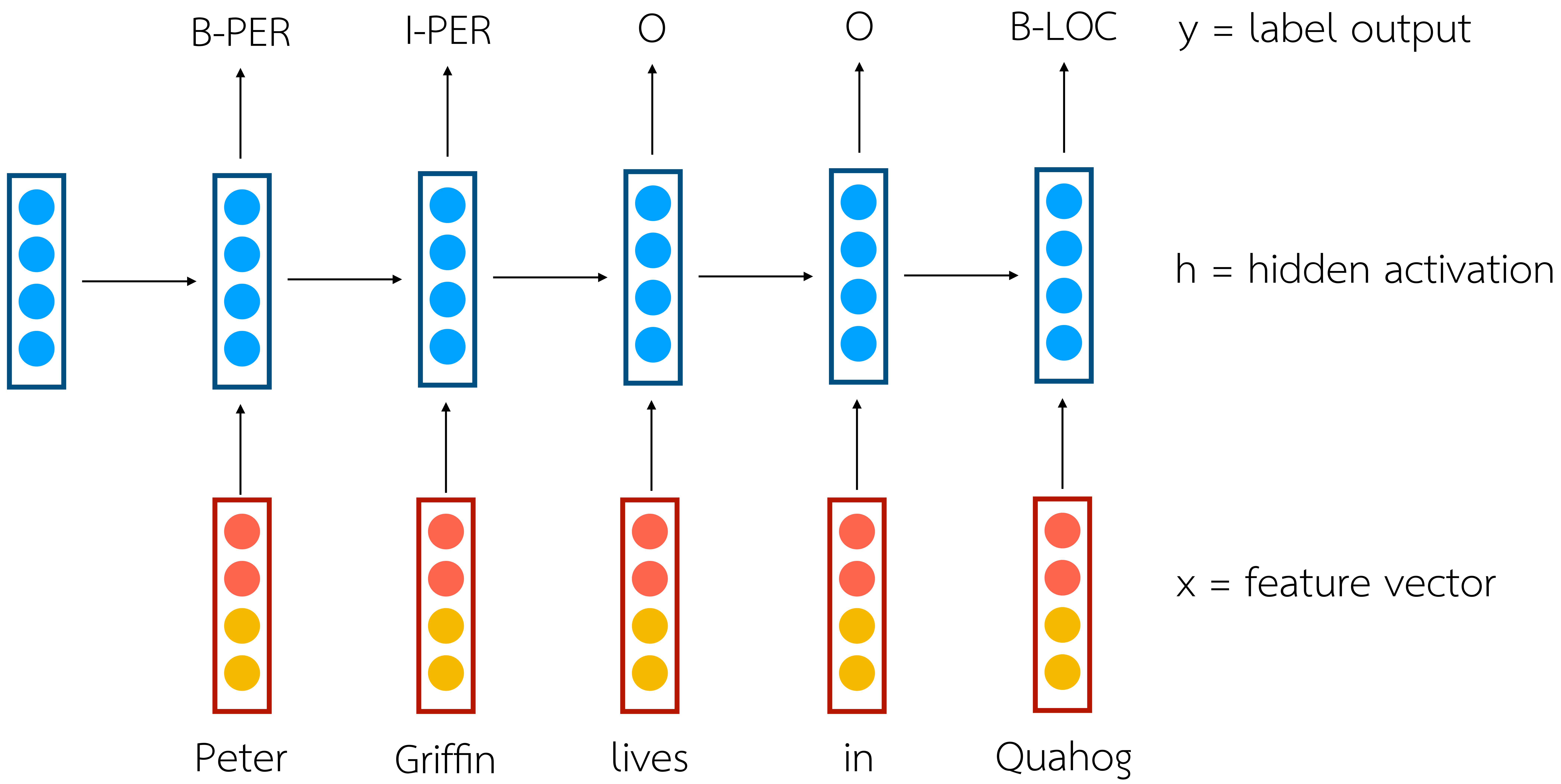
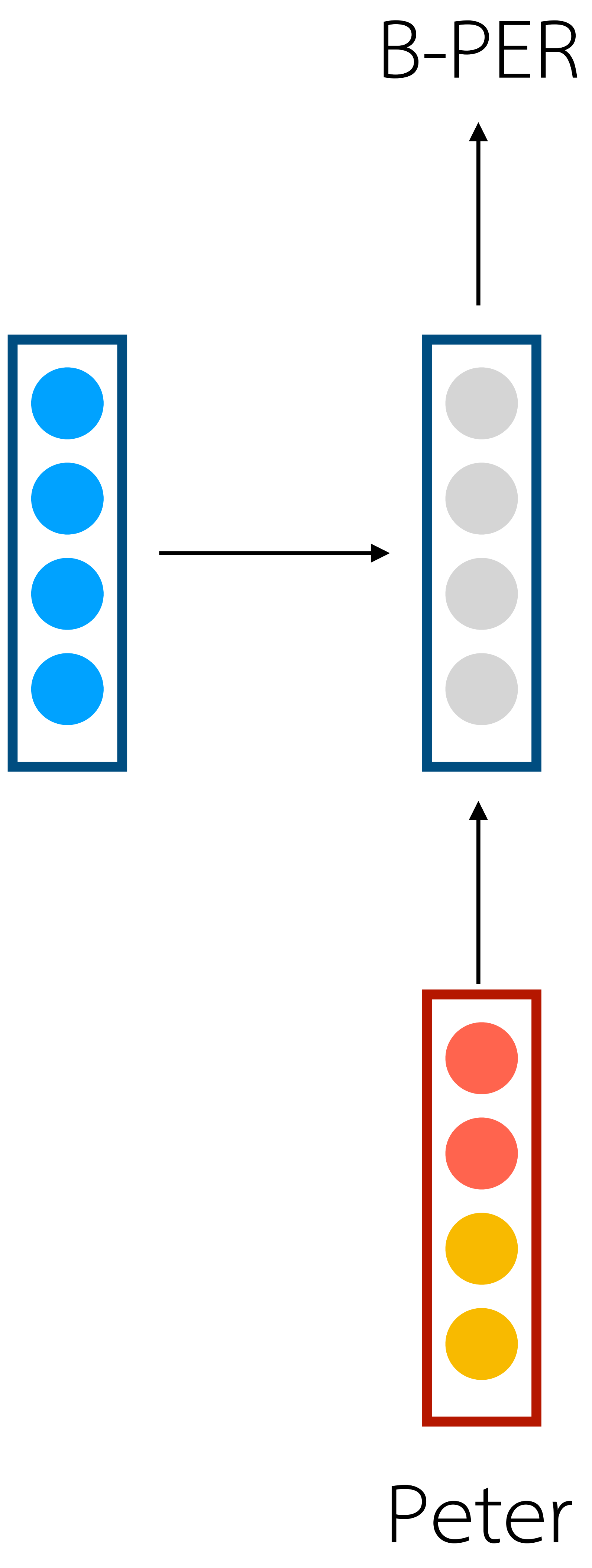
Recurrent Neural Network

Agenda

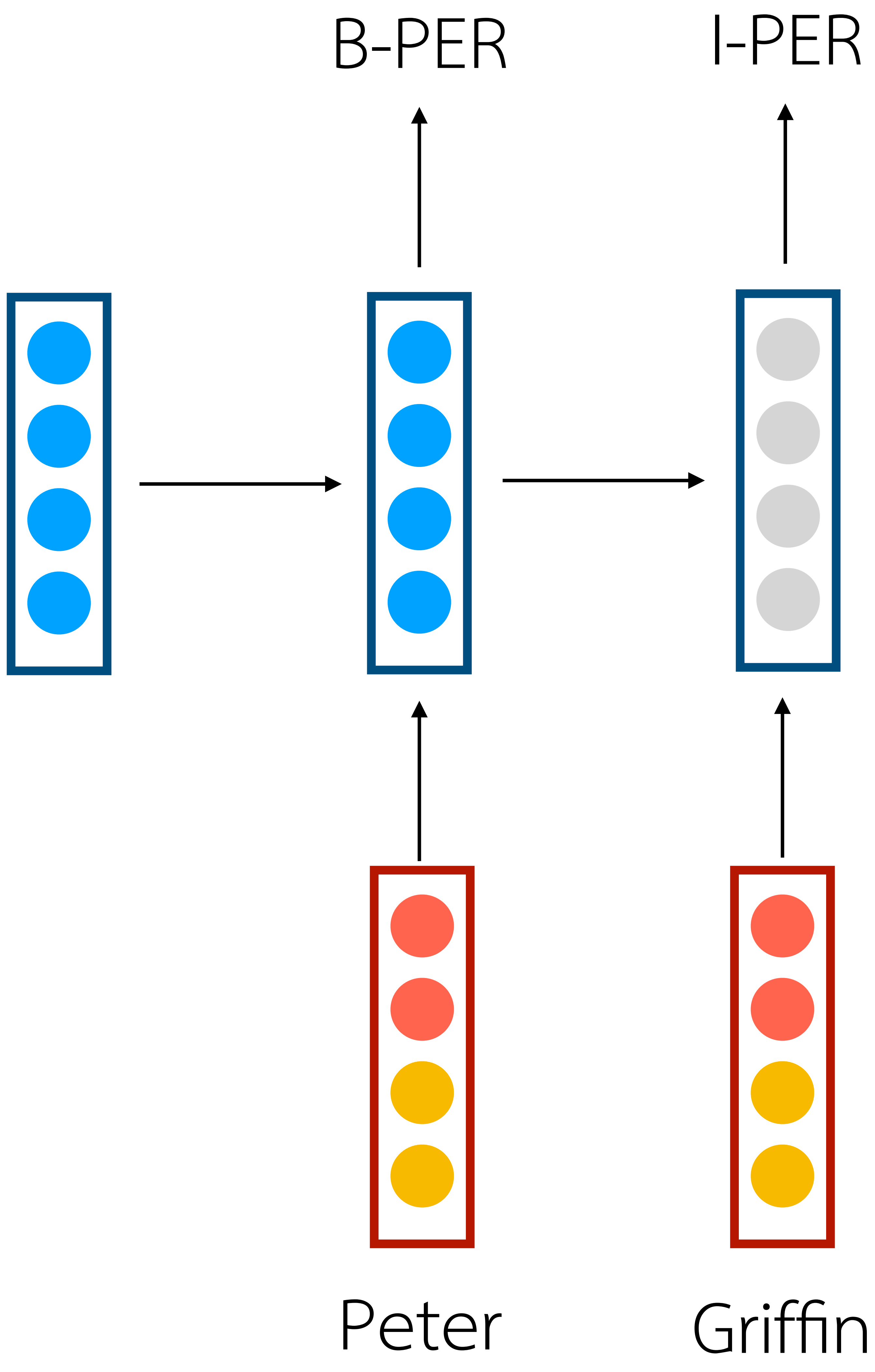
- Recurrent Neural Network ทำไมถึงเหมาะกับการทำ NER หรือ sequence tagging
- RNN มีวิธีการทำงานอย่างไร มี parameter อะไรบ้าง


B-PER
I-PER
O
O
B-LOC
y = label output
h = hidden activation
x = feature vector
Peter
Griffin
lives
in
Quahog



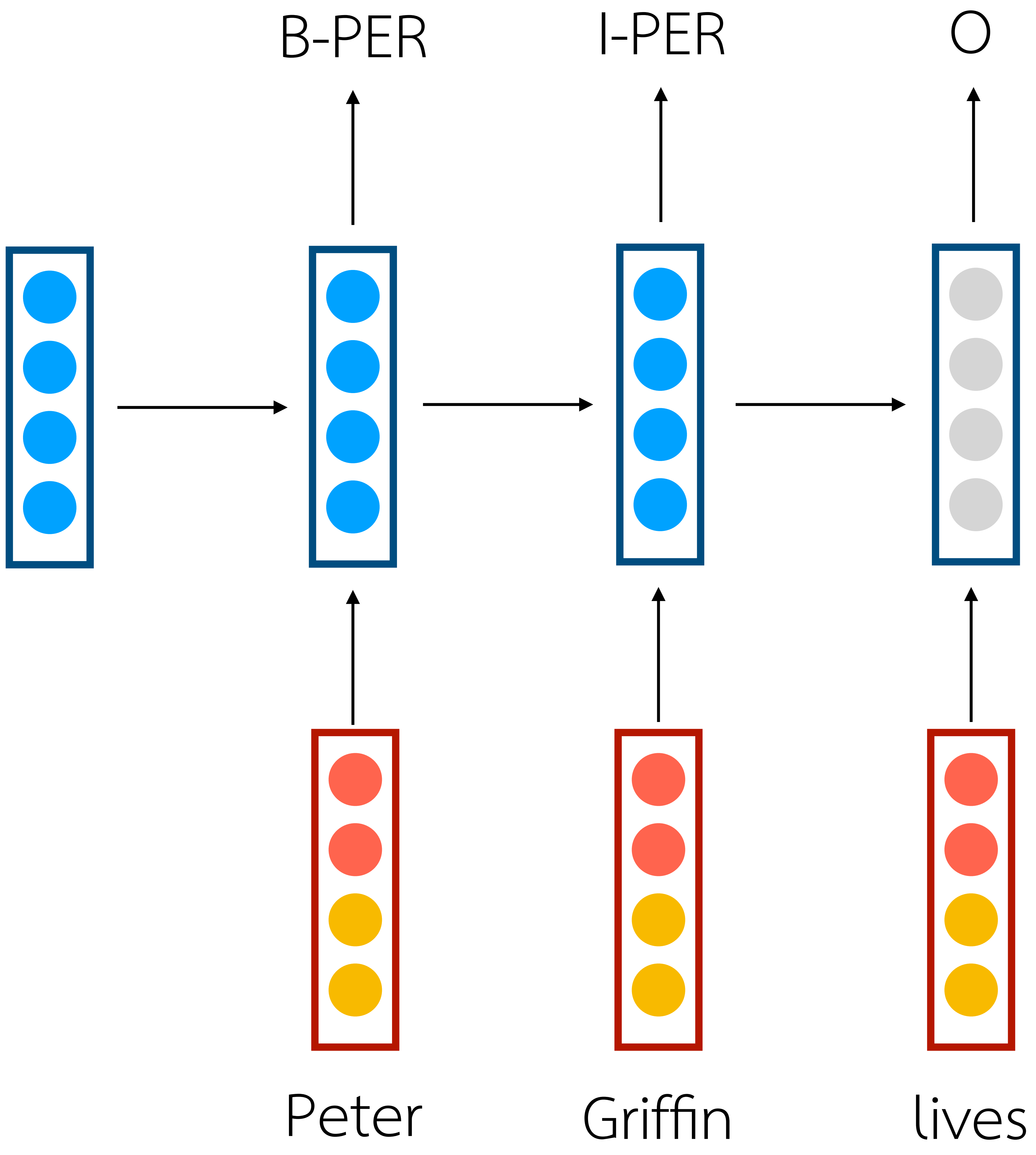
$$y_1 = \mathrm{softmax}(W_y\, h_1 + b_y)$$

