CS 4248
Natural Language Processing

Professor NG Hwee Tou
Department of Computer Science
School of Computing
National University of Singapore
nght@comp.nus.edu.sg

Materials

- NNM4NLP Chapter 17

Fixed-Window Neural Language Model

- Input to multilayer perceptron: a sequence of k words
- Output: a probability distribution over the next word

Fixed Window Neural Language Model

$$v(w) = \boldsymbol{E}_{[w]}$$

$$\boldsymbol{x} = [v(w_1); v(w_2); \ldots; v(w_k)]$$

$$\boldsymbol{h} = g(\boldsymbol{x}\boldsymbol{W}^{\mathbf{1}} + \boldsymbol{b}^{\mathbf{1}})$$

$$\widehat{\boldsymbol{y}} = \mathrm{softmax}(\boldsymbol{h}\boldsymbol{W}^{\mathbf{2}} + \boldsymbol{b}^{\mathbf{2}})$$

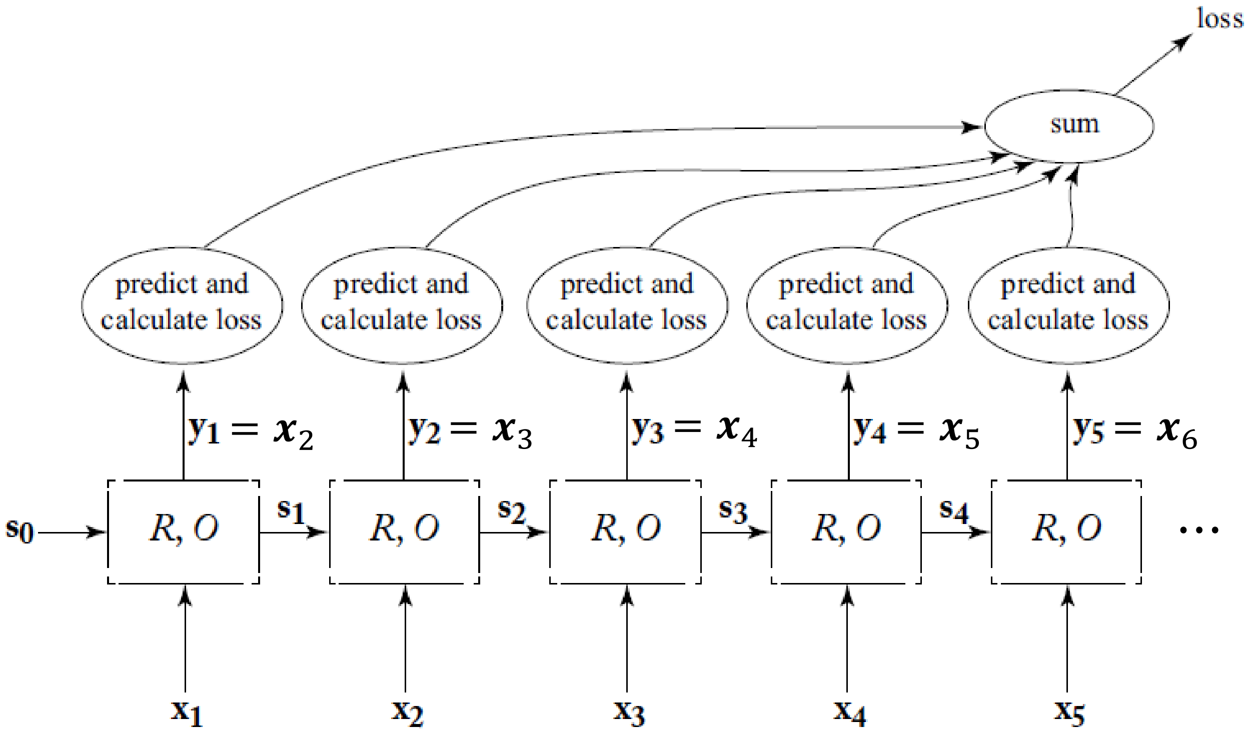
$$w_i \in V \quad \boldsymbol{E} \in \mathbb{R}^{|V| \times d_w} \quad \boldsymbol{W}^{\mathbf{1}} \in \mathbb{R}^{k \cdot d_w \times d_{\mathrm{hid}}} \quad \boldsymbol{b}^{\mathbf{1}} \in \mathbb{R}^{d_{\mathrm{hid}}}$$

$$\boldsymbol{h} \in \mathbb{R}^{d_{\mathrm{hid}}} \quad \boldsymbol{W}^{\mathbf{2}} \in \mathbb{R}^{d_{\mathrm{hid}} \times |V|} \quad \boldsymbol{b}^{\mathbf{2}} \in \mathbb{R}^{|V|}$$

