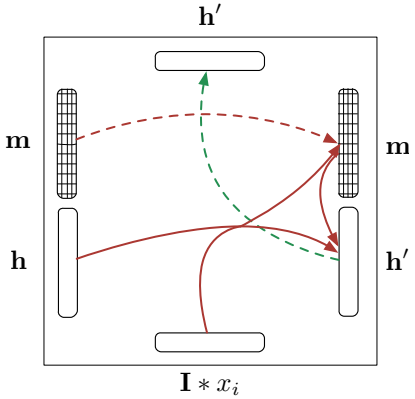
- Clockwork RNN. Combination of different RNNs with different step processing [Koutník+14, Liu+15, (Chung+16)]
- Gated Feedback RNN. Feedback outputs into lower layers with gate [Chung+15]
- Depth-Gated LSTM, Highway LSTM: Cell are connected to the upper layer‘s cell with gate [Yao+15, Chen+15]
- k-th layer’s input is from k-1th’s input and output [Zhou+16]
- Hierarchical RNN [Serban+2015]



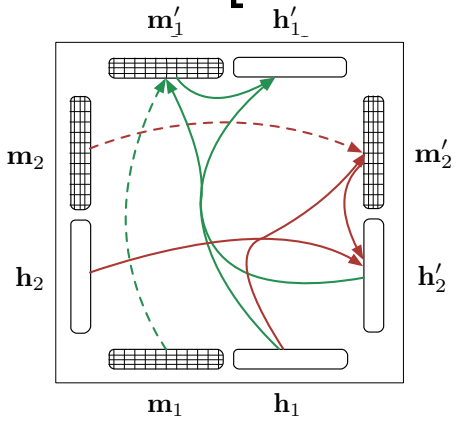


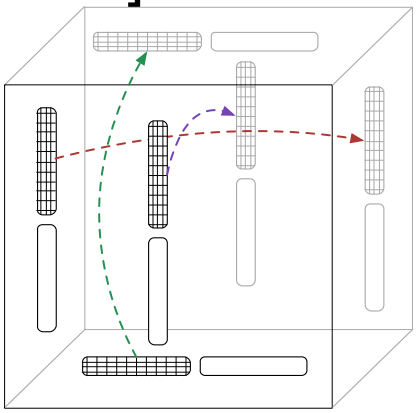
Connections in Multi-RNNs

- Grid LSTM. Each axis has each LSTM for multi-dimensional applications [Kalchbrenner+15]

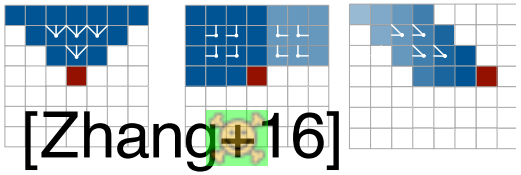


Standard LSTM block

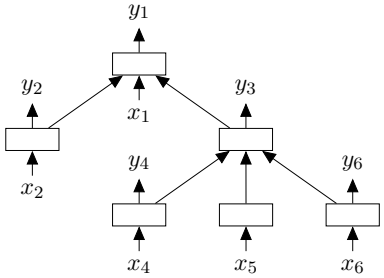




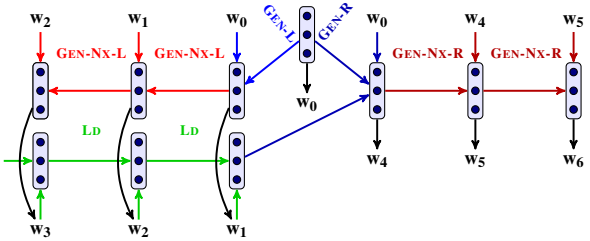
- RNN for DAG, (image) pixel [Shuai+15, Zhu+16, Oord+16]
- Structure complexity of RNN model [Zhang+16]



