Probabilistic language models

Сергей Аксенов

Высшая Школа Экономики

10 октября 2022 г.

Today

Overview

Языковая модель

1. Вычислить вероятность последовательности слов:

$$P(w_1, w_2, \ldots, w_n)$$

2. Предсказать следующее слово:

$$P(w_n | w_1, w_2, \ldots, w_{n-1})$$

Языковые модели используются:

- Машинный перевод: выбрать
- Проверка правописания: найти слово с ошибкой
- Распознавание речи: выбрать наилучшую транскрипцию
- Автодополнение
- Генерация текстов
- Суммаризация

Markov assumptions

1. Chain rule:

$$P(W) = P(w_1, w_2, \ldots, w_n) = \prod_i P(w_i | w_1, \ldots, w_{i-1})$$

2. Максимальное правдоподобие:

$$P(w_i | w_1, \ldots, w_{i-1}) = \frac{\text{count}(w_1, w_2, \ldots, w_i)}{\text{count}(w_1, \ldots, w_{i-1})}$$

3. Марковское предположение (k-го порядка):

$$P(w_i | w_1, \ldots, w_{i-1}) \approx P(w_i | w_{i-k}, \ldots w_{i-1})$$

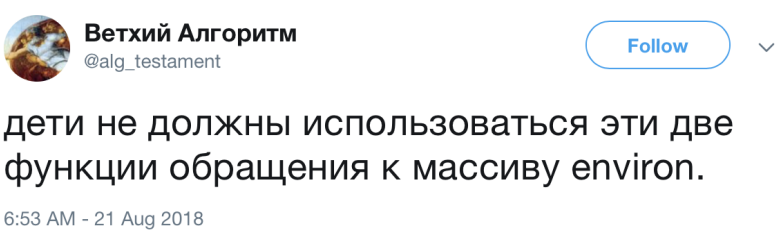
Модель n-грам

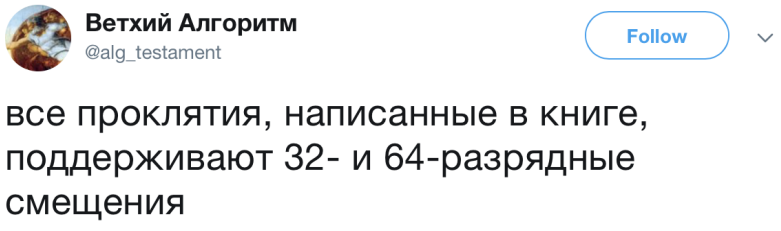
1. Модель униграм: $P(W) = P(w_1, \ldots, w_n) \approx \prod_i P(w_i)$
2. Модель биграм: $P(W) = P(w_1, \ldots, w_n) \approx \prod_i P(w_i|w_{i-1})$
3. Перплексия: $PP(W) = P(w_1, w_2, \ldots w_n)^{-\frac{1}{N}}$
   Чем ниже перплексия, тем лучше модель предсказывает новое слово
4. Сглаживание: $P(w_i|w_1, \ldots, w_{i-1}) = \frac{\text{count}(w_1,w_2,\ldots,w_i)+1}{\text{count}(w_1,\ldots,w_{i-1})+\alpha|V|}$, где $|V|$-размер словаря
5. Интерполяция: $\hat{P}(w_i|w_{i-1}) = \lambda P_{MLE}(w_i|w_{i-1}) + (1-\lambda)P_{MLE}(w_i)$

Модель n-грам для генерации текстов

На основе слова w_i:

1. Выбираем наиболее вероятное слово w_{i+i}
2. Случайно выбираем слово из вероятностного распределения для следующего слова