Fixed Window Neural LM

- Fixed window context: unable to use a larger context
- RNN language model
  - Able to use a context of any length
  - Model size does not increase with larger context

RNN Language Model

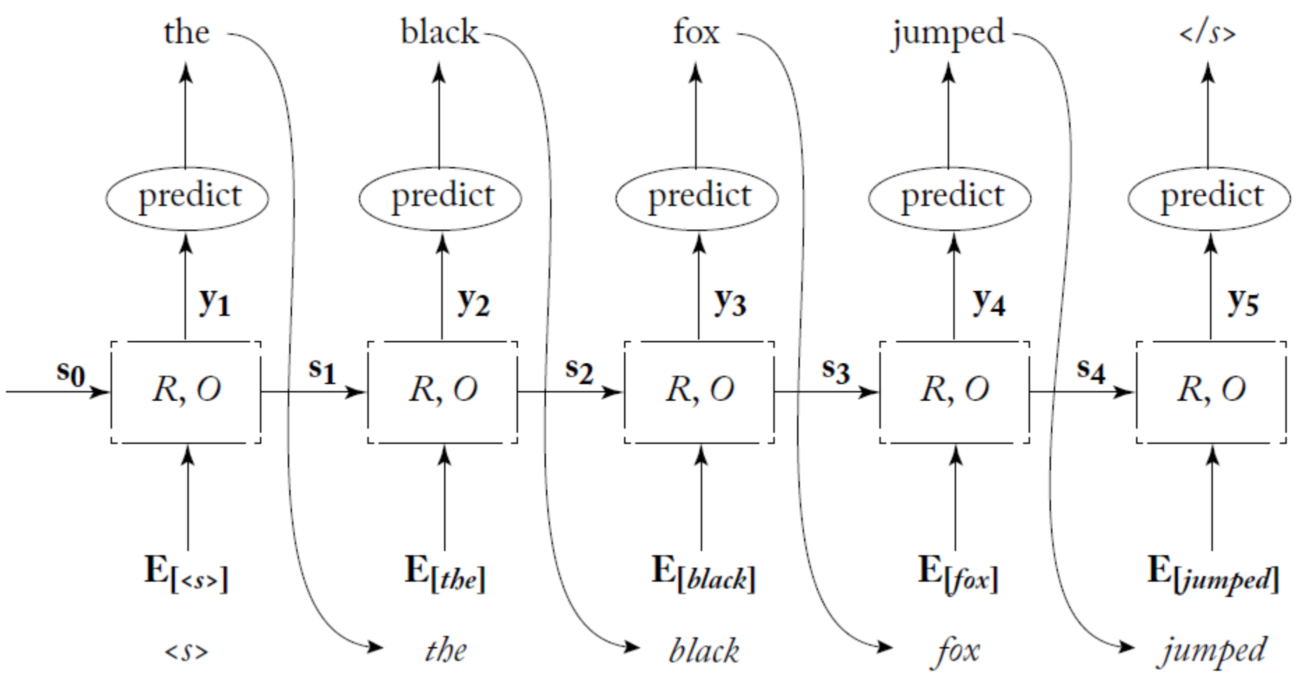


RNN Language Model

- Categorical cross-entropy loss (negative log likelihood)
- $L = -\sum_t \log_2(\hat{y}_t)$
- RNN LMs are better language models (have lower perplexity) compared to n-gram LMs