Tree-LSTM

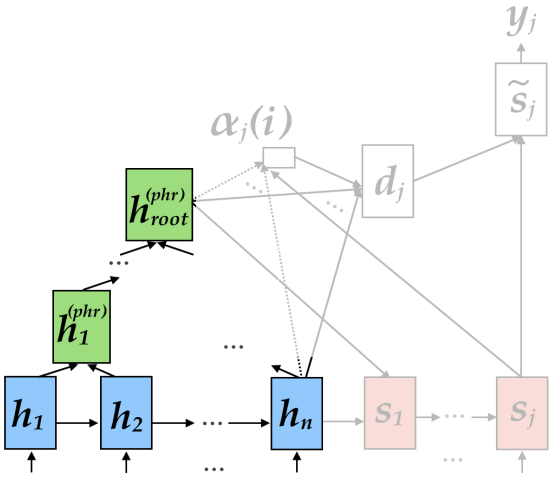


- Tree-LSTM [Tai+15] Apply LSTM to follow directed edges (child to parent) of tree structure. Most cited “Tree-LSTM”
  - S-LSTM [Zhu+15] Add peephole and remove input x
  - LSTM-RecursiveNN [Le+15] Control forget and input gates with untied matrices of each cell and ouput (h). Input gate is applied before tanh.
- Top-down TreeLSTM [Zhang+16] Sentence generation from root of dependency tree



Combination of Tree and Seq

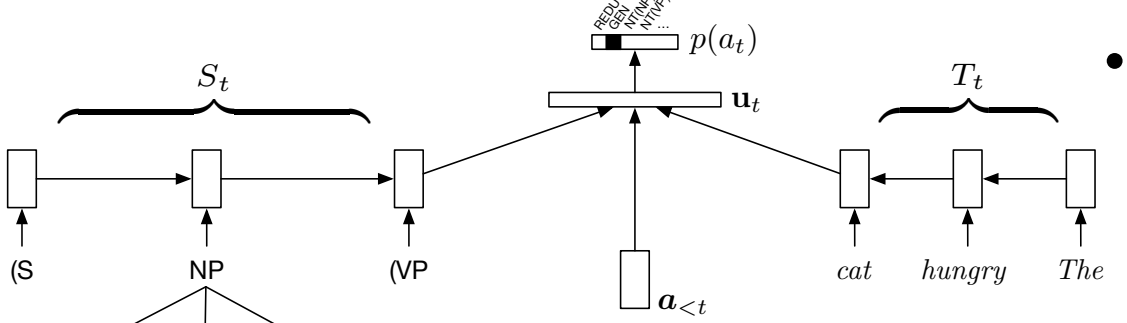
- Tree-based and Sequential Encoder for Attention. [Eriguchi+16]
  - Tree-LSTM composition with leaf nodes output from seq-LSTM
  - “The cutest approach!”, Kyunghyun Cho said at SedMT, NAACL16.
  - Undercoated seq-LSTM makes nodes more context-aware and less ambiguous. +[Bowman+16]



Recurrent Neural Network Grammars

[Dyer+16]

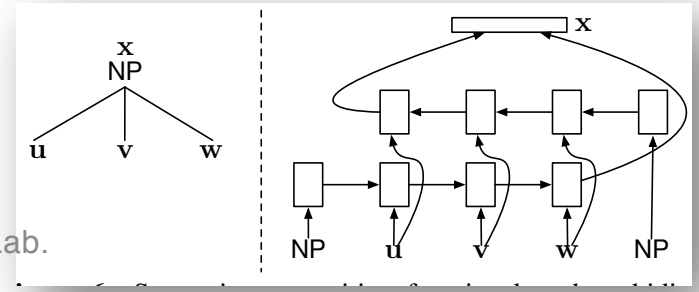
- Generation by sequential actions from {GEN(word), REDUCE, NT(non-terminal symbol)}. Features for action decisions are LSTM's outputs of (1) terminals (2) stack (3) action history.