$$h_1 = \tanh\left(W_h \, [x_1; h_0] + b_h \right)$$

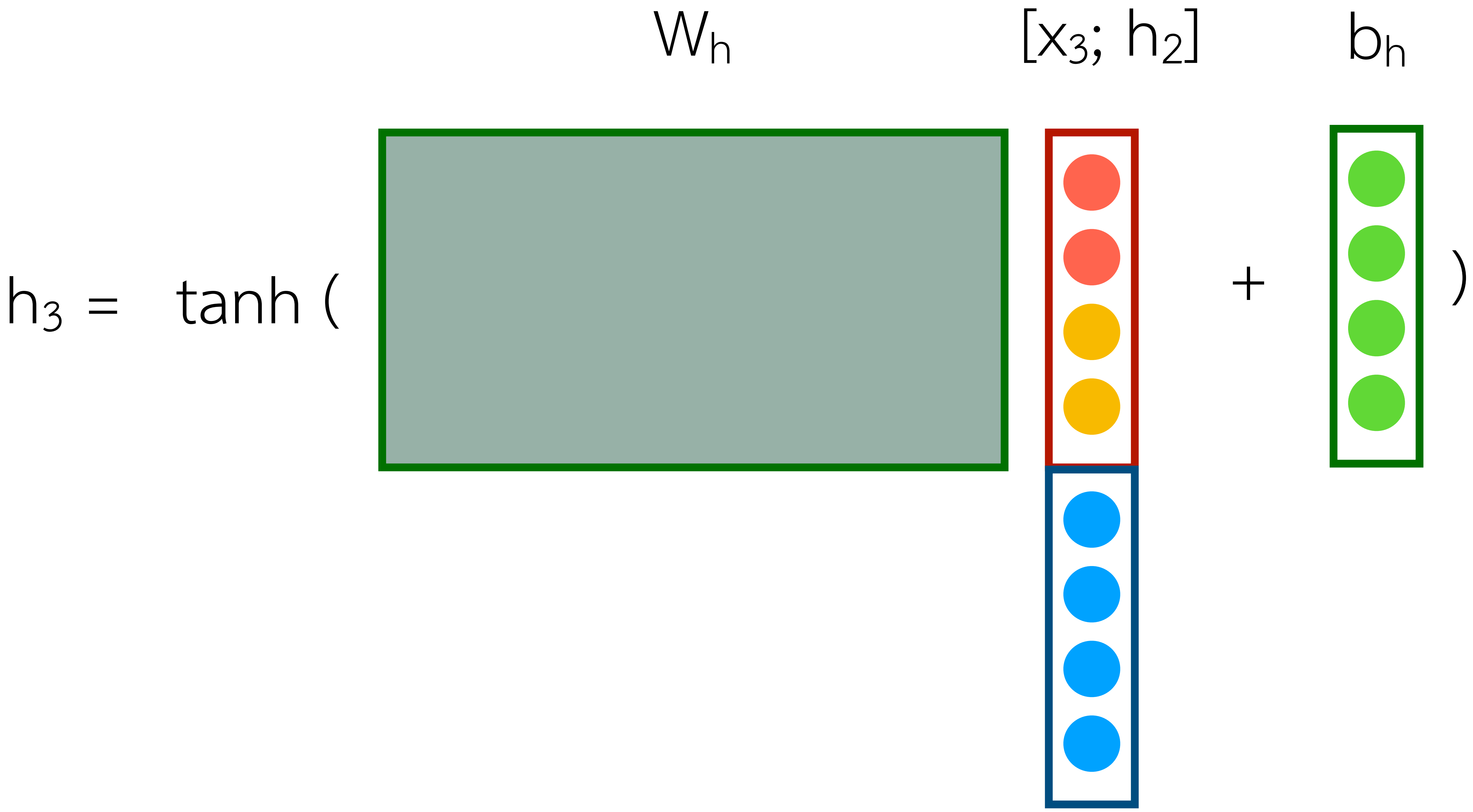


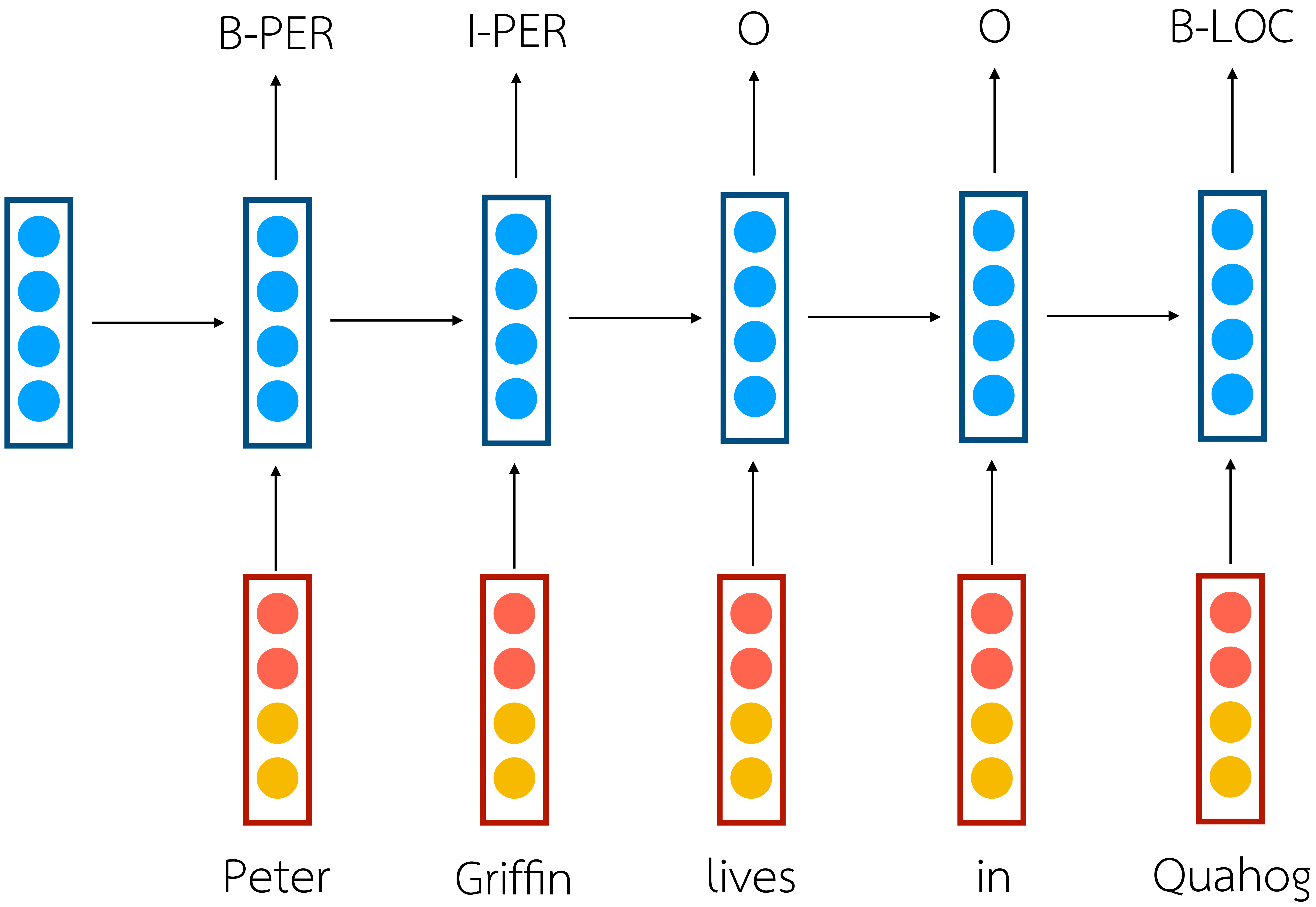
$$y_2 = \mathrm{softmax}(W_y\, h_2 + b_y)$$