RNN Generators

Sequence generation



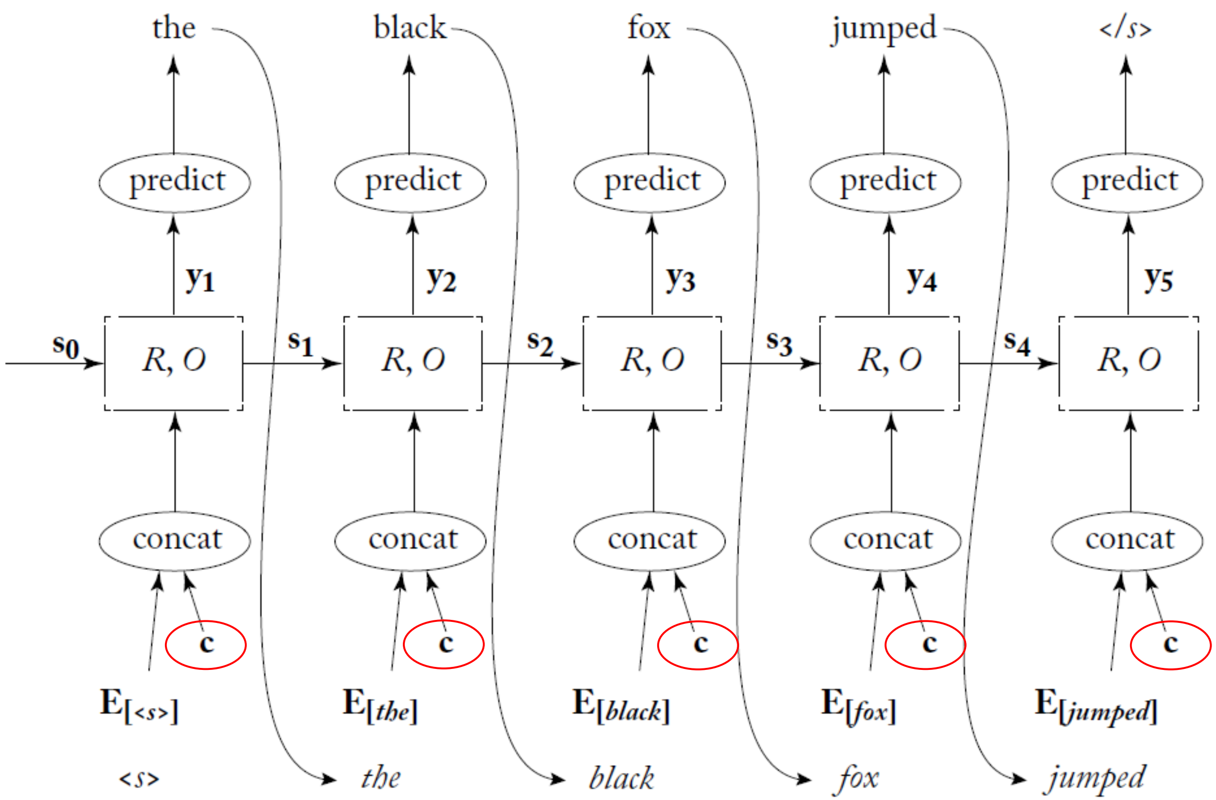
RNN Generators

$$\boldsymbol{s}_j = R(\boldsymbol{s}_{j-1}, \hat{\boldsymbol{t}}_j)$$

$$\boldsymbol{y}_j = O(\boldsymbol{s}_j)$$

$$p(\hat{t}_{j+1} | \hat{t}_{1:j}) = \mathrm{softmax}(\mathrm{MLP}(\boldsymbol{y}_j))$$

Conditioned Generation



Conditioned Generation

$$\boldsymbol{s}_j = R(\boldsymbol{s}_{j-1}, [\hat{\boldsymbol{t}}_j; \boldsymbol{c}])$$