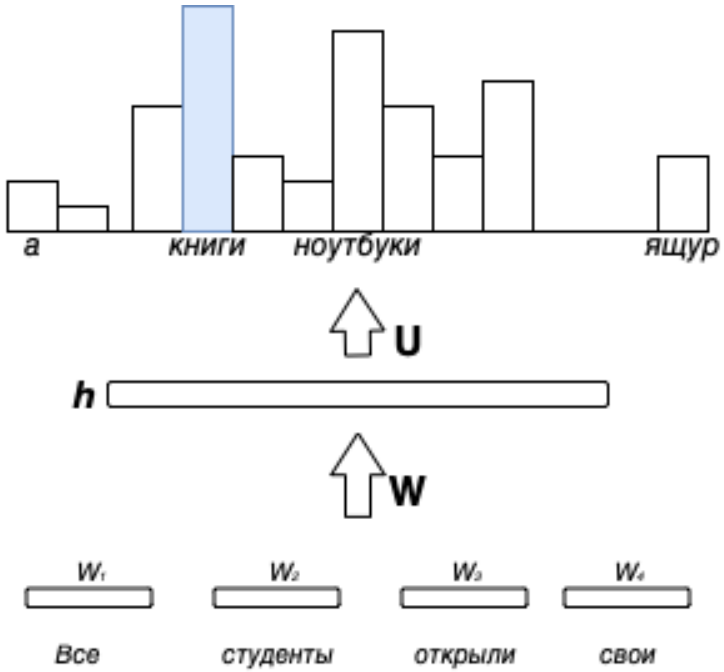
`https://twitter.com/alg_testament`

Нейросетевая языковая модель

- **Модель n-грам**: по $n-1$ слову предсказываем n-ное слово.
- **Вход**: $w_1, .., w_{n-1}$
- **Выход**: w_n

Все студенты открыли свои ***книги***

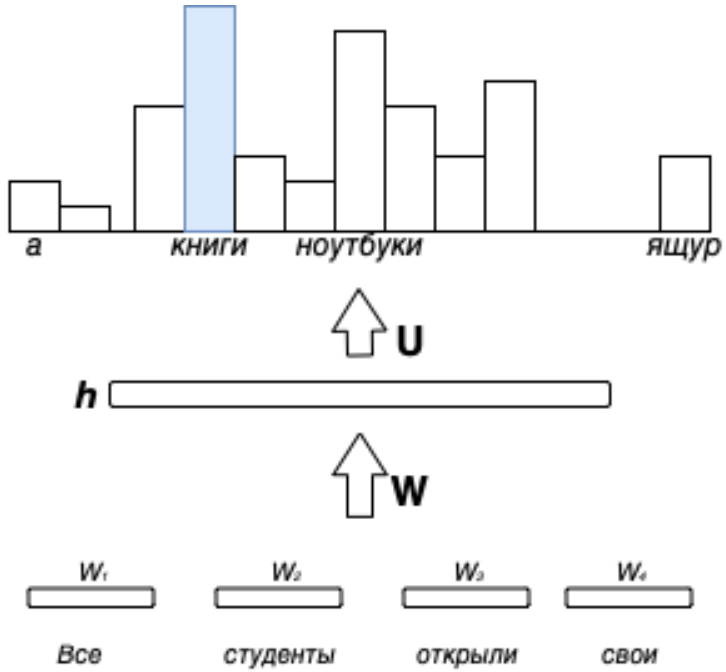
- **Вход:** *Все студенты открыли свои*
- **Выход:** ***книги***



Нейросетевая языковая модель

- **Первый слой**: каждому слову ставим в соответствие d-мерный вектор f_i.
- Конкатенируем векторы эмбеддингов, получаем вектор $x = [f_1, ..f_{n-1}]$. Его размерность $(n-1)d$
- **Скрытый слой**: $h = \sigma(Wx + b)$
- W, b - обучаемые параметры
- Вероятность следующего слова из y:

$$y = softmax(Uh + b)$$



Нейросетевая языковая модель

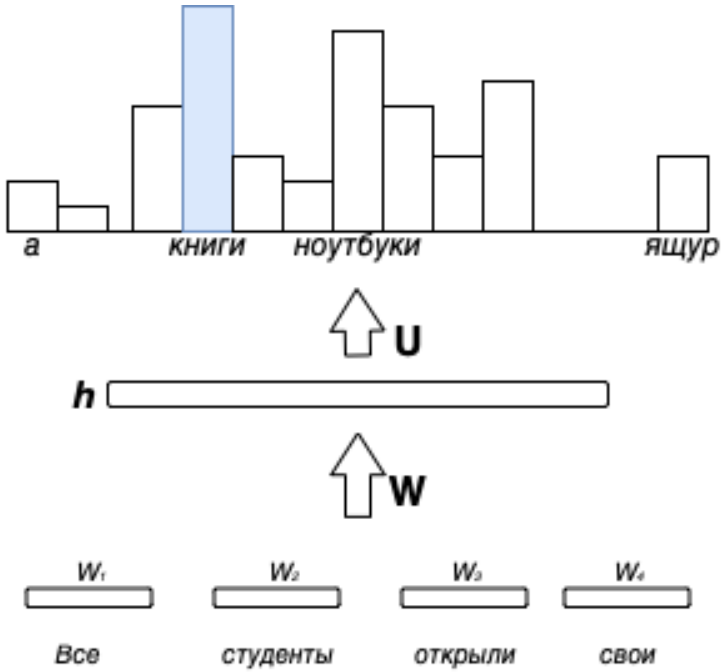
Обучение

- Разбиваем корпус на T n-грамм
- $n-1$ слов контекста подаем на вход сети, предсказываем n-ое слово