$$h_2 = \tanh\left(W_h\, [x_2; h_1] + b_h \right)$$



$$y_3 = \mathrm{softmax}(W_y\, h_3 + b_y)$$

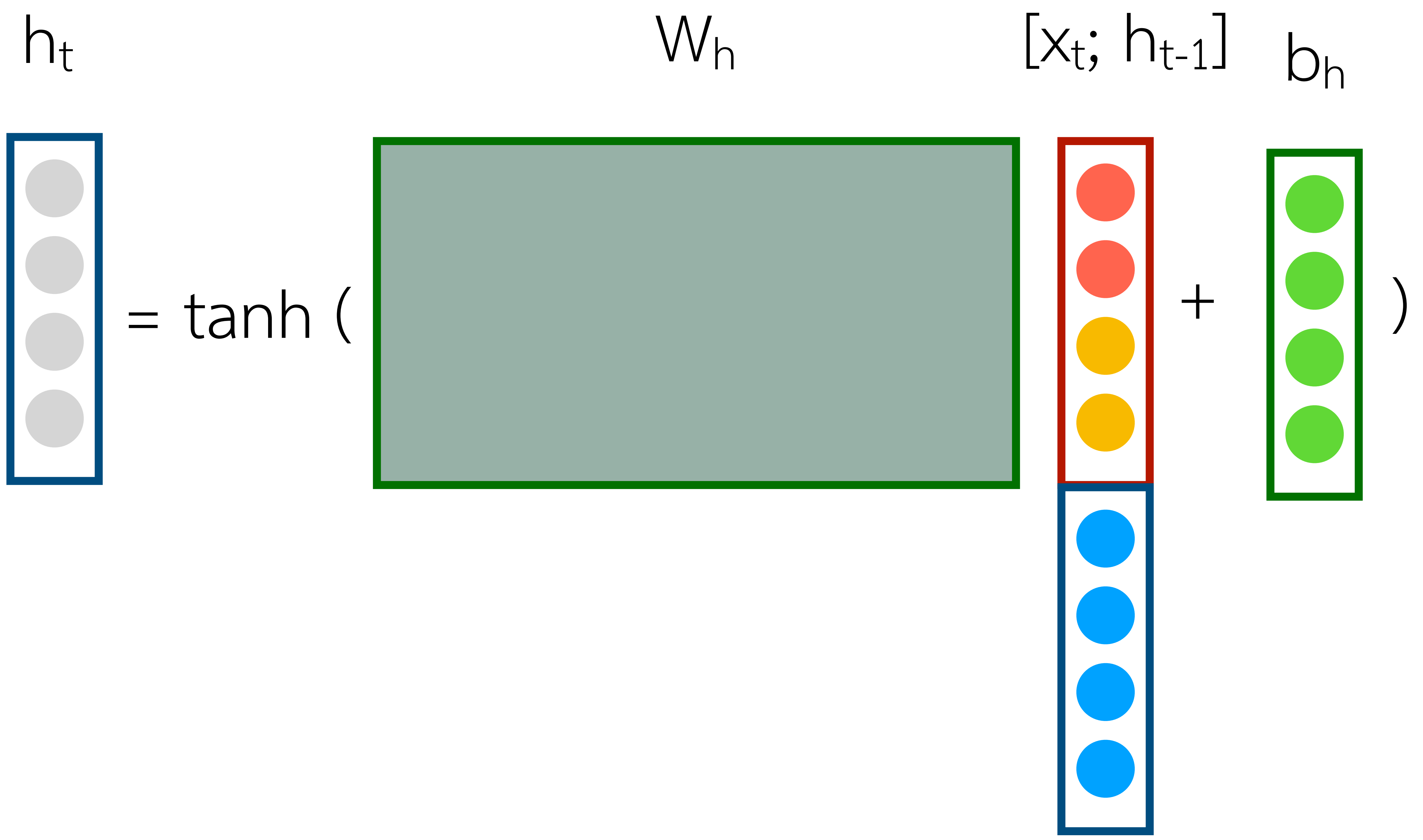


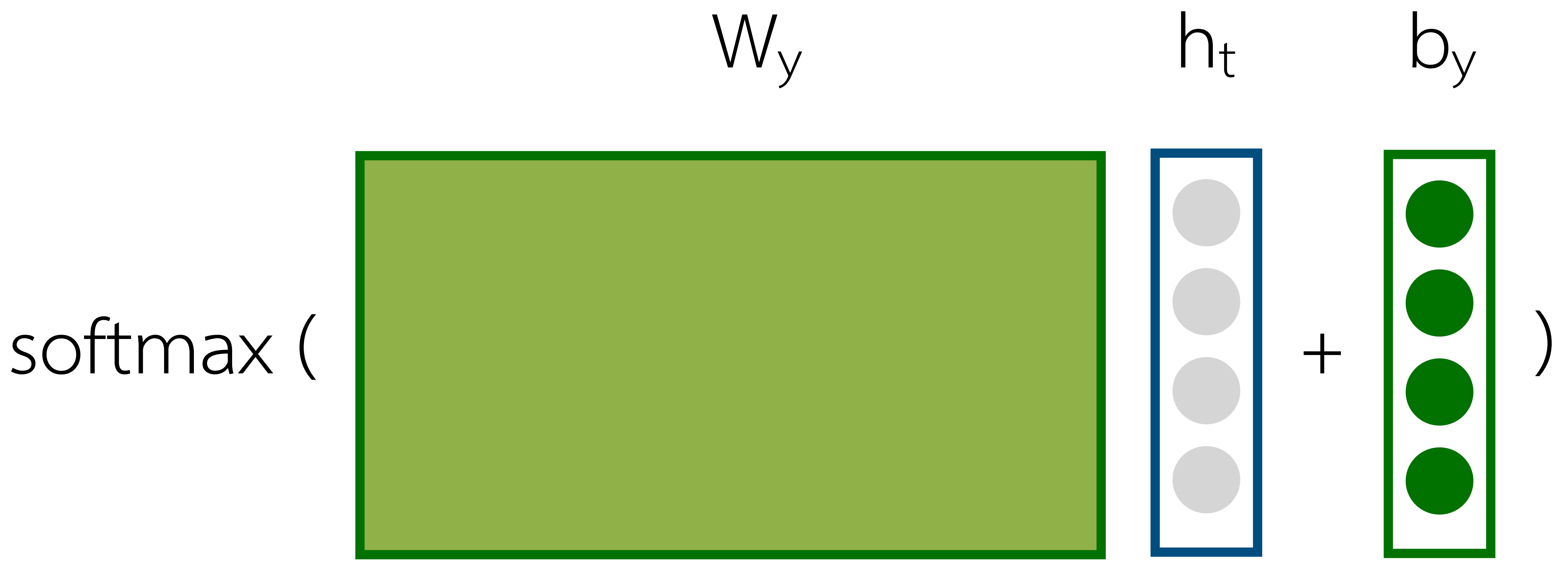

B-PER
I-PER
O
O
B-LOC
Peter
Griffin
lives
in
Quahog

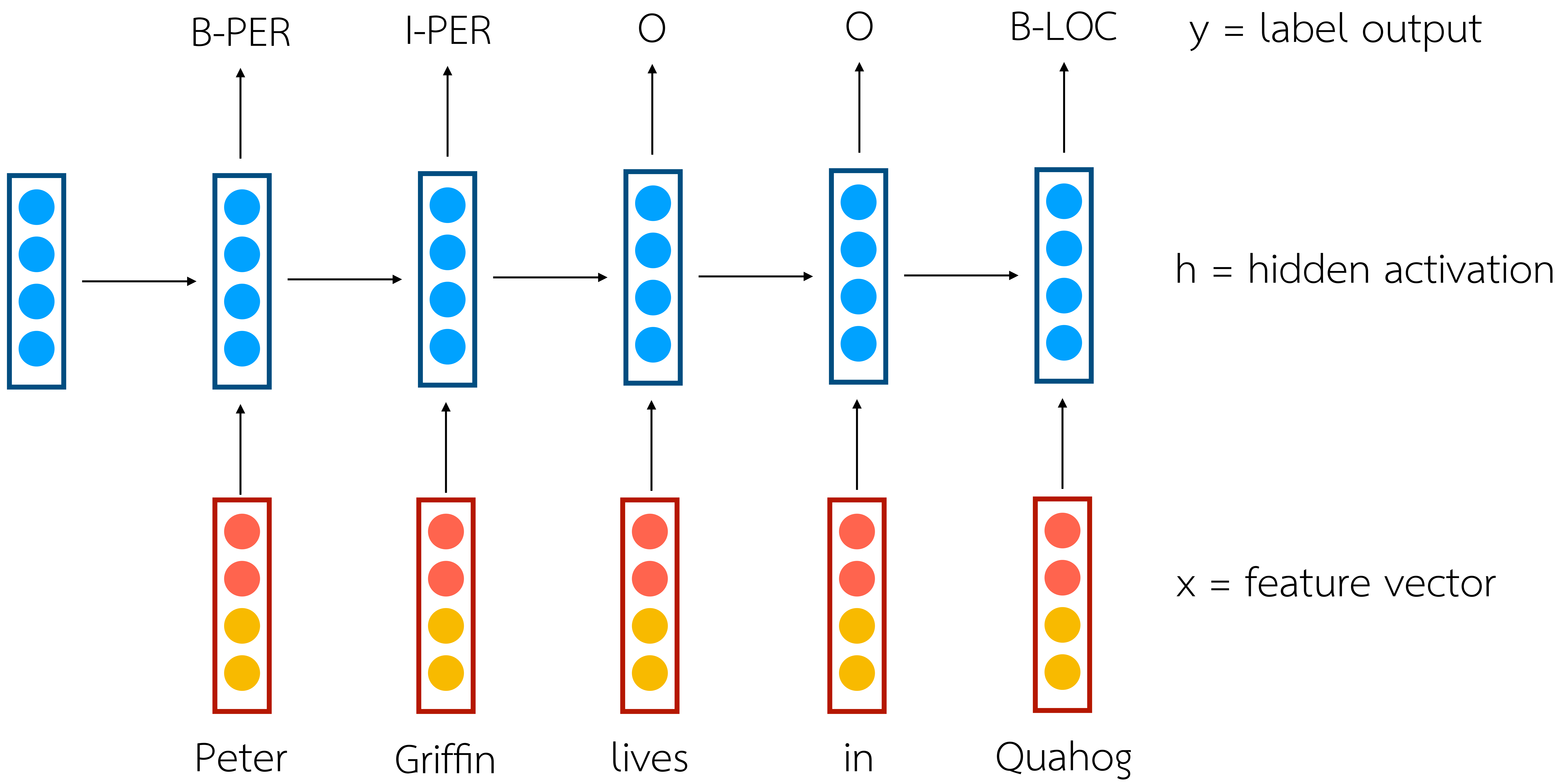
Recurrent Neural Network Parameters

$$h_t = \tanh(W_h \cdot [x_t; h_{t-1}]) + b_h$$

$$y_t = softmax(W_y \cdot h_t + b_y)$$

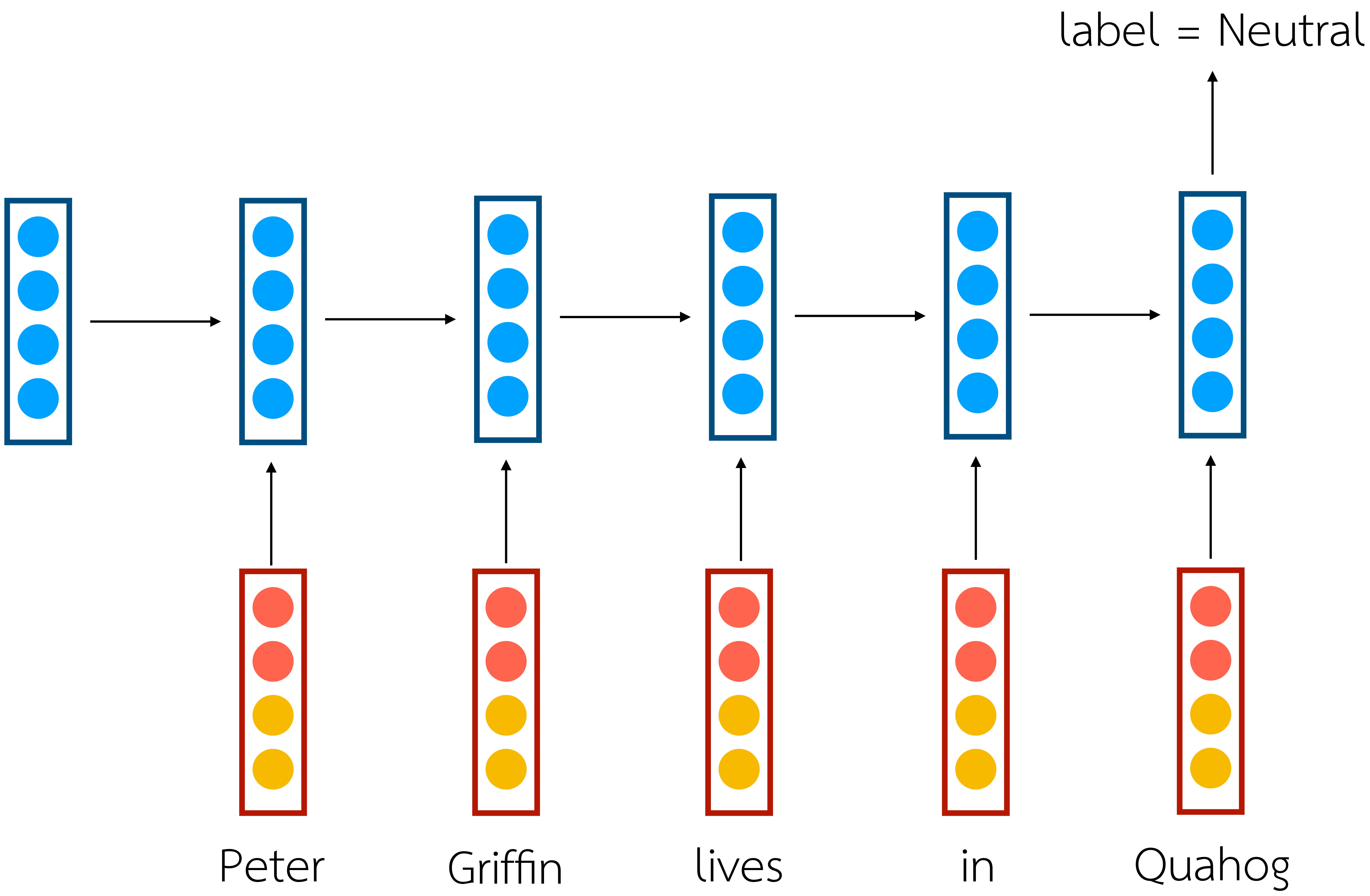






B-PER
I-PER
O
O
B-LOC
y = label output
h = hidden activation
x = feature vector
Peter
Griffin
lives
in
Quahog

RNN as a Classifier



Recurrent Neural Network

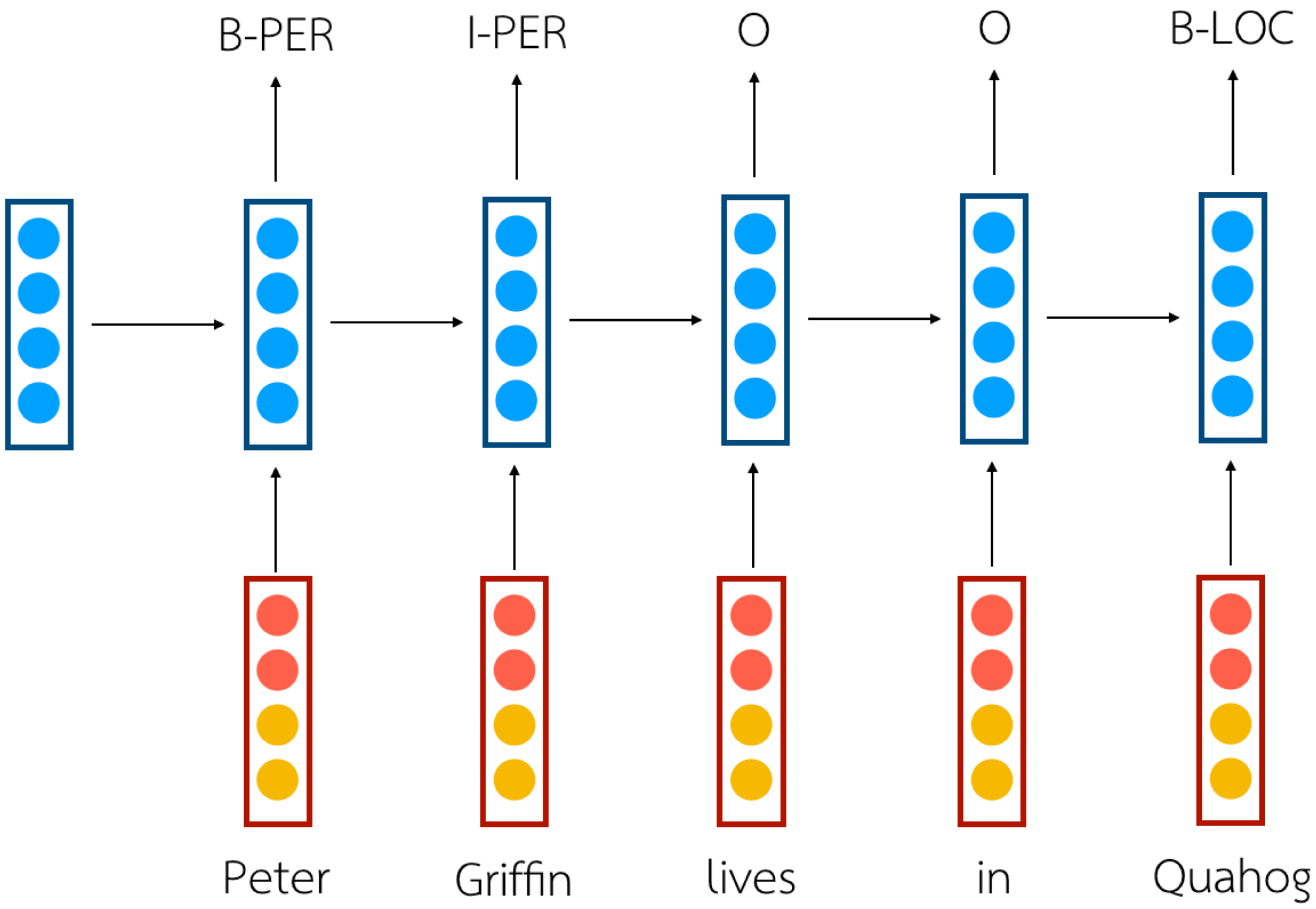
- เหมาะกับ Sequence Labeling ที่ต้องใช้บริบทกว้าง เช่น Language Modeling, NER, ตัดคำ
- เหมาะกับการใช้เป็น classifier เพราะเก็บบริบทได้ครบ
- ในทางปฏิบัติแล้ว train ลำบาก

Training RNN

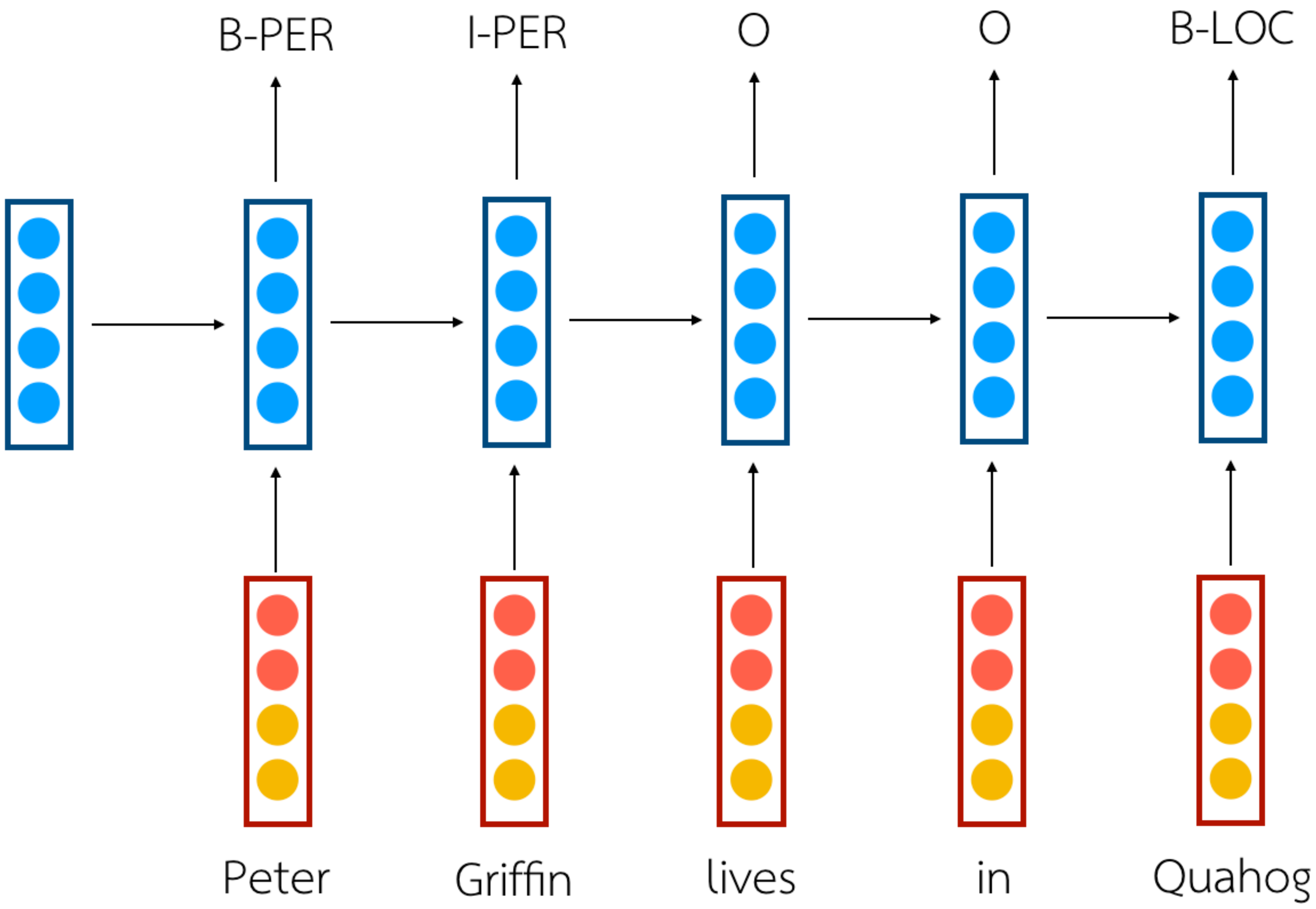
Concept ที่สำคัญ

- Backpropagation Through Time (BPTT) algorithm
- Exploding gradient
- Vanishing gradient