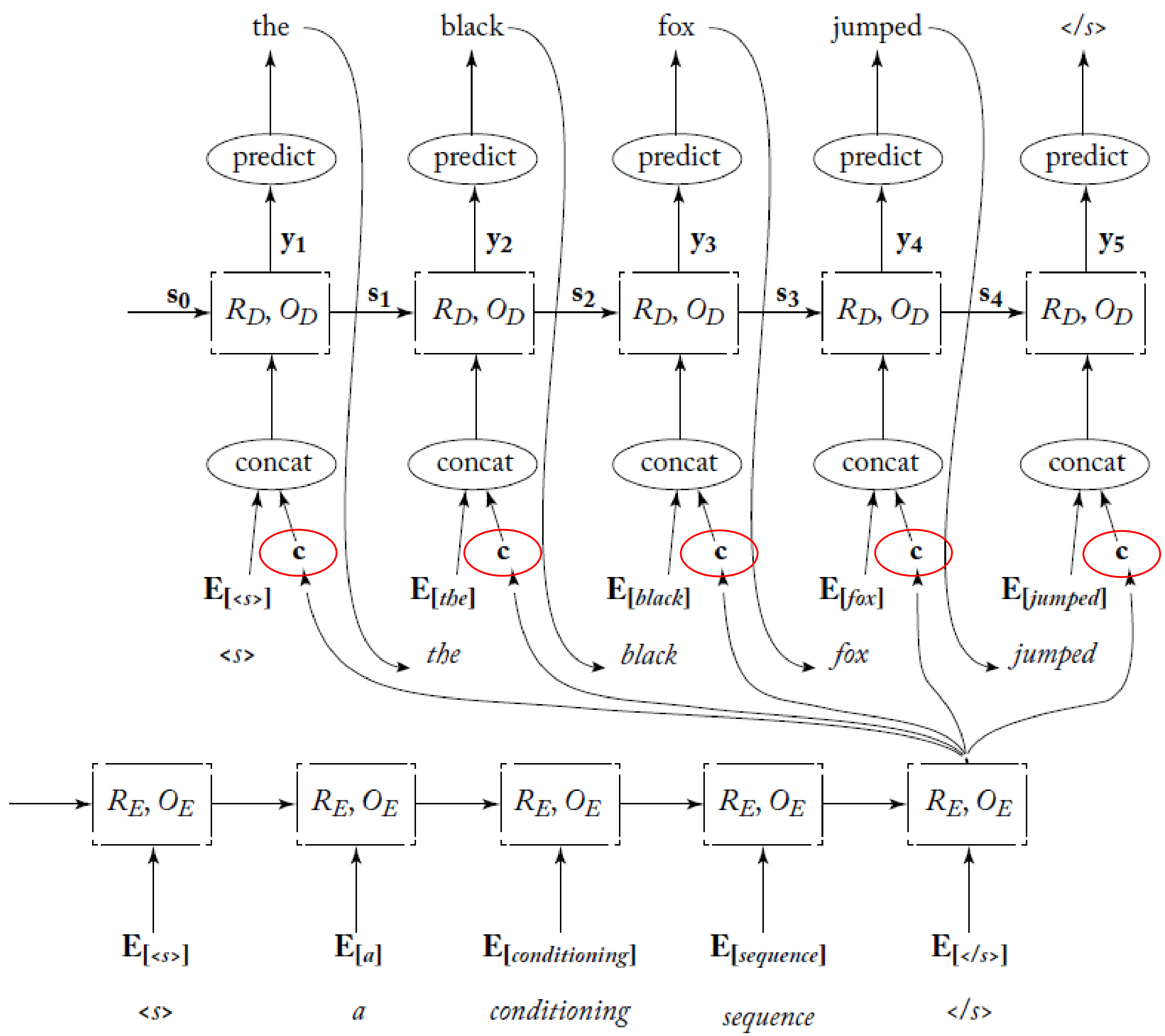
$$\boldsymbol{y}_j = O(\boldsymbol{s}_j)$$

$$p(\hat{t}_{j+1} | \hat{t}_{1:j}, c) = \text{softmax}(\text{MLP}(\boldsymbol{y}_j))$$

Sequence to Sequence Model

- Aka encoder-decoder model
- Application: neural machine translation, grammatical error correction
  - Translate a source sentence $\boldsymbol{x}_{\mathbf{1}:\boldsymbol{n}}$ into a target sentence $\boldsymbol{t}_{\mathbf{1}:\boldsymbol{m}}$
- Encoder: $\boldsymbol{c} = \mathrm{RNN}_{\mathrm{enc}}(\boldsymbol{x}_{\mathbf{1}:\boldsymbol{n}})$
- Decoder: a conditioned generator RNN
- The whole model (encoder and decoder) is trained end-to-end

Sequence to Sequence Model



Seq2Seq Model with Attention

$$\boldsymbol{s}_j = R_{\text{dec}}(\boldsymbol{s}_{j-1}, [\hat{\boldsymbol{t}}_j; \boldsymbol{c}^j])$$

$$\boldsymbol{y}_j = O_{\text{dec}}(\boldsymbol{s}_j)$$

$$p(\hat{t}_{j+1} | \hat{t}_{1:j}, \boldsymbol{x}_{1:n}) = \text{softmax}(\text{MLP}^{\text{out}}(\boldsymbol{y}_j))$$

Seq2Seq Model with Attention

$$\boldsymbol{c}^{j} = \sum_{i=1}^{n} \boldsymbol{\alpha}_{[i]}^{j} \cdot \boldsymbol{c}_i \qquad \longleftarrow \text{attend}$$

$$\boldsymbol{c}_{1:n} = \text{biRNN}_{\text{enc}}^{*}(\boldsymbol{x}_{1:n})$$

$$\bar{\boldsymbol{\alpha}}_{[i]}^{j} = \boldsymbol{v} \cdot \tanh([\boldsymbol{s}_{j-1}; \boldsymbol{c}_i]\boldsymbol{U} + \boldsymbol{b}) \qquad \longleftarrow \text{MLP}^{\text{att}}$$

$$\boldsymbol{\alpha}^{j} = \text{softmax}(\bar{\boldsymbol{\alpha}}_{[1]}^{j}, \ldots, \bar{\boldsymbol{\alpha}}_{[n]}^{j})$$

Seq2Seq Model with Attention

- Attention weights $\boldsymbol{\alpha}_{[i]}^{j}$ reveal which parts i of the source sentence the decoder finds relevant at output step j.
- Improved interpretability