Функция потерь

- Отрицательный логарифм правдоподобия

$$L = -\frac{1}{T} log(P(y))$$



Today

1. Language models
2. Sequence modelling
   - Definition
   - Training
   - Gated architectures
3. RNN generators

Последовательные данные

1. Временные ряды
   - Финансовые данные: курс акций, облигаций, валют
   - Медицинские данные: пульс, уровень сахара
2. Текст и речь: распознавание речи, генерация текста
3. Пространственно-временные данные
   - Беспилотники
   - Тектоническая активность
4. Физические данные
5. etc.

Sequence modelling I

Классификация последовательностей

1. $x = x_1, x_2, \ldots, x_n$, $x_i \in V$ - объекты
2. $y \in \{1, \ldots, L\}$ - метки классов
3. $\{(x^{(1)}, y_1), (x^{(2)}, y_2), \ldots, (x^{(m)}, y_m)\}$ – данные для обучения

Задача классификации: $\gamma : x \to y$

1. Activity recognition: x – пульс, y – активность (ходьба, бег, покой)
2. Сентимент-анализ: x – sentence, y – сентимент (позитивный, негативный)
3. Биржевая торговля: x – курс акции, y – действие (продать, купить, ничего не делать)

Sequence modelling II

Разметка последовательностей

1. $x = x_1, x_2, \ldots, x_n$, $x_i \in V$ - объекты
2. $y = y_1, y_2, \ldots, y_n$, $y_i \in \{1, \ldots, L\}$ - метки классов
3. $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$ – training data
4. Экспоненциально большое число возможных вариантов : если $\texttt{length}(x) = n$, то существует L^n возможных вариантов разметки

Задача классификации: $\gamma : x \to y$

1. POS-tagging: x – слово, y – часть речи (глагол, прилагательное, etc.)
2. Разметка генома: x – ДНК, y – ген
3. Распознавание именованных сущностей

Sequence labelling tasks

POS tagging and Named Entity Recognition

X (words)	the	cat	sat	on	a	mat
Y (tags)	DET	NOUN	VERB	PREP	DET	NOUN

Таблица: POS tagging

Alex	is	going	to	Los	Angeles
B-PER	O	O	O	B-LOC	I-LOC

Таблица: NER (IOB2)

Alex	travels	with	Marty	A.	Rick	to	NY	city
S-PER	O	O	B-PER	I-PER	E-PER	O	B-LOC	E-LOC

Таблица: NER (IOBES)

Sequence modelling III

Преобразование последовательностей

1. $x = x_1, x_2, \ldots, x_n$, $x_i \in V_{source}$ - объекты
2. $y = y_1, y_2, \ldots, y_n$, $y_i \in V_{target}$ - объекты
3. $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$ – данные для обучения
4. $x^{(1)}$, $y^{(1)}$ имеют разную длину

Задача преобразования последовательности: $x_{source} \rightarrow y_{target}$

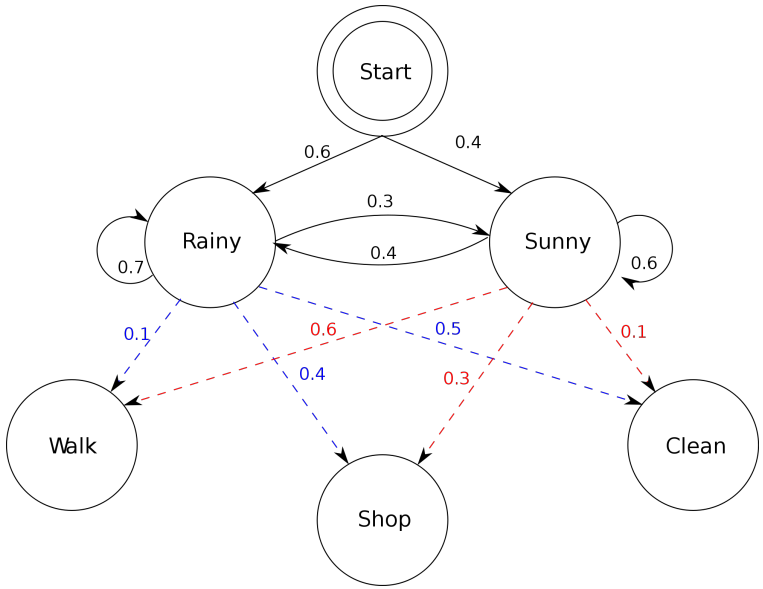
1. Машинный перевод: x – предложение на русском, y – предложение на английском
2. Распознавание речи: x – речь, y – текст
3. Чат-боты: x – вопрос, y – ответ