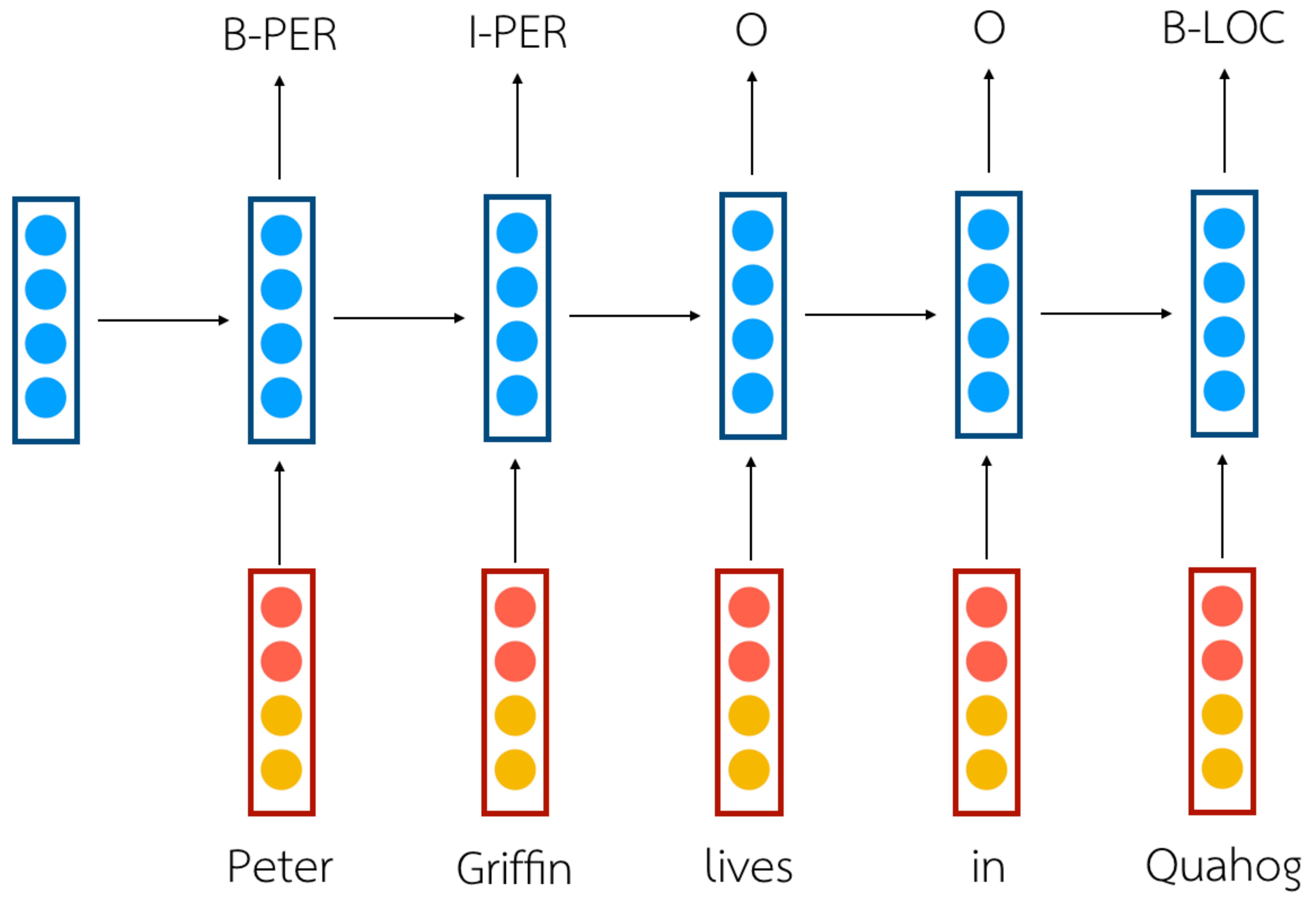
Backpropagation Through Time



Exploding Gradient



Vanishing Gradient



การเทรน RNN

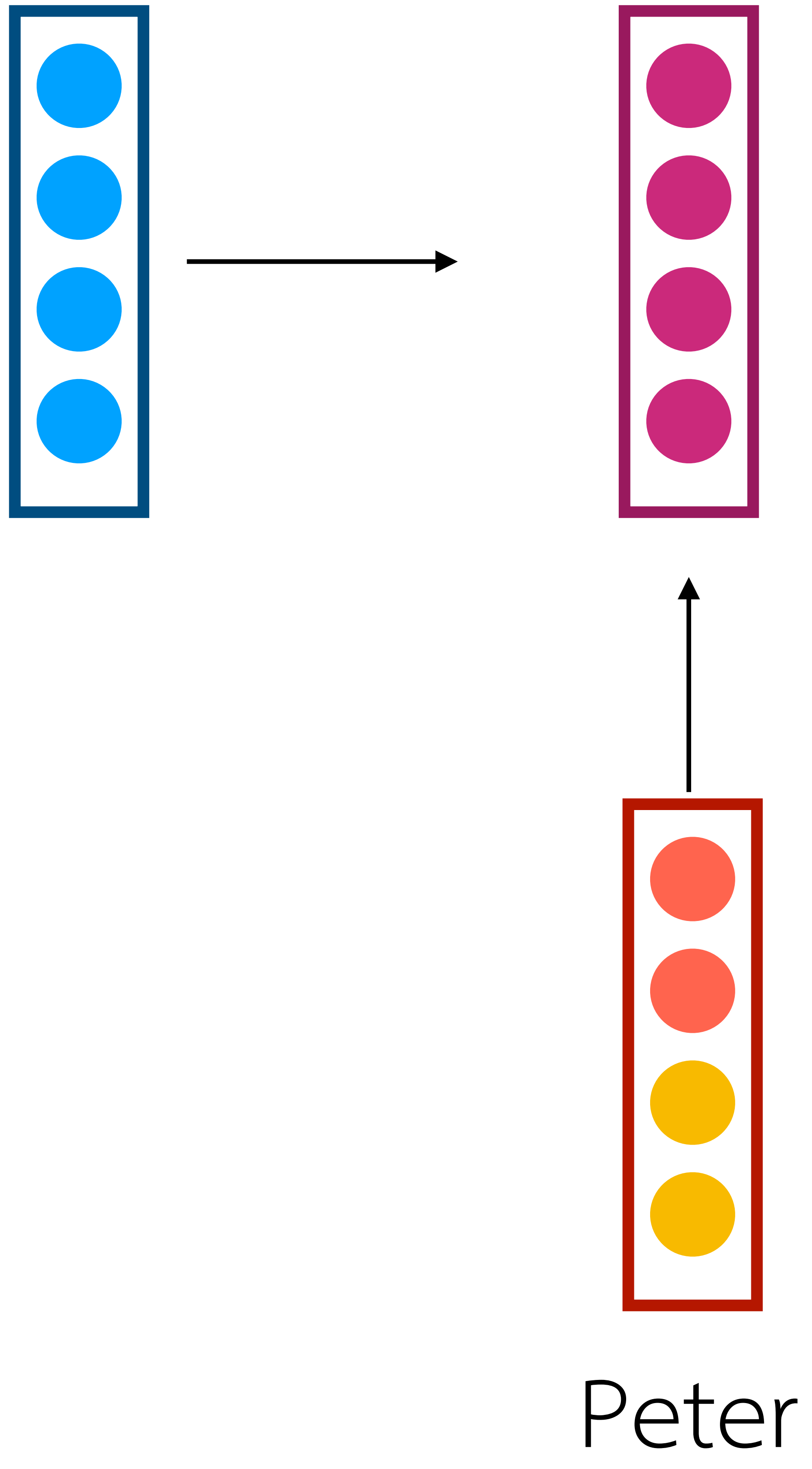
- RNN Parameter น้อย แต่ว่าเทรนลำบาก
- Exploding gradient ทำให้ Loss เป็น NaN หรือ parameter แกว่งมากในแต่ละ iteration --> Gradient Clipping
- Vanishing gradient ทำให้ network ไม่เขยื้อน —> GRU, LSTM

Gated Recurrent Unit (GRU)

+ Long Short-Term Memory (LSTM)

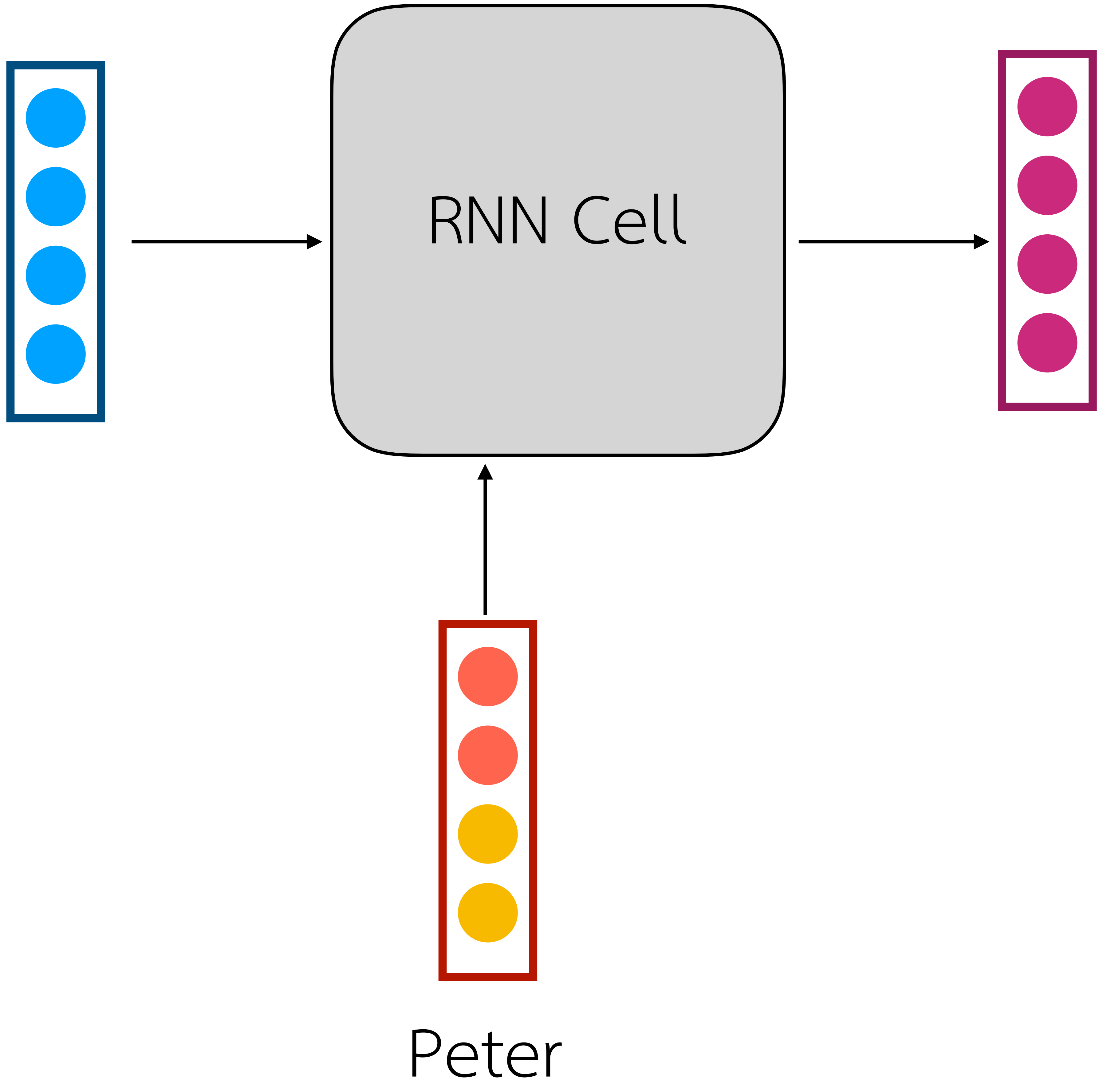
RNN Cell

$$c_t = \tanh(W_c \cdot [c_{t-1}; x_t] + b_c)$$



RNN Cell

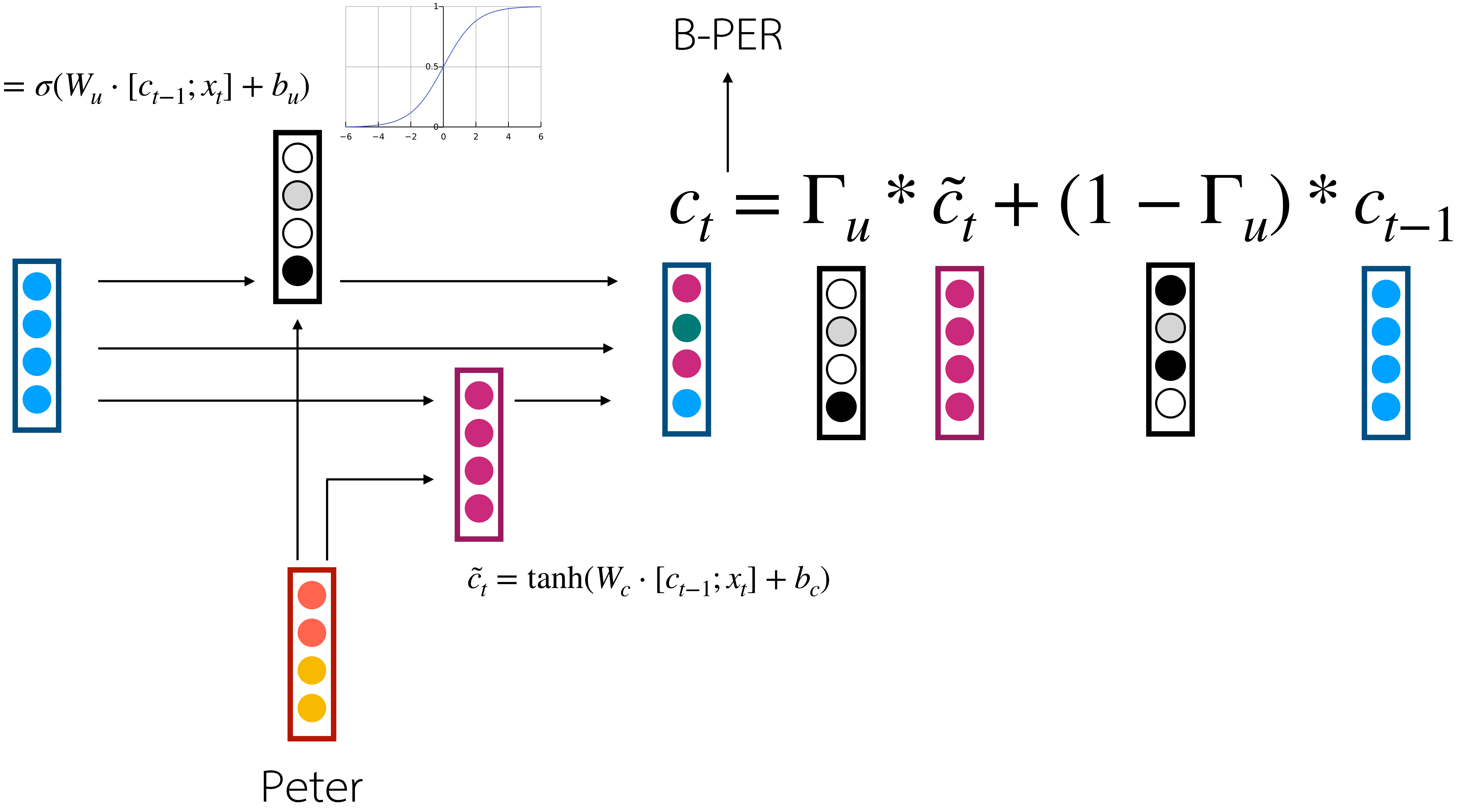
$$c_t = \tanh(W_c \cdot [c_{t-1}; x_t] + b_c)$$



(Simplified) Gated Recurrent Unit

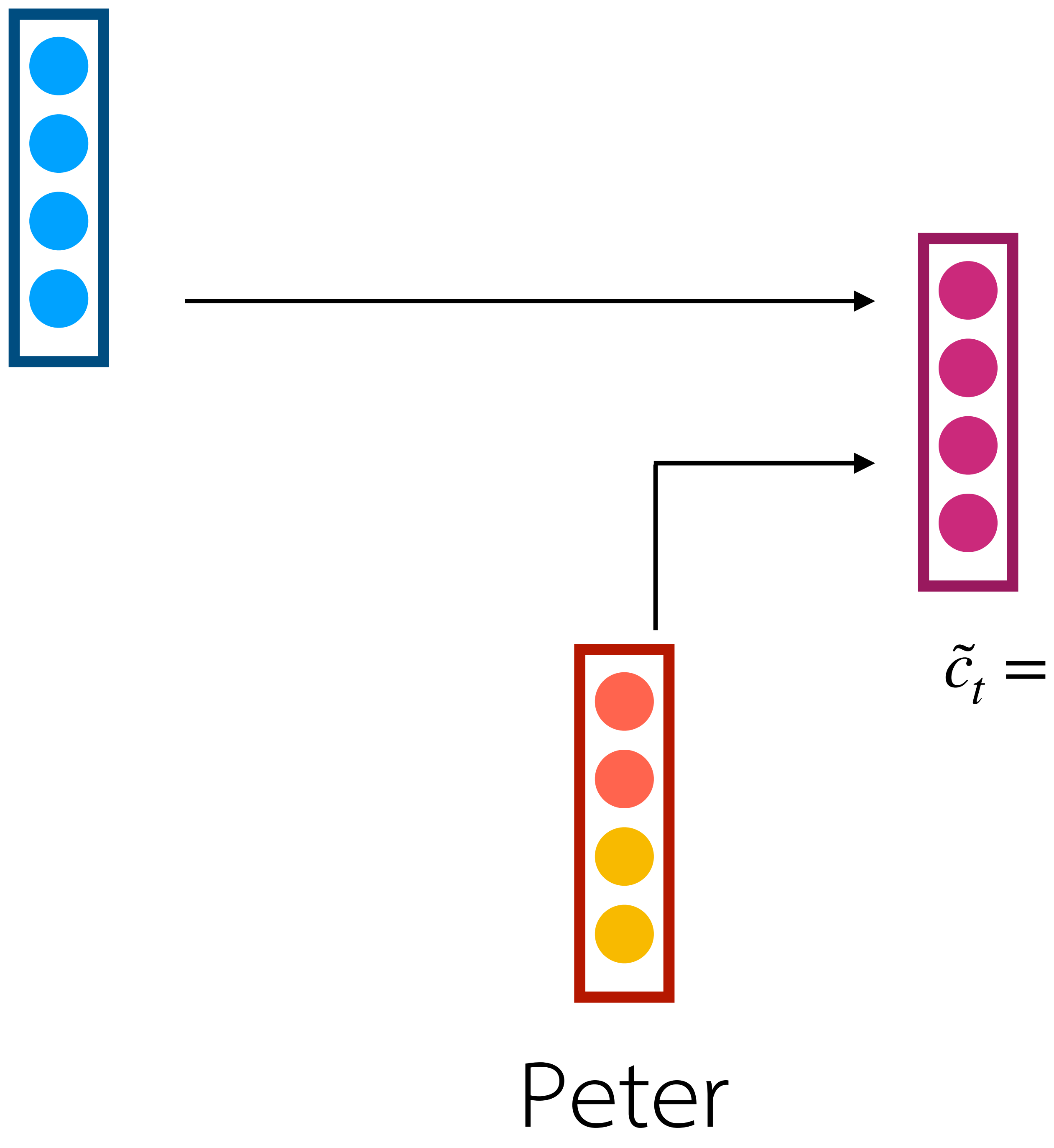
$$\Gamma_u = \sigma(W_u \cdot [c_{t-1}; x_t] + b_u)$$