	Stack *The hungry cat*	Terminals	Action
0			NT(S)
1	(S		NT(NP)
2	(S | (NP		GEN(*The*)
3	(S | (NP | *The*	*The*	GEN(*hungry*)
4	(S | (NP | *The* | *hungry*	*The* | *hungry*	GEN(*cat*)
5	(S | (NP | *The* | *hungry* | *cat*	*The* | *hungry* | *cat*	REDUCE
6	(S | (NP *The hungry cat*)	*The* | *hungry* | *cat*	NT(VP)
7	(S | (NP *The hungry cat*) | (VP	*The* | *hungry* | *cat*	GEN(*meows*)
8	(S | (NP *The hungry cat*) | (VP *meows*	*The* | *hungry* | *cat* | *meows*	REDUCE
9	(S | (NP *The hungry cat*) | (VP *meows*)	*The* | *hungry* | *cat* | *meows*	GEN(.)
10	(S | (NP *The hungry cat*) | (VP *meows*) | .	*The* | *hungry* | *cat* | *meows* | .	REDUCE
11	(S (NP *The hungry cat*) (VP *meows*) .)	*The* | *hungry* | *cat* | *meows* | .	

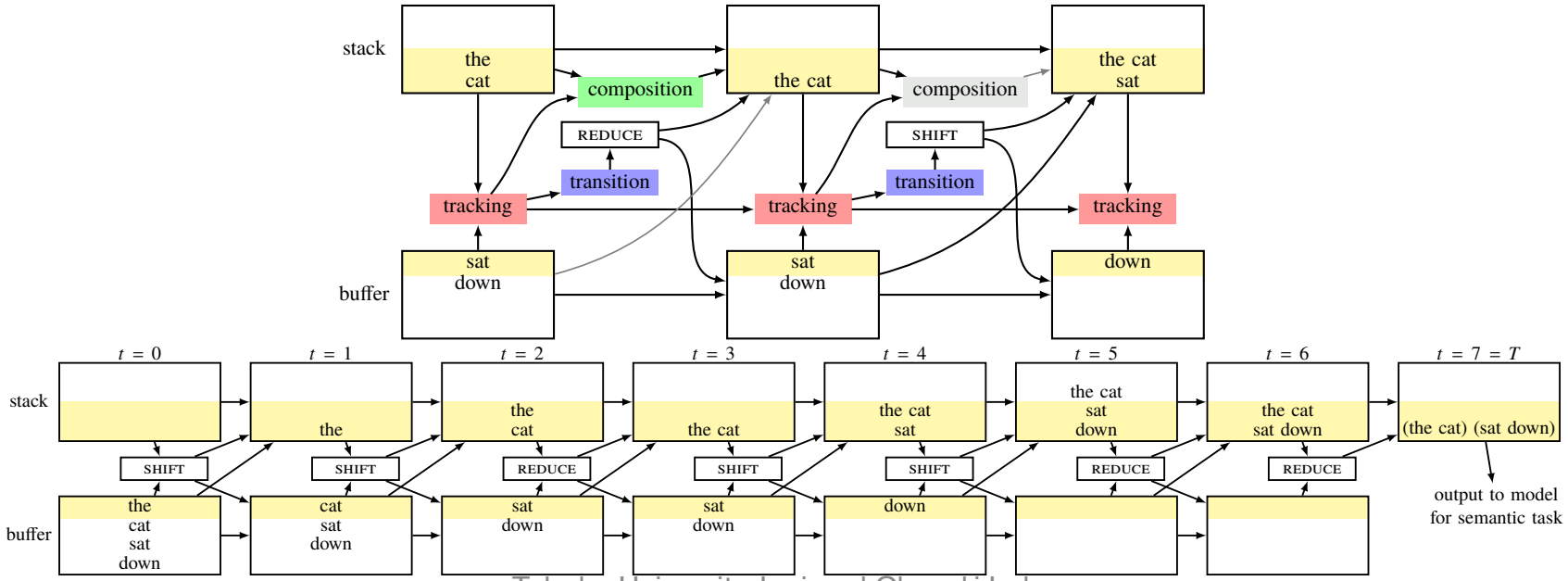
- REDUCE action weaves a new chunk vector by bi-LSTM and re-stacks it.