Traditional ML approaches to sequence modeling

- Hidden Markov Models (HMM)
- Conditional Random Fields (CRF)
- Local classifier: for each x define features, based on x_{-1}, x_{+1}, etc, and perform classification n times

Problems:

1. Markov assumption: fixed length history
2. Computation complexity

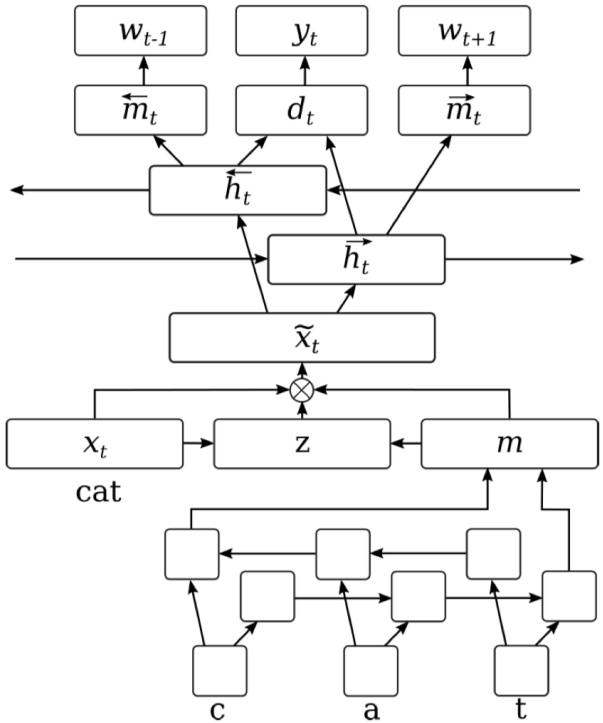


DL approaches to sequence modeling

- Neural networks
- Recurrent neural network and its modifications: LSTM, GRU, Highway
- 2D Convolutional Neural Network
- Transformer
- Pointer network

Problems:

1. Training time
2. Amount of training data



Recurrent neural network

- Input: sequence of vectors
- $x_{1:n} = x_1, x_2, \ldots, x_n$, $x_i \in \mathbb{R}^{d_{in}}$
- Output: a single vector
  $y_n = RNN(x_{1:n})$, $y_n \in \mathbb{R}^{d_{out}}$
- For each prefix $x_{i:j}$ define an output vector y_i:
  $y_i = RNN(x_{1:i})$
- RNN^* is a function returning this sequence for input sequence $x_{1:n}$:
  $y_{1:n} = RNN^*(x_{1:n})$, $y_i \in \mathbb{R}^{d_{out}}$

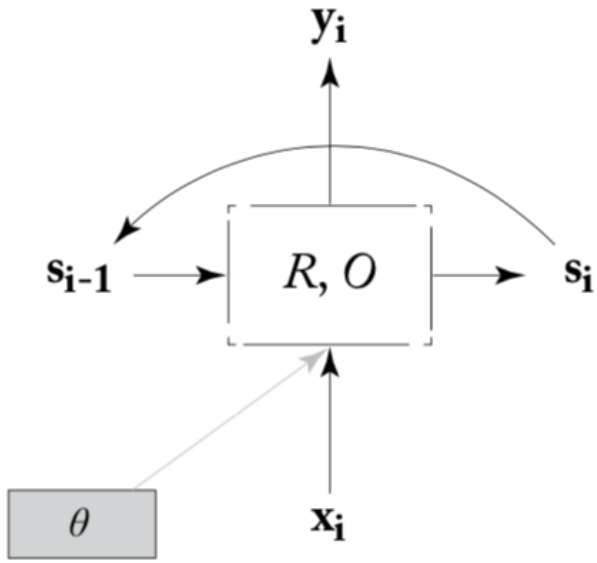


Figure: Goldberg, Yoav. Neural network methods for natural language processing

Sequence modelling with RNN

1. Sequence classification
   Put a dense layer on top of RNN to predict the desired class of the sequence after the whole sequence is processed