$$c_t = \Gamma_u * \tilde{c}_t + (1 - \Gamma_u) * c_{t-1}$$



$$\tilde{c}_t = \tanh(W_c \cdot [c_{t-1}; x_t] + b_c)$$

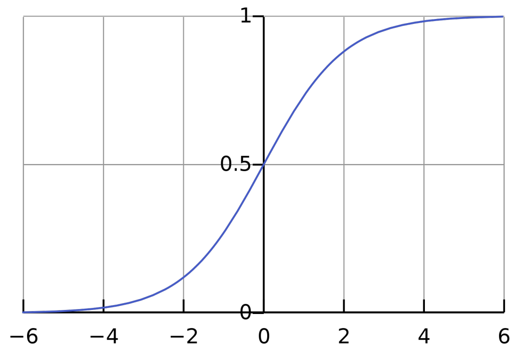
(Simplified) Gated Recurrent Unit

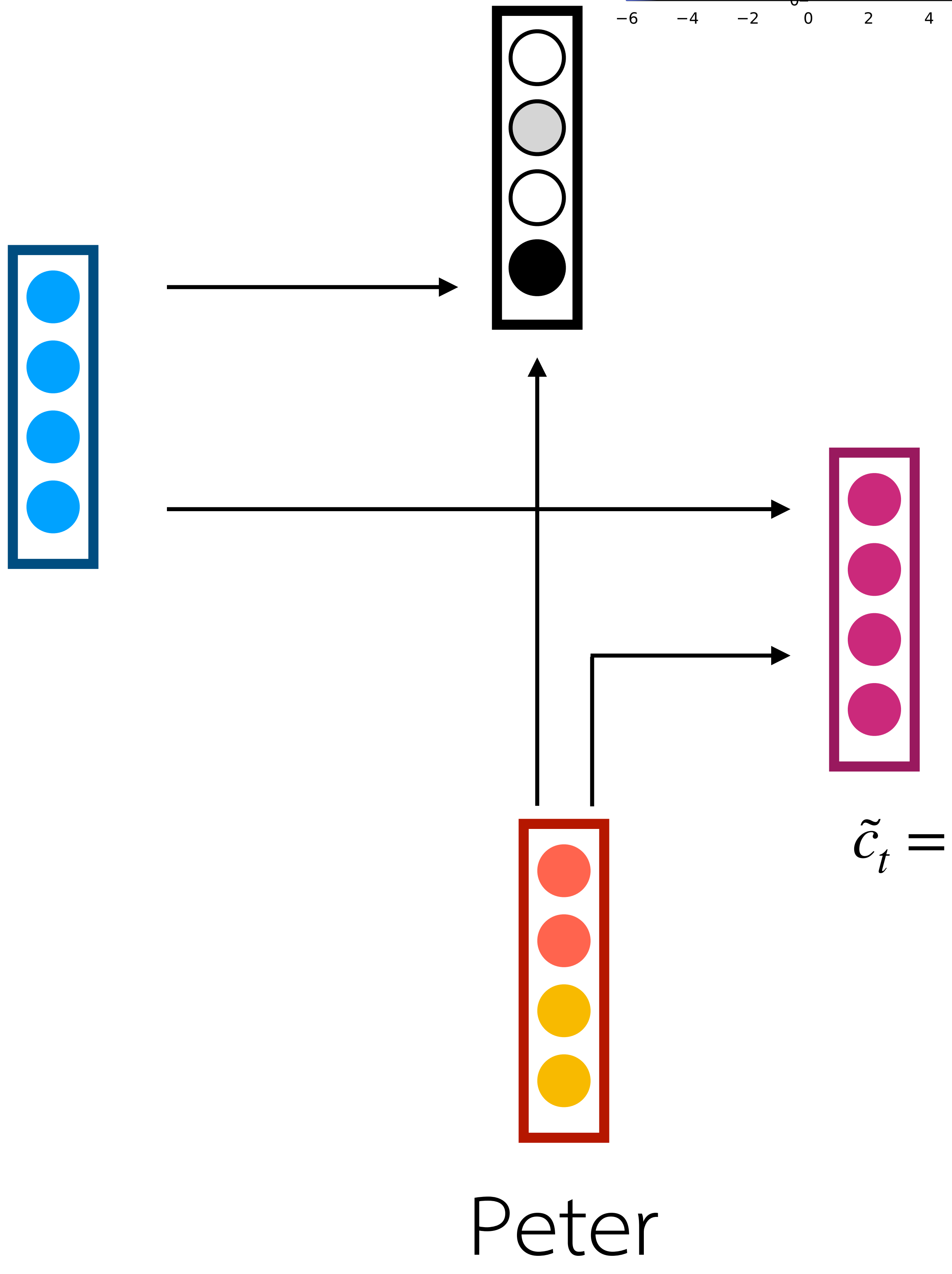


$\tilde{c}_t = \tanh(W_c \cdot [c_{t-1}; x_t] + b_c)$

(Simplified) Gated Recurrent Unit

$$\Gamma_u = \sigma(W_u \cdot [c_{t-1}; x_t] + b_u)$$

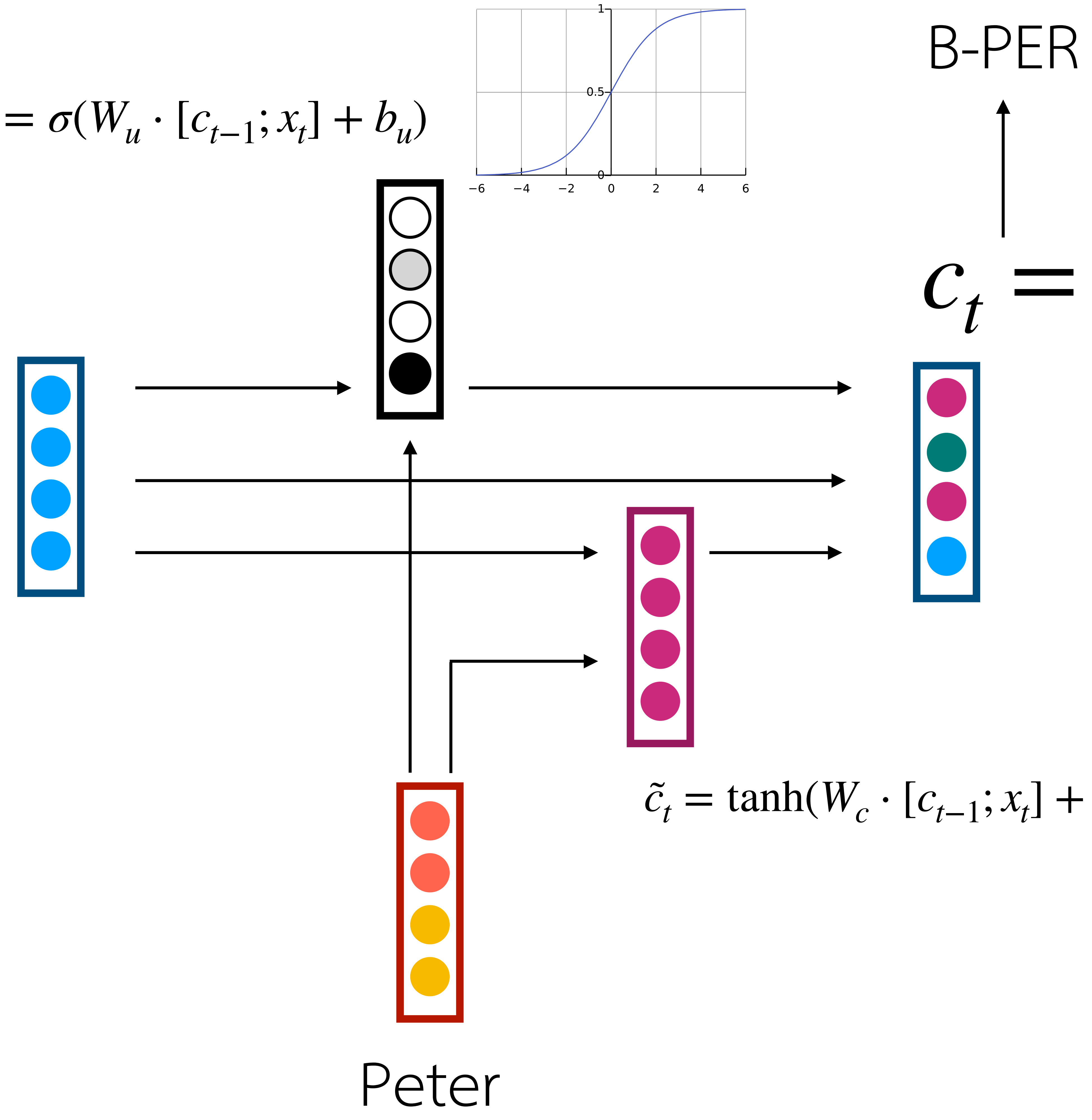




$$\tilde{c}_t = \tanh(W_c \cdot [c_{t-1}; x_t] + b_c)$$

(Simplified) Gated Recurrent Unit

$\Gamma_u = \sigma(W_u \cdot [c_{t-1}; x_t] + b_u)$



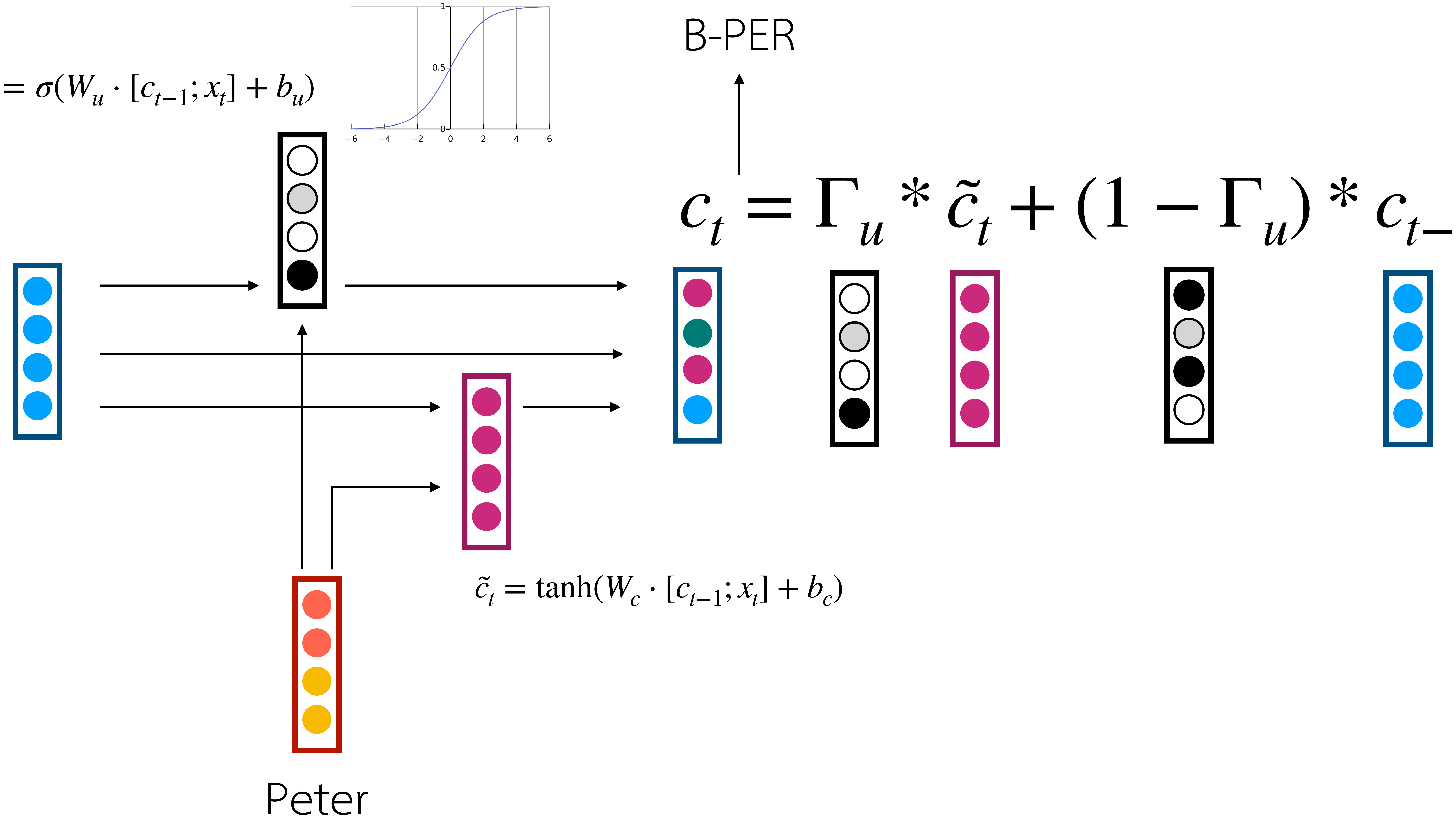
$$c_t = \Gamma_u * \tilde{c}_t + (1 - \Gamma_u) * c_{t-1}$$

$\tilde{c}_t = \tanh(W_c \cdot [c_{t-1}; x_t] + b_c)$

(Simplified) Gated Recurrent Unit

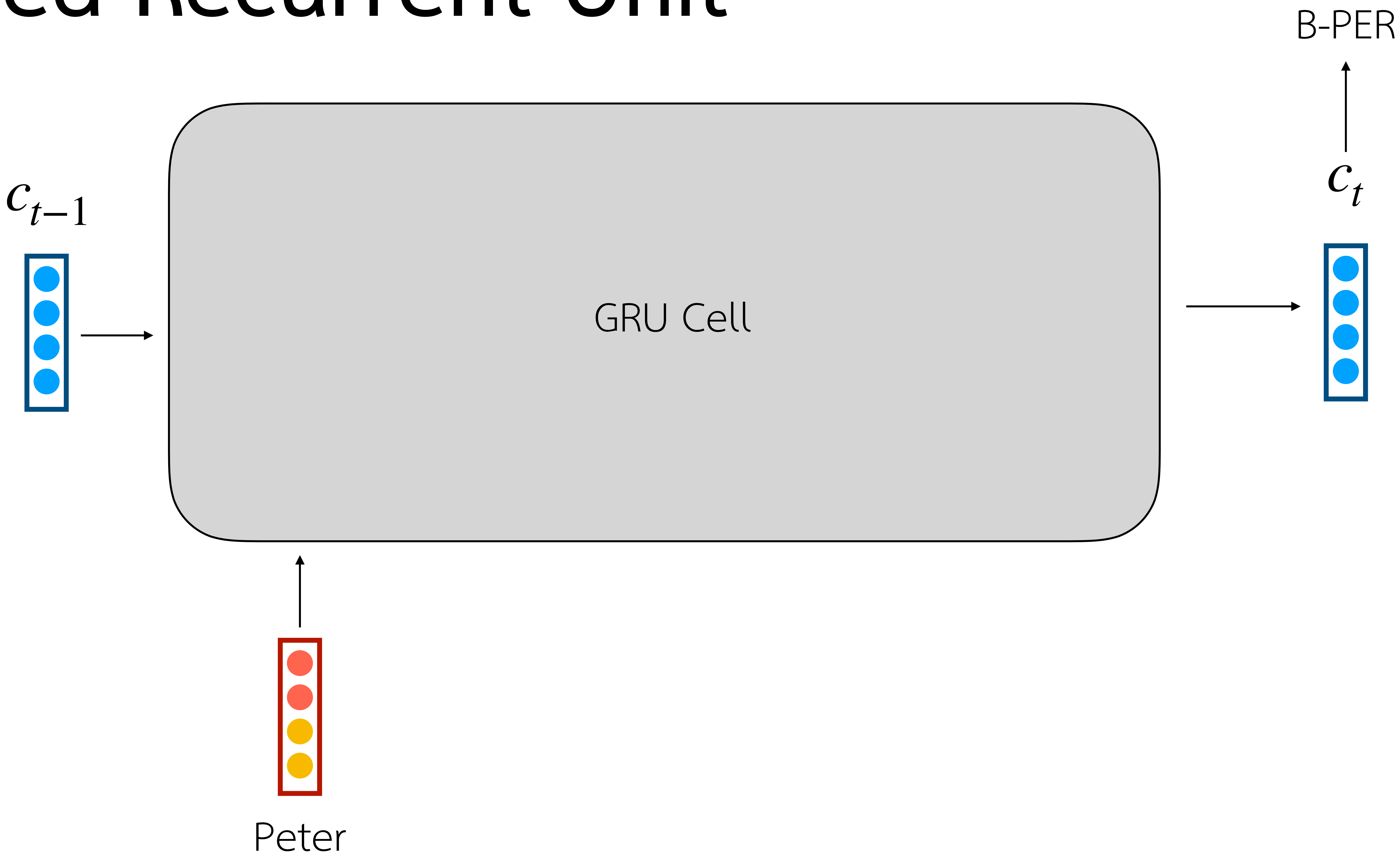
$\Gamma_u = \sigma(W_u \cdot [c_{t-1}; x_t] + b_u)$