"*NP→the→hungry→cat*"



SPINN Stack-augmented Parser-Interpreter Neural Network [Bowman+16]

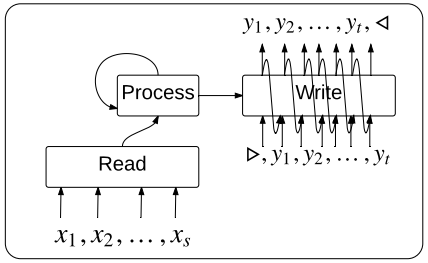
- Joint learning of shift-reduce parsing and sentence-level classification with shift-reduce-based tree-LSTM composition. When REDUCE the top 2 chunks are composed by tree-LSTM.
- Speedy tree composition (like Recurrent NN).



(b) The fully unrolled SPINN for *the cat sat down*, with neural network layers omitted for clarity.

(Encoders without C/RNN)

- Repeated attention with LSTM captures input vectors as a set [Vinyals+15]



$$
\begin{aligned}
q_t &= LSTM(q^*_{t-1}) \\
e_{i,t} &= f(m_i, q_t) \\
a_{i,t} &= \frac{\exp(e_{i,t})}{\sum_j \exp(e_{j,t})} \\
r_t &= \sum_i a_{i,t} m_i \\
q^*_t &= [q_t\ r_t]
\end{aligned}
$$

- (End-to-end) Memory Networks [Sukhbaatar+15]

Sentence encoding by weighted sum.

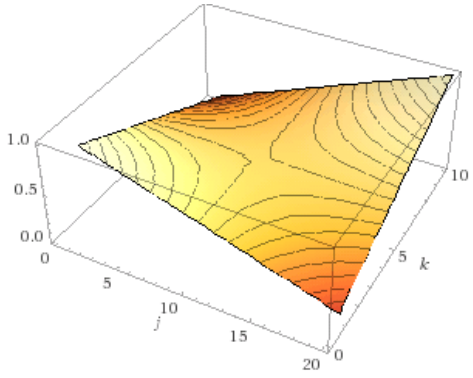
Earlier word('s vector) has larger weights at smaller dims.

Later word('s vector) has larger weights at larger dims.

$$m_i = \sum_j l_j \cdot Ax_{ij}$$

$$l_{kj} = (1 - j/J) - (k/d)(1 - 2j/J)$$

- e.g., When sentence length d is 10 and vector dimension J is 20,
  value of 1st vec at 1st dim: (1-1/20)-(1/10)(1-2*1/20) = 0.86
  value of 1st vec at 20th dim: (1-20/20)-(1/10)(1-2*20/20) = 0.1
  value of 10th vec at 1st dim: (1-1/20)-(10/10)(1-2*1/20) = 0.05
  value of 10th vec at 20th dim: (1-20/20)-(10/10)(1-2*20/20) = 1.0



3. Learning Tricks

- Regularizations
- Dropout in RNN
- Batch Normalization in RNN
- Other Regularizations
- Multi-task learning and pre-training of encoder(-decoder)

Dropout

- Dropout [Hinton+12, Srivastava+14]
  Drop nodes by probability p and multiply 1/(1-p).
  - RNN (mainly LSTM and GRU) need some tricks
  - At upward (inter-layer) connections [Zaremba+14]
  - At update terms [Semeniuta+16]

LSTM
$$\mathbf{c}_t = \mathbf{f}_t * \mathbf{c}_{t-1} + \mathbf{i}_t * d(\mathbf{g}_t)$$

GRU
$$\mathbf{h}_t = (1 - \mathbf{z}_t) * \mathbf{h}_{t-1} + \mathbf{z}_t * d(\mathbf{g}_t)$$

  - Use one consistent dropout mask in one seq. (Effect looks obscure now) [Semeniuta+16, Gal15]
  - Zoneout. Stochastically preserve previous c and h [Krueger+16]
- Word dropout. Stochastically use zero/mean/<unk> vec as a word vec. [Iyyer+15, Dai&Le15, Dyer+15, Bowman+15]