$$p(l_j|x_{1:n}) = \texttt{softmax}(RNN(x_{1:n}) \times W + b)_{[j]}$$

2. Sequence labelling
   Produce an output y_i for each input RNN reads in. Put a dense layer on top of each output to predict the desired class of the input

$$p(l_j|x_j) = \texttt{softmax}(RNN(x_{1:j}) \times W + b)_{[j]}$$

More details on RNN

- $RNN^*(x_{1:n}, s_0) = y_{1:n}$
- $y_i = O(s_i)$ – simple activation function
- $s_i = R(s_{i-1}, x_i)$, where R is a recursive function, s_i is a state vector
- s_0 is initialized randomly or is a zero vector
- $x_i \in \mathbb{R}^{d_{in}}$, $y_i \in \mathbb{R}^{d_{out}}$, $s_i \in \mathbb{R}^{f(d_{out})}$
- θ – shared weights

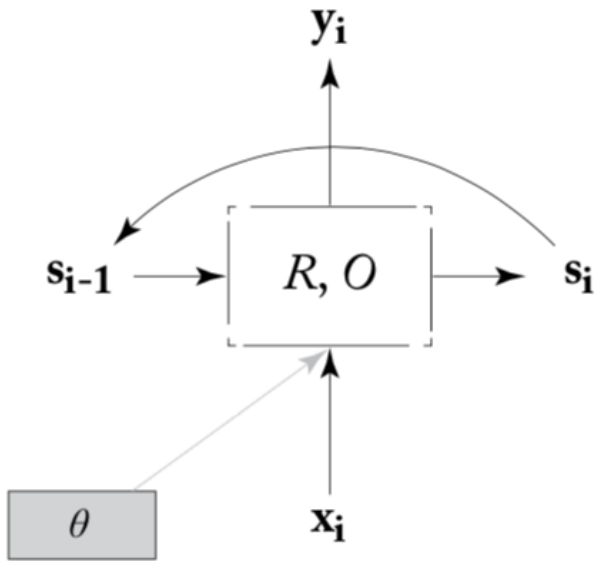


Figure: Goldberg, Yoav. Neural network methods for natural language processing

More details on RNN

- $s_i = R(x_i, s_{i-1}) = g(s_{i-1}W^s + x_i W^x + b)$
- $y_i = O(s_i) = s_i$
- $y_i, s_i, b \in \mathbb{R}^{d_{out}}$, $x_i \in \mathbb{R}^{d_{in}}$
- $W^x \in \mathbb{R}^{d_{in} \times d_{out}}$, $W^s \in \mathbb{R}^{d_{out} \times d_{out}}$

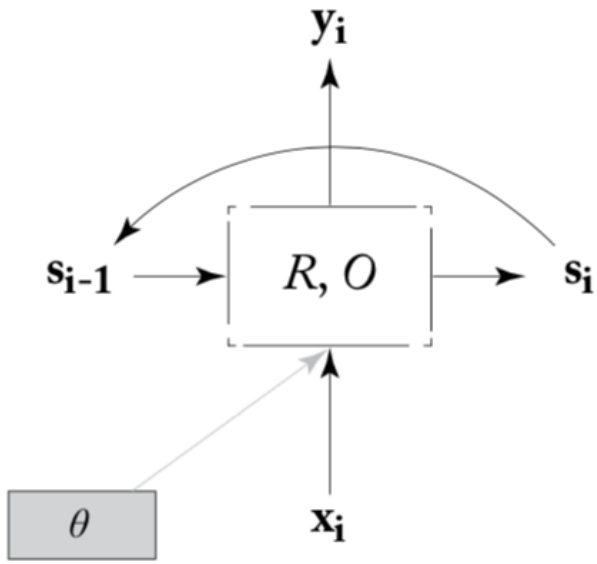
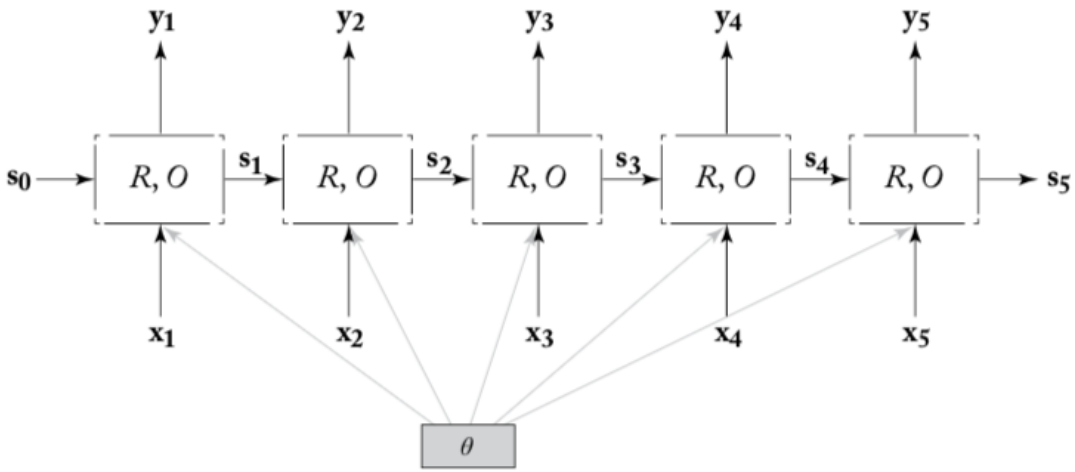