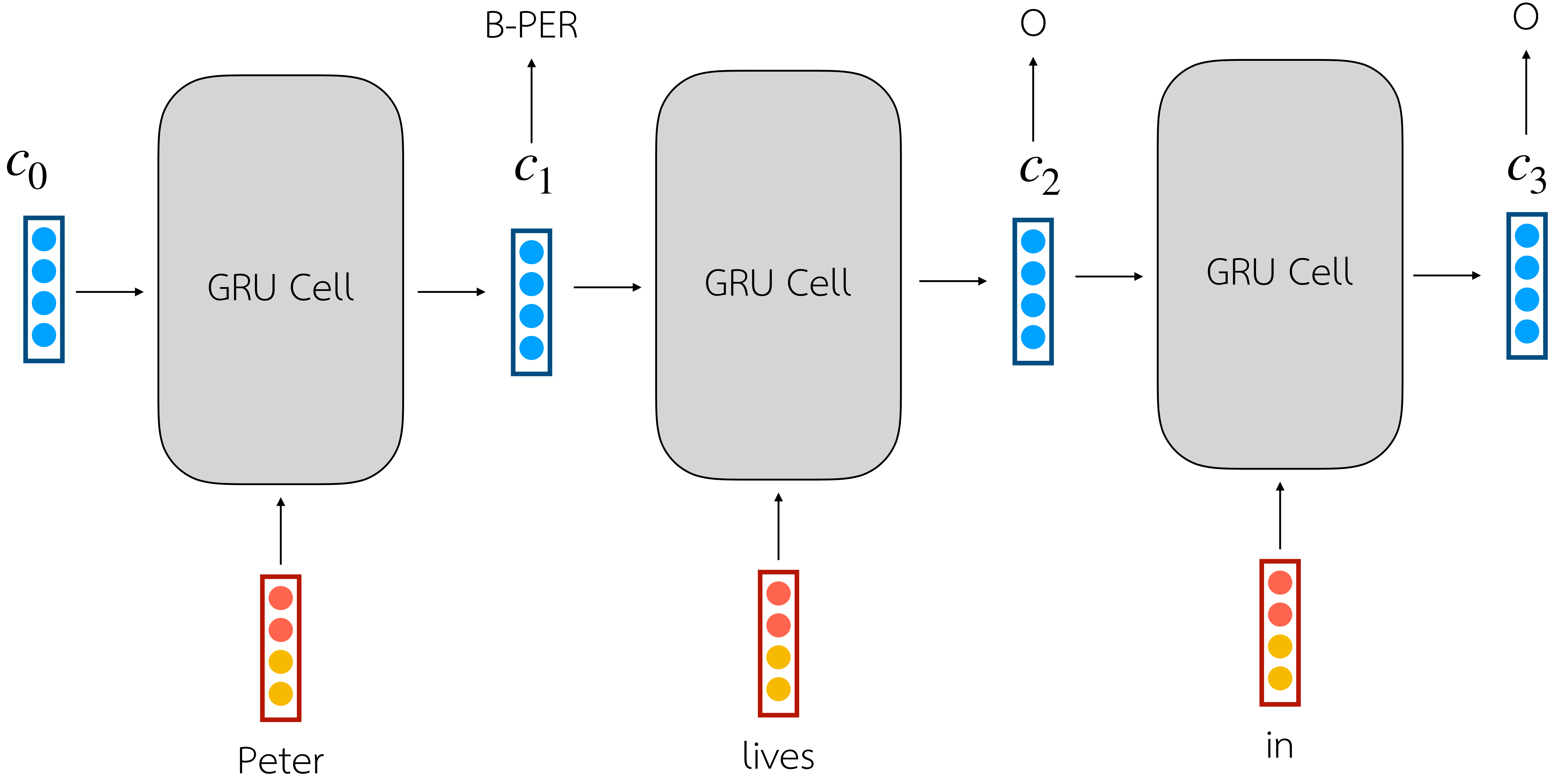
$c_t = \Gamma_u * \tilde{c}_t + (1 - \Gamma_u) * c_{t-1}$



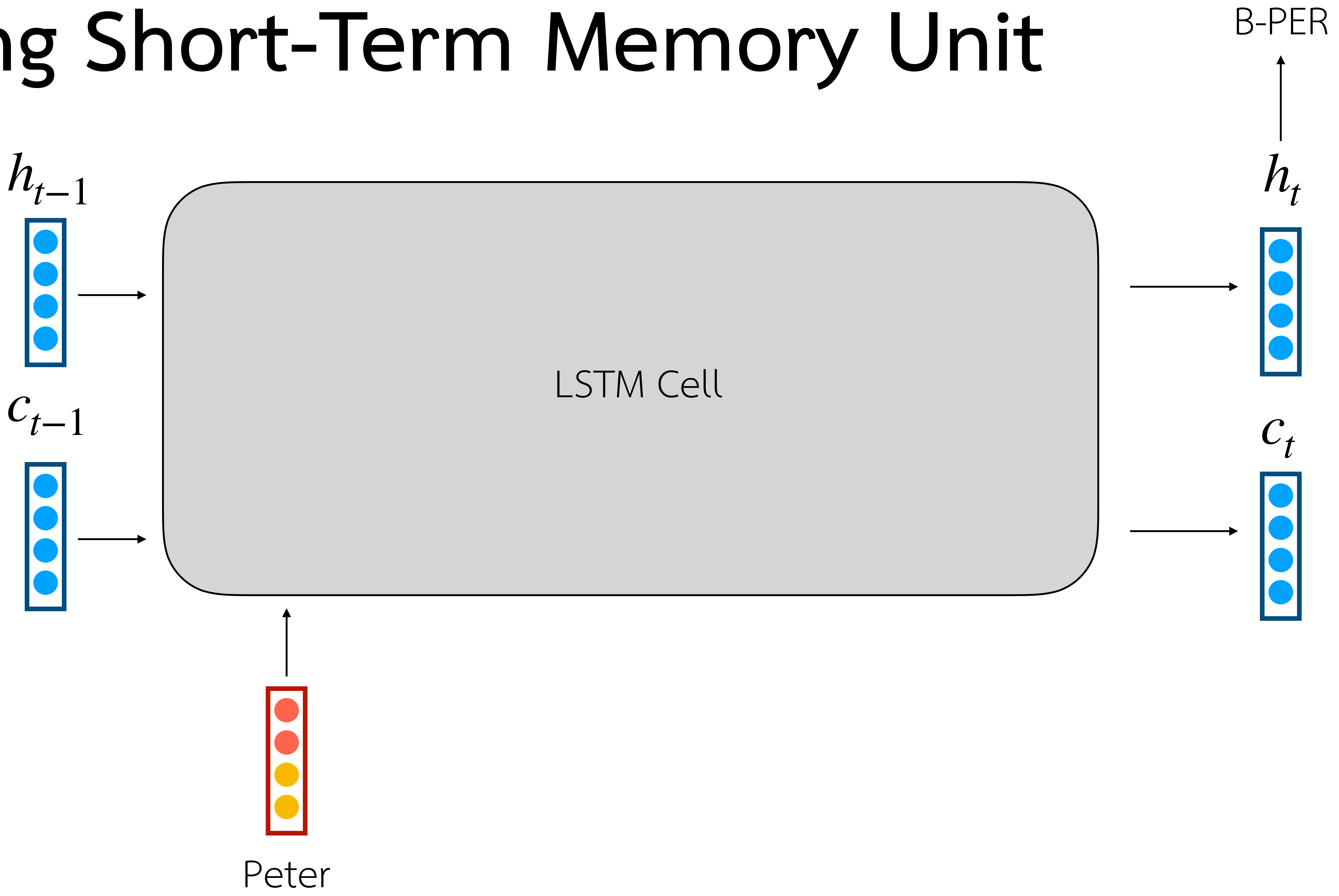
$\tilde{c}_t = \tanh(W_c \cdot [c_{t-1}; x_t] + b_c)$