Batch Normalization

- Batch Normalization [Ioffe+15] Normalization, scale and shift function at intermediate layer. Popular for CNN (DNN).
- RNN needs tricks [Cooijmans+16, Laurent+15]
  - Mean and Variance value for normalization are prepared at each time step
  - Don't insert at cell's recurrence
  - Initialize scale value lower (0.1). Prevent sigm/tanh saturation initially.

$$\begin{pmatrix} \tilde{\mathbf{f}}_t \\ \tilde{\mathbf{i}}_t \\ \tilde{\mathbf{o}}_t \\ \tilde{\mathbf{g}}_t \end{pmatrix} = \mathrm{BN}(\mathbf{W}_h \mathbf{h}_{t-1}; \gamma_h, \beta_h) + \mathrm{BN}(\mathbf{W}_x \mathbf{x}_t; \gamma_x, \beta_x) + \mathbf{b}$$

$$\mathbf{c}_t = \sigma(\tilde{\mathbf{f}}_t) \odot \mathbf{c}_{t-1} + \sigma(\tilde{\mathbf{i}}_t) \odot \tanh(\tilde{\mathbf{g}}_t)$$

$$\mathbf{h}_t = \sigma(\tilde{\mathbf{o}}_t) \odot \tanh(\mathrm{BN}(\mathbf{c}_t; \gamma_c, \beta_c))$$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i)$$

Regularization for Successiveness

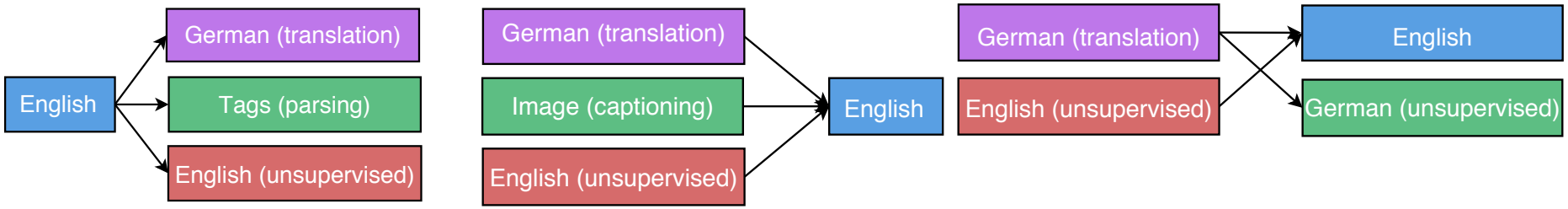
- Norm-stabilizer [Krueger+15]
  - Penalize the difference between the norms of hidden vectors at successive time steps

$$\beta \frac{1}{T} \sum_{t=1}^{T} (\|h_t\|_2 - \|h_{t-1}\|_2)^2$$

- Temporal Coherence Loss [Jonschkowski&Brock15]
  - Penalize the difference between the hidden vectors at successive time steps.

Multi-task Learning

- Auto-(sentence-)encoder and language modeling as pre-training for sentence classifications. (But, joint learning is not good.) [Dai&Le15]
- Multi-task learning for encoder-decoder.
  The coefficients of tasks' losses are so important.
  (Multi-language translation, parsing, image captioning, auto-encoder, skip-thought vectors) [Luong+15]