Figure: Goldberg, Yoav. Neural network methods for natural language processing

RNN unrolled

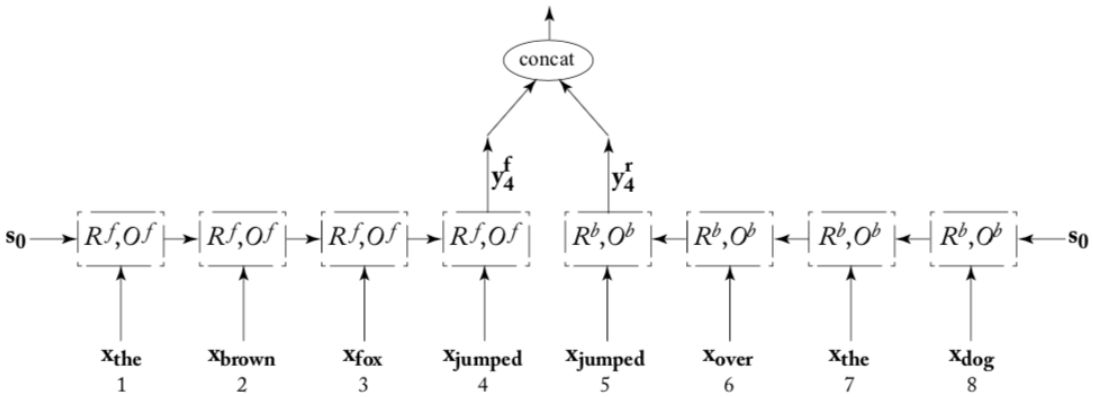


$$s_4 = R(s_3, x_4) = R(R(s_2, x_3), x_4) = R(R(R(s_1, x_2), x_3), x_4) =$$
$$= R(R(R(R(s_0, x1), x_2), x_3), x_4)$$

Figure: Goldberg, Yoav. Neural network methods for natural language processing

Bidirectional RNN (Bi-RNN)

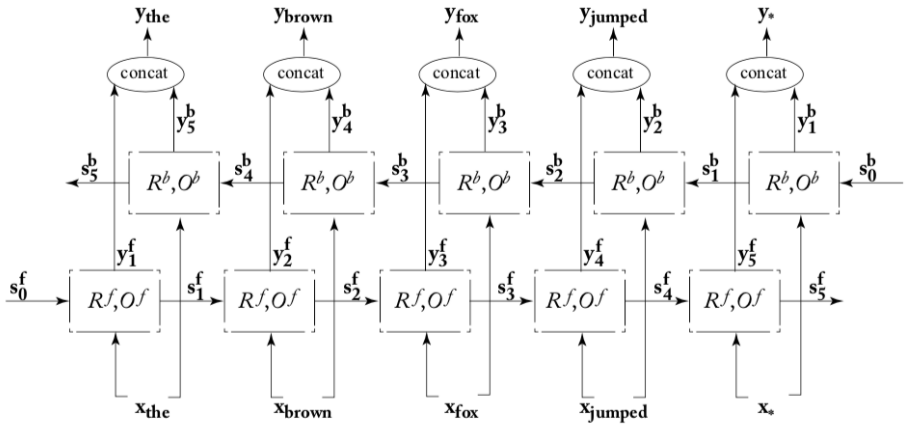
The input sequence can be read from left to right and from right to left. Which direction is better?