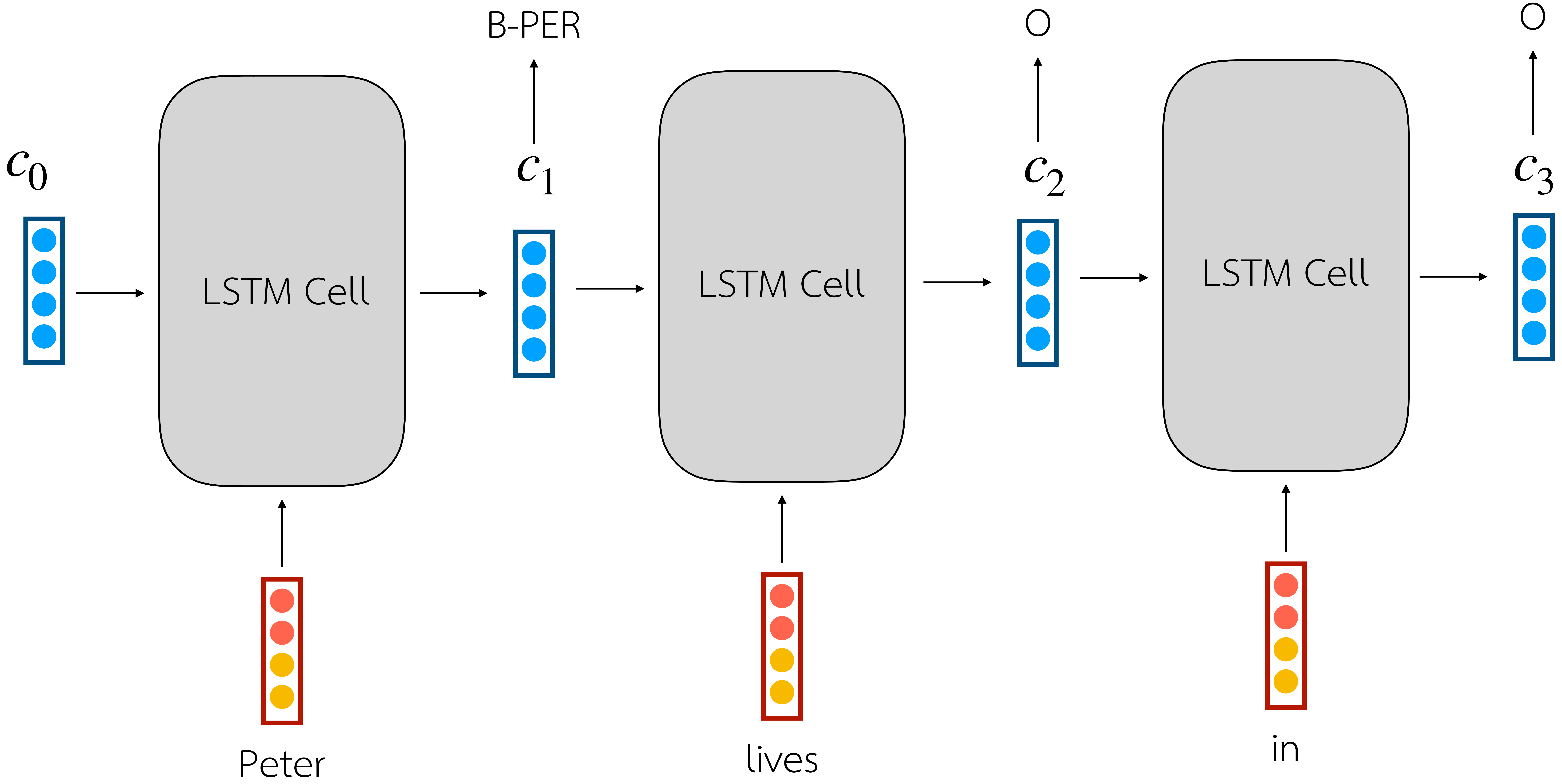
Gated Recurrent Unit

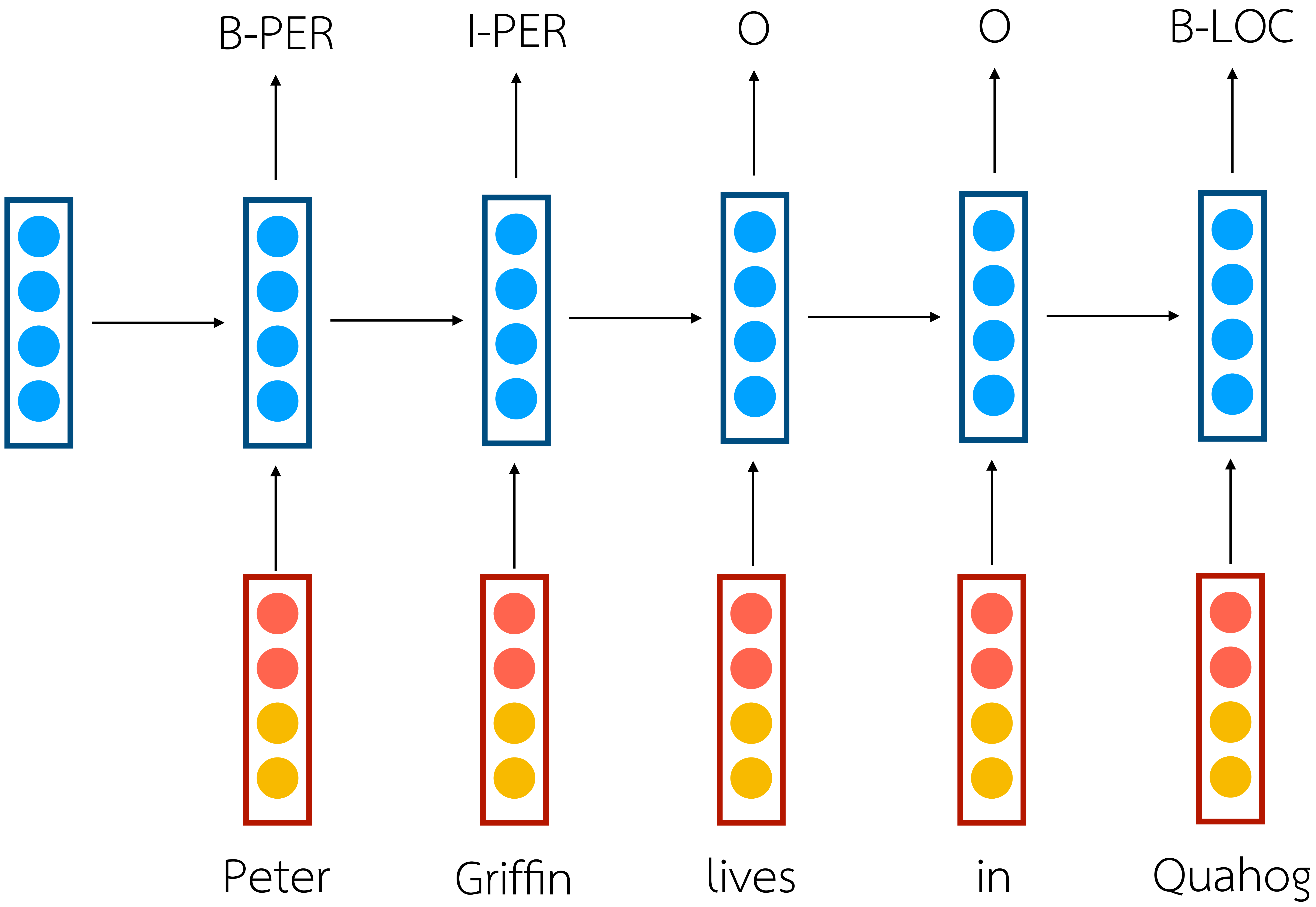


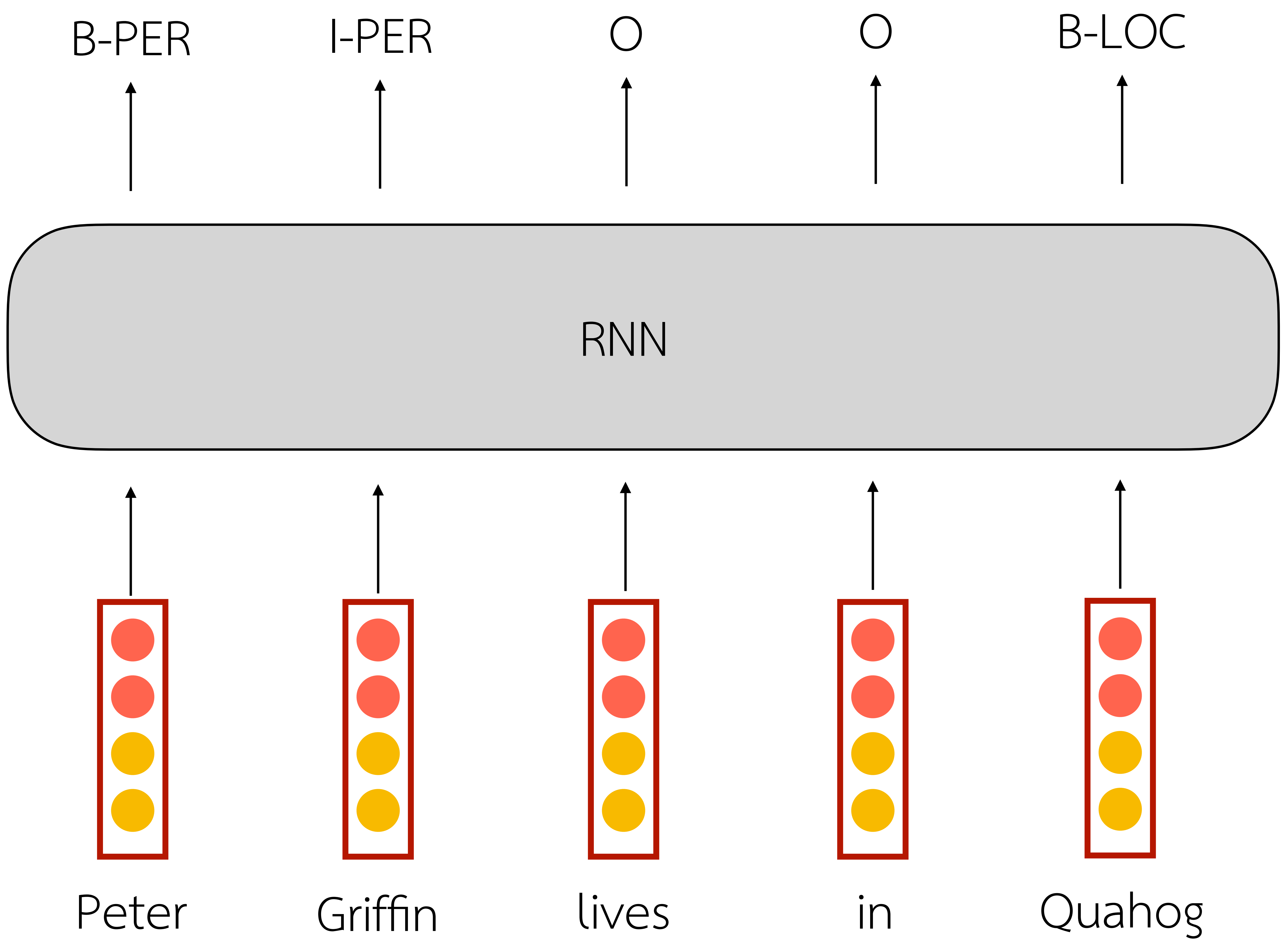

B-PER
O
O
c_0
c_1
c_2
c_3
GRU Cell
GRU Cell
GRU Cell
Peter
lives
in

Long Short-Term Memory Unit




B-PER
O
O
c_0
c_1
c_2
c_3
LSTM Cell
LSTM Cell
LSTM Cell
Peter
lives
in


B-PER
I-PER
O
O
B-LOC
Peter
Griffin
lives
in
Quahog


B-PER
I-PER
O
O
B-LOC
RNN
Peter
Griffin
lives
in
Quahog

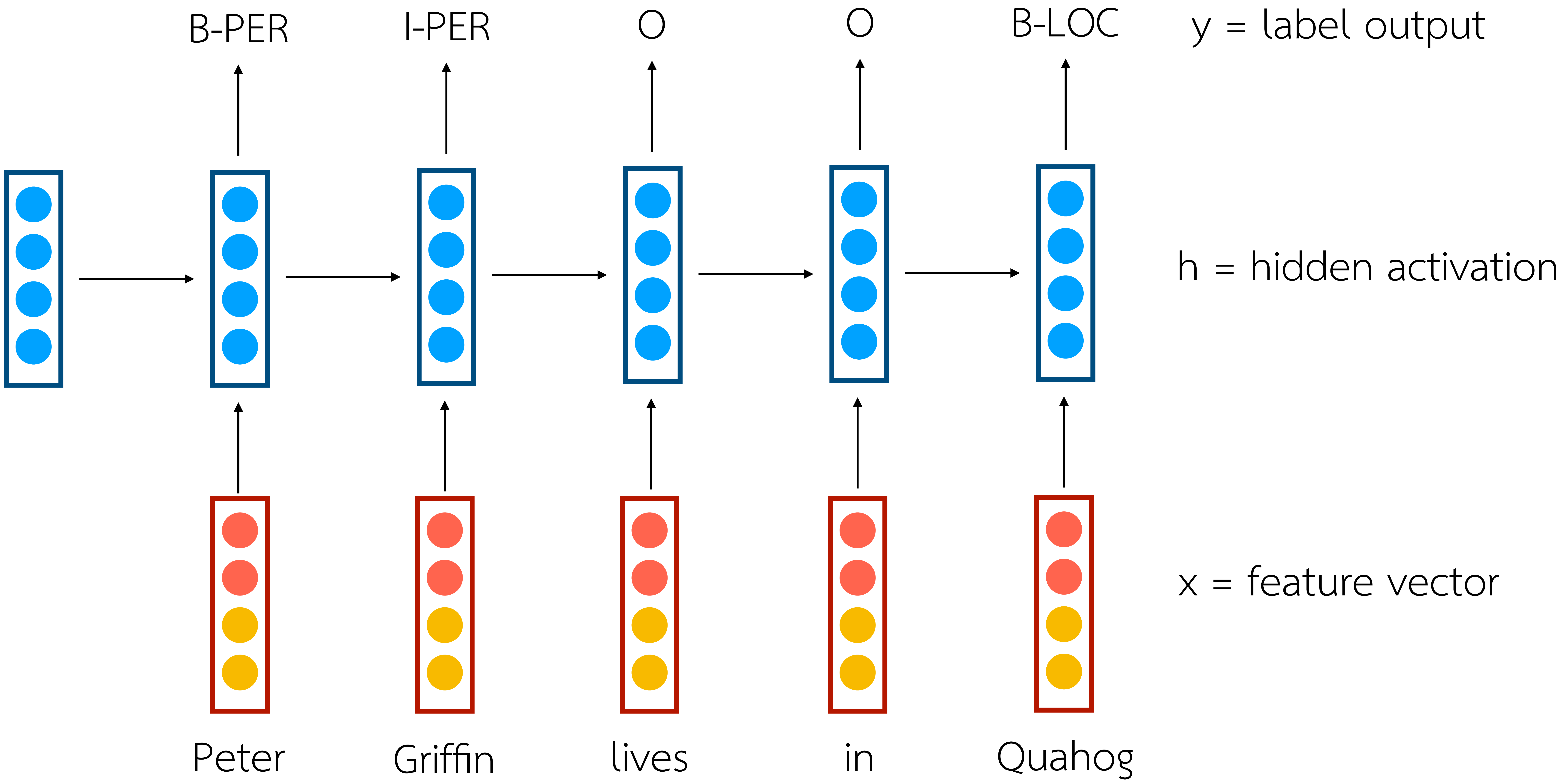
Gated Recurrent Unit

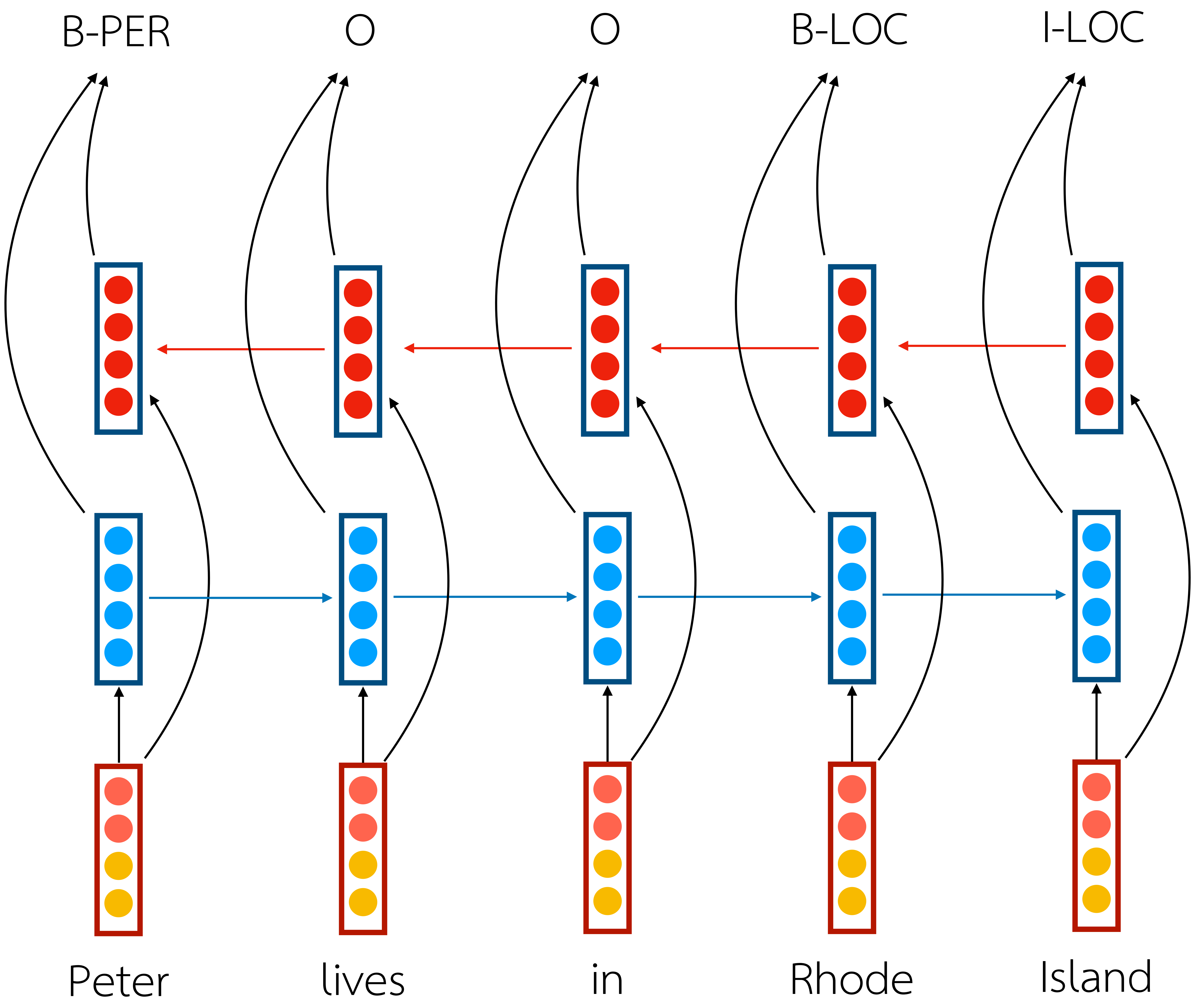
- RNN โดยทั่วไป เรียกว่า Vanilla RNN
- GRU และ LSTM เป็น RNN แบบที่เทรนง่ายขึ้นเพราะ แก้ปัญหา Vanishing gradient ได้ดี แต่ parameter เยอะขึ้น