4. Decoding

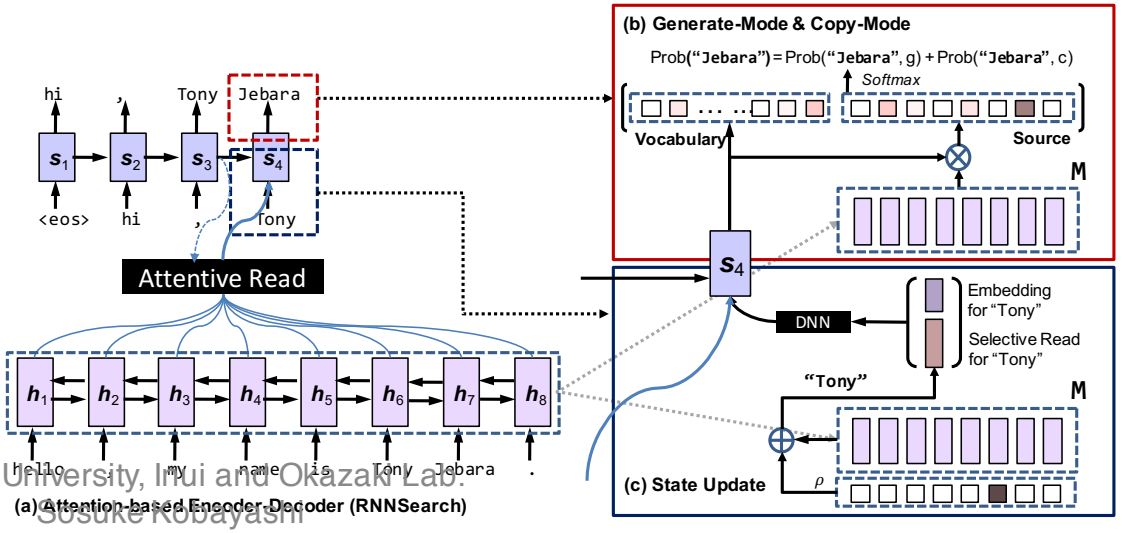
- Lighten Calculation of Softmax output
- Copy mechanism
- Character-based
- Global Optimization of Decoding

Lighten Softmax

- Softmax over large vocabulary (class) has large time and space computational complexity.
  Lighten (replace) it by
  - Sampled Softmax
  - Class-factored Softmax
  - Hierarchical Softmax
  - BlackOut
  - Noise Constrastive Estimation; NCE
  - Self-normalization
  - Negative Sampling

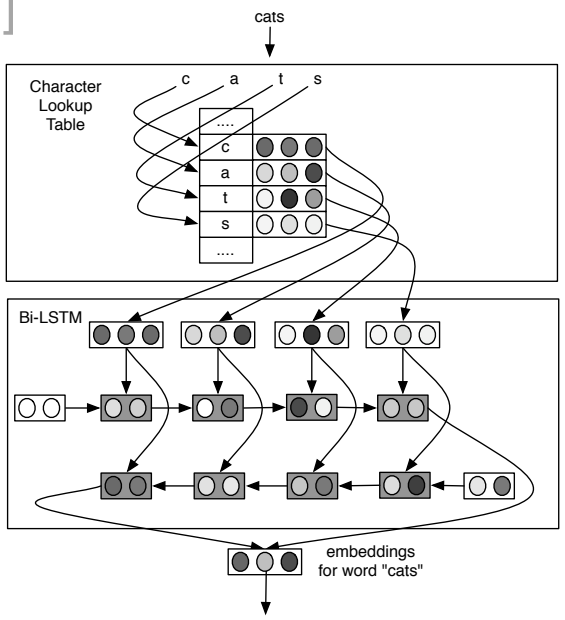
Copy Mechanism

- Copy function from source sentence
- [Gulcehre+16] calculates attention distribution over source sentence('s LSTM outputs).
  (Pointer Networks [Vinyals+15])
  Sigmoid gate-based weighted sum of common vocabulary output probability and copy vocabulary probability distribution.
- [Gu+16]
  Similar, but more complicated structure