$$biRNN(x_{1:n}, i) = y_i = [RNN^f(x_{1:i}); RNN^r(x_{n:i})]$$

Figure: Goldberg, Yoav. Neural network methods for natural language processing

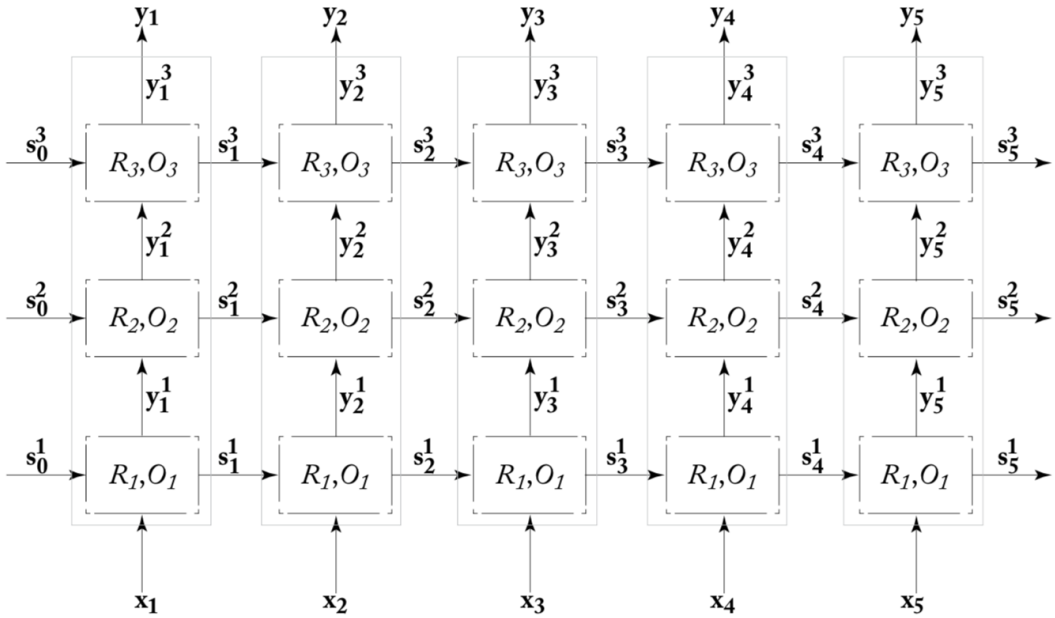
Bi-RNN



$$biRNN^*(x_{1:n}, i) = y_{1:n} = biRNN(x_{1:n}, 1) \dots biRNN(x_{1:n}, n)$$

Figure: Goldberg, Yoav. Neural network methods for natural language processing

Multilayer RNN



Connections between different layers are possible too: $y_1^2 = \texttt{concat}(x_1, y_1^1)$

Figure: Goldberg, Yoav. Neural network methods for natural language processing

Sequence classification

- $\hat{y}_n = O(s_n)$
- $\texttt{prediction} = MLP(\hat{y}_n)$
- Loss: $L(\hat{y}_n, y_n)$
- L can take any form: cross entropy, hinge, margin, etc.

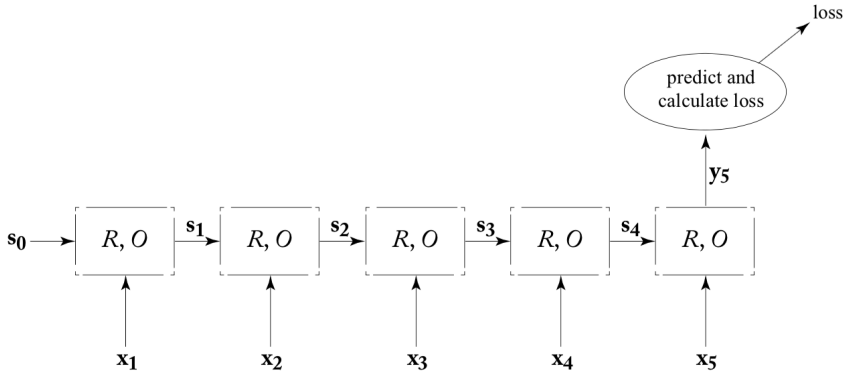


Figure: Goldberg, Yoav. Neural network methods for natural language processing

Sequence labelling

- Output $\hat{t}_i$ for each input $x_{1,i}$
- Local loss: $L_{local}(\hat{t}_i, t_i)$
- Global loss: $L(\hat{t}_n, t_n) = \sum_i L_{local}(\hat{t}_i, t_i)$
- L can take any form: cross entropy, hinge, margin, etc.