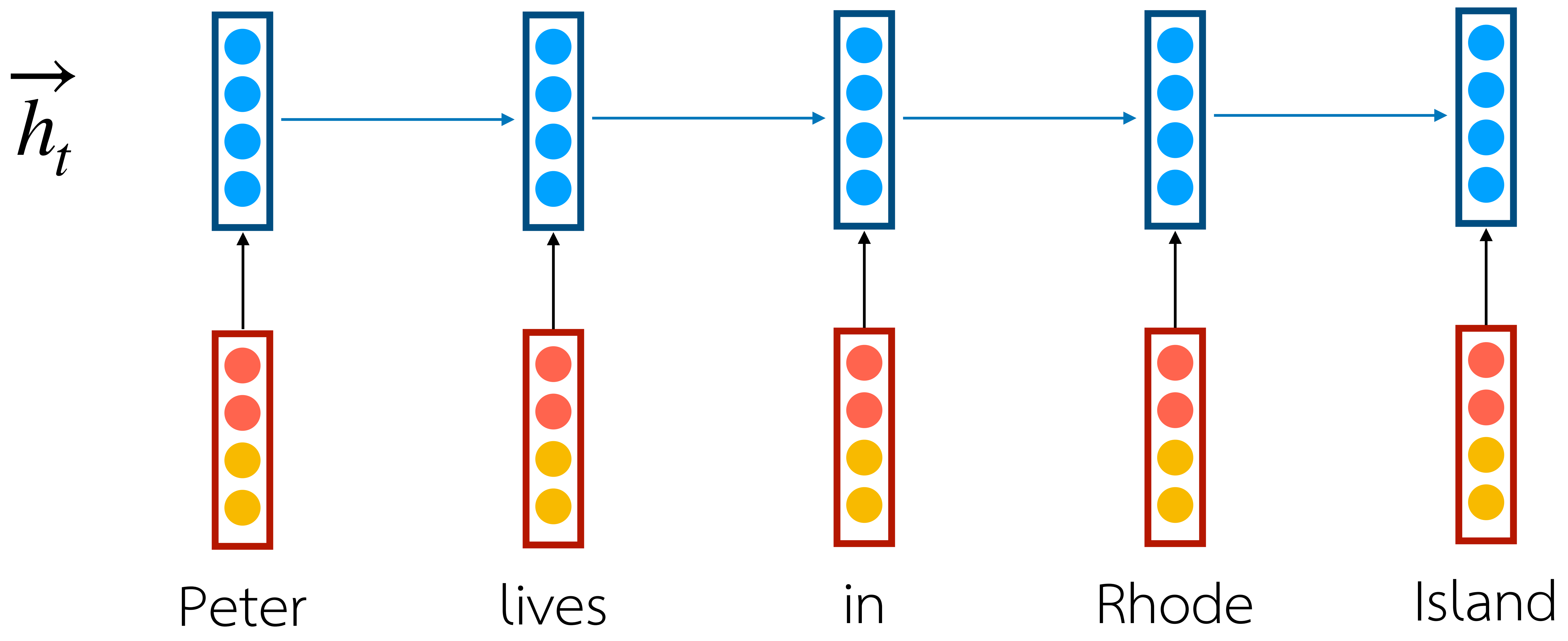
Bidirectional RNN

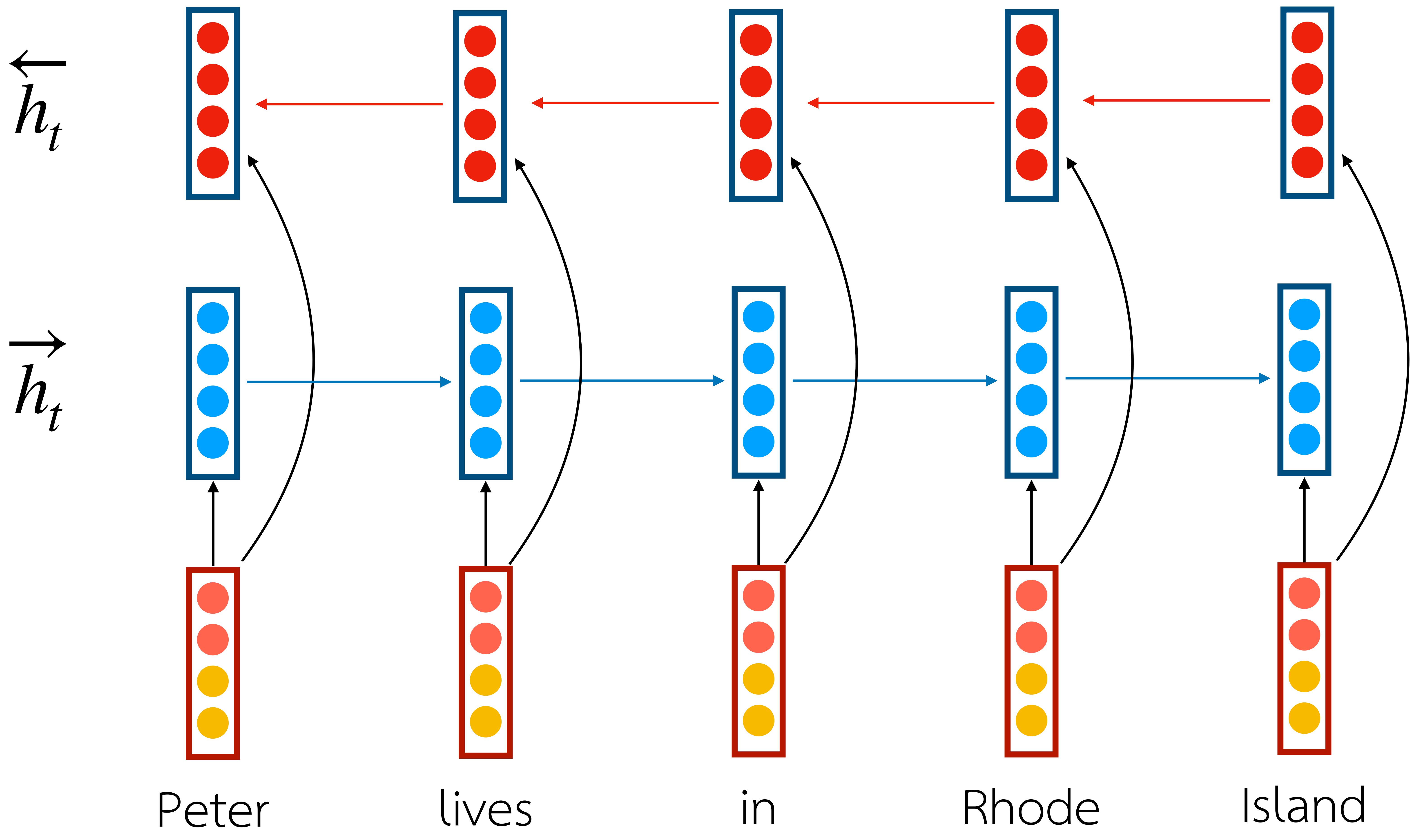
Bidirectional RNN

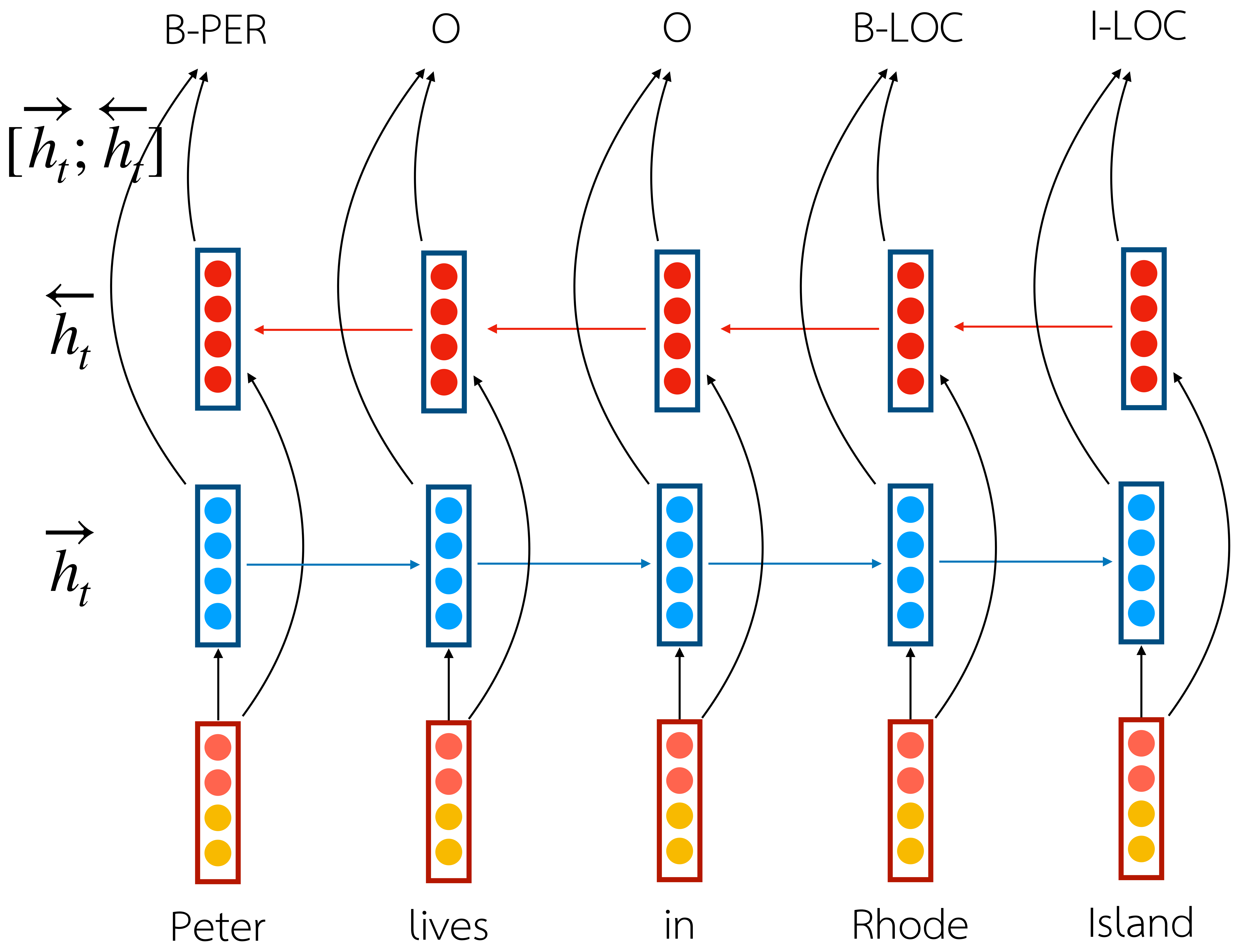
- Bidirectional Gated Recurrent Unit (Bi-GRU)
- Bidirectional Long Short-Term Memory (Bi-LSTM)
- BiLSTM + CRF


B-PER
I-PER
O
O
B-LOC
y = label output
h = hidden activation
x = feature vector
Peter
Griffin
lives
in
Quahog

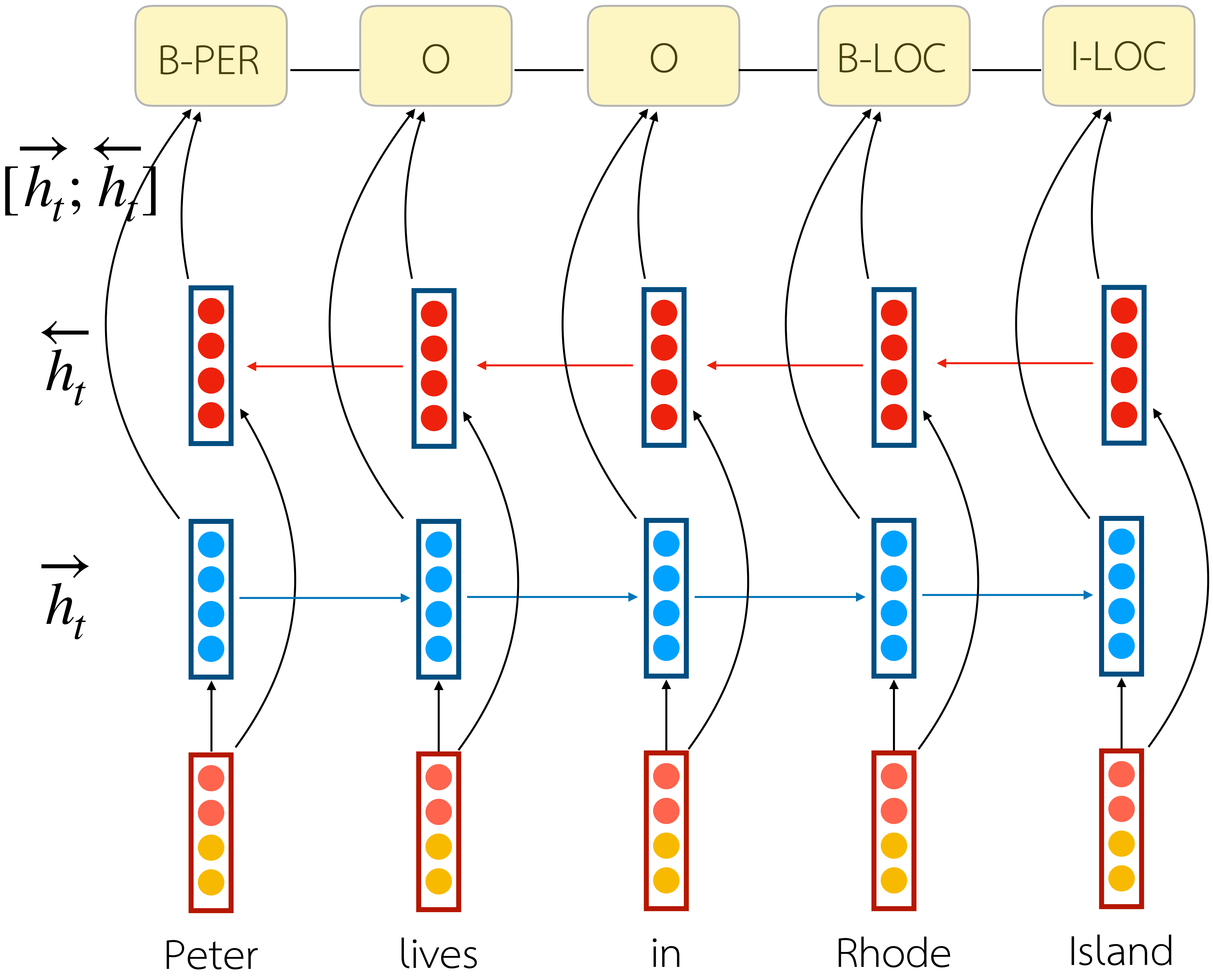

B-PER
O
O
B-LOC
I-LOC
Peter
lives
in
Rhode
Island


$\overrightarrow{h_t}$
Peter
lives
in
Rhode
Island


$\overleftarrow{h_t}$
$\overrightarrow{h_t}$
Peter
lives
in
Rhode
Island



BI-LSTM / BI-GRU


B-PER
O
O
B-LOC
I-LOC
$[\overrightarrow{h_t}; \overleftarrow{h_t}]$
$\overleftarrow{h_t}$
$\overrightarrow{h_t}$
Peter
lives
in
Rhode
Island

BI-LSTM-CRF

Bi-LSTM-CRF in Practice

Word Embedding vs Discrete Features

Tagging performance on POS, chunking and NER tasks with only word features.

		POS	CoNLL2000	CoNLL2003
Senna	LSTM	94.63 (-2.66)	90.11 (-2.88)	75.31 (-8.43)
	BI-LSTM	96.04 (-1.36)	93.80 (-0.12)	83.52 (-1.65)
	CRF	94.23 (-3.22)	85.34 (-8.49)	77.41 (-8.72)
	LSTM-CRF	95.62 (-1.92)	93.13 (-1.14)	81.45 (-6.91)
	BI-LSTM-CRF	**96.11** (-1.44)	**94.40** (-0.06)	**84.74** (-4.09)

- Discrete features เหมาะกับ CRF
- Word embedding เหมาะกับ LSTM

Huang, Zhiheng, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF models for sequence tagging." *arXiv preprint arXiv:1508.01991* (2015).

ควรใช้ Pre-trained Embedding

		POS	CoNLL2000	CoNLL2003
Random	Conv-CRF (Collobert et al., 2011)	96.37	90.33	81.47
	LSTM	97.10	92.88	79.82
	BI-LSTM	97.30	93.64	81.11
	CRF	97.30	93.69	83.02
	LSTM-CRF	**97.45**	93.80	84.10
	BI-LSTM-CRF	97.43	**94.13**	**84.26**
Senna	Conv-CRF (Collobert et al., 2011)	97.29	94.32	88.67 (89.59)
	LSTM	97.29	92.99	83.74
	BI-LSTM	97.40	93.92	85.17
	CRF	97.45	93.83	86.13
	LSTM-CRF	97.54	94.27	88.36
	BI-LSTM-CRF	**97.55**	**94.46**	**88.83 (90.10)**

Huang, Zhiheng, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF models for sequence tagging." *arXiv preprint arXiv:1508.01991* (2015).

Almost State-of-the-art POS tagging

System	accuracy	extra data
Maximum entropy cyclic dependency network (Toutanova et al., 2003)	97.24	No
SVM-based tagger (Gimenez and Marquez, 2004)	97.16	No
Bidirectional perceptron learning (Shen et al., 2007)	97.33	No
Semi-supervised condensed nearest neighbor (Soegaard, 2011)	97.50	Yes
CRFs with structure regularization (Sun, 2014)	97.36	No
Conv network tagger (Collobert et al., 2011)	96.37	No
Conv network tagger (senna) (Collobert et al., 2011)	97.29	Yes
BI-LSTM-CRF (ours)	**97.43**	No
BI-LSTM-CRF (Senna) (ours)	**97.55**	Yes

Huang, Zhiheng, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF models for sequence tagging." *arXiv preprint arXiv:1508.01991* (2015).

Almost State-of-the-art NER

System	accuracy
Combination of HMM, Maxent etc. (Florian et al., 2003)	88.76
MaxEnt classifier (Chieu., 2003)	88.31
Semi-supervised model combination (Ando and Zhang., 2005)	89.31
Conv-CRF (Collobert et al., 2011)	81.47
Conv-CRF (Senna + Gazetteer) (Collobert et al., 2011)	89.59
CRF with Lexicon Infused Embeddings (Passos et al., 2014)	**90.90**
BI-LSTM-CRF (ours)	84.26
BI-LSTM-CRF (Senna + Gazetteer) (ours)	90.10

Huang, Zhiheng, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF models for sequence tagging." *arXiv preprint arXiv:1508.01991* (2015).

สรุปคือยังไง

- Bi-LSTM-CRF เป็นโมเดลที่มีประสิทธิภาพ เทรนไม่ยากมาก และใช้กันแพร่หลายตอนนี้ (ปี 2020)
- ควรจะใช้ pre-trained embedding + discrete features
- ไม่แน่เสมอไปว่าจะดีกว่า CRF หรือแม้แต่ Maximum Entropy