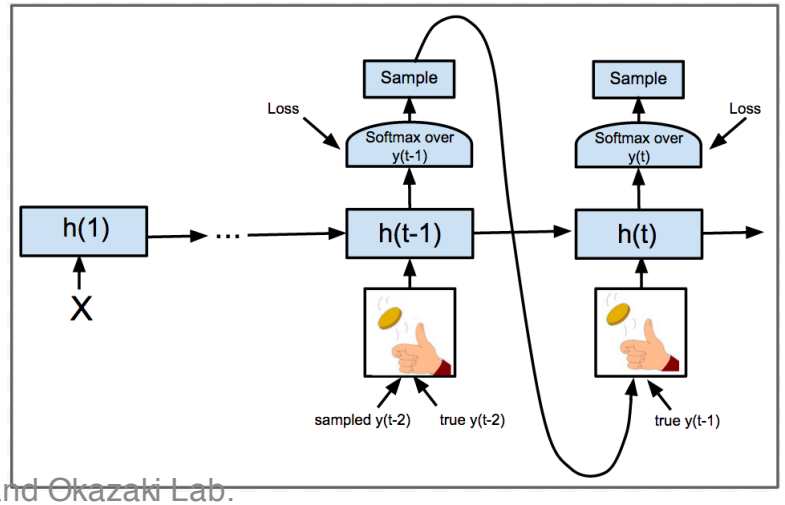
Character-based

- In/output chunk is a character, not a defined word
  - Language model, various task's input features, machine translation's decoding LM: [Sutskever+11, Graves13, Ling+15a, Kim+15], MT: [Chung+16, Ling+15b, Costa-jussa&Fonollosa16, Luong+16]
  - Combination of words and characters [Kang+11, Józefowicz+16, Miyamoto&Cho16]
  - (Not only RNN composition, but also CNN)
  - Good in terms of morphology and rare word problem



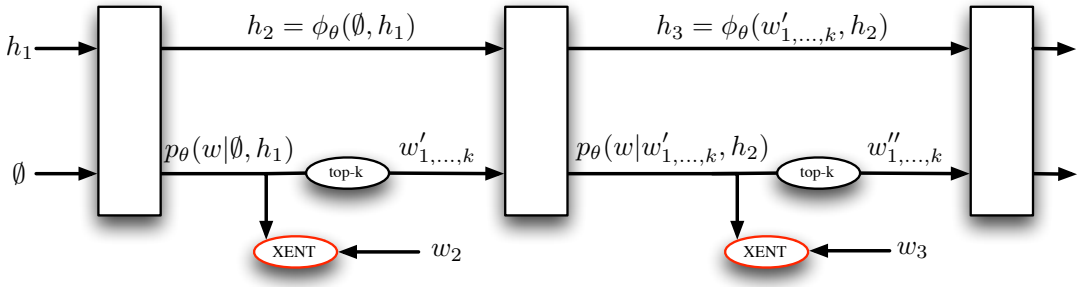
Global Decoding

- Reinventing the wheel (of non-NN research)?
  (Even so, these are useful and good next steps.)
- Use model's prediction as next input in training (while usually only true input is used) [Bengio+15]
  - Similar to DAgger [Daumé III16; Blog]
  - Use dynamic oracle [Daumé III16; Blog] [Ballesteros+16, Goldberg&Nivre13]



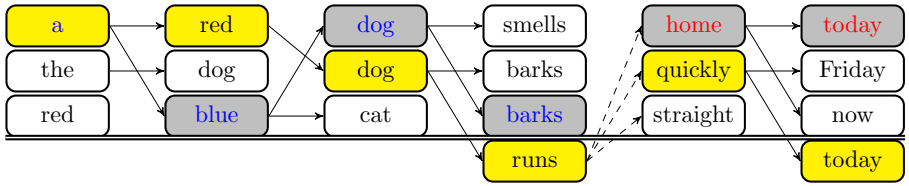
Global Decoding

- REINFORCE to optimize BLEU/ROUGE [Ranzato+15]



$$L_\theta = -\sum_{w_1^g,\ldots,w_T^g} p_\theta(w_1^g,\ldots,w_T^g) r(w_1^g,\ldots,w_T^g) = -\mathbb{E}_{[w_1^g,\ldots w_T^g]\sim p_\theta} r(w_1^g,\ldots,w_T^g),$$

- Minimum Risk Training [Shen+15, Ayana+16]
- Optimization for beam search [Wiseman&Rush16]



Summary

- Better RNN's units and connections are produced, however, their impacts are small now
  (compared to ones between "vanilla and LSTM" or "1-layer or multi-layer".)
- Analysis is more needed in general and each task
- Designing models with (reasonable) idea may good result, e.g., tree composition
- Regularization and learning tricks increased
- Other decoding training or inference algorithms are required