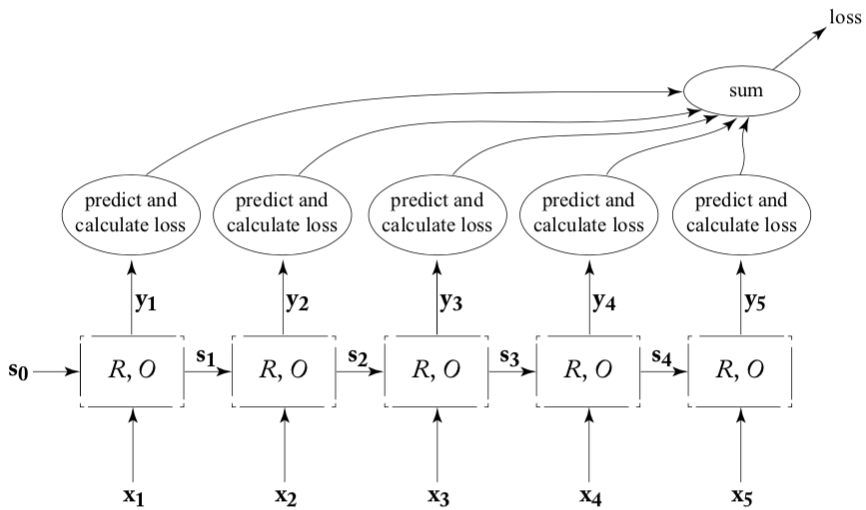
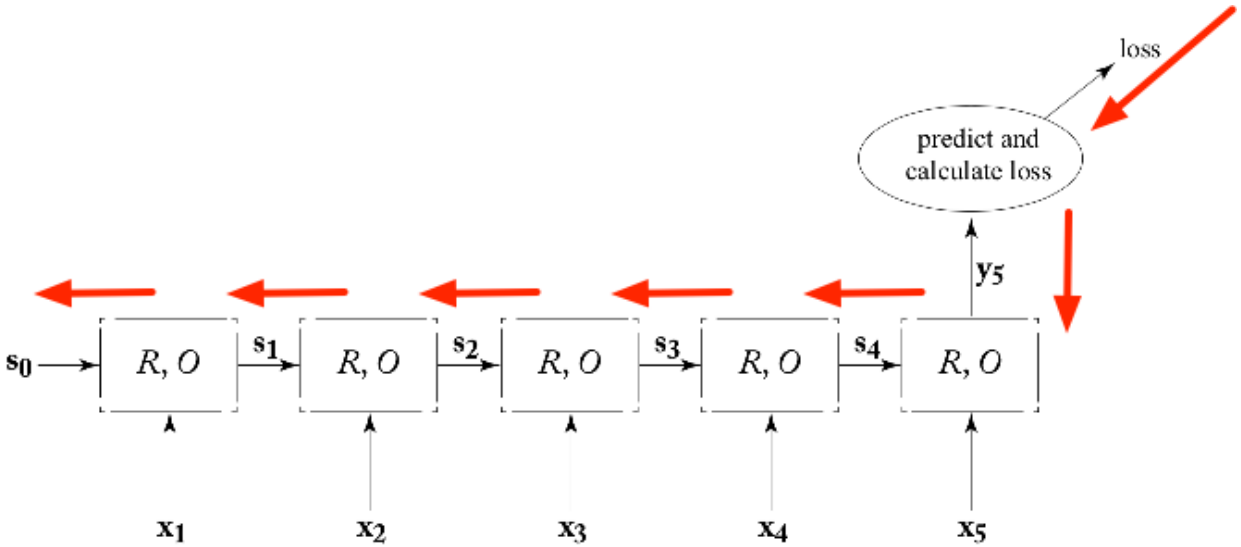


Figure: Goldberg, Yoav. Neural network methods for natural language processing

Backpropogation through time



$$s_i = R(x_i, s_{i-1}) = g(s_{i-1}W^s + x_i W^x + b)$$

Chain rule: $\frac{\partial L}{\partial w} = \frac{\partial L}{\partial p(\hat{y_5})} \frac{\partial p(\hat{y_5})}{\partial s_4} \left(\frac{\partial s_4}{\partial w} + \frac{\partial s_4}{\partial s_3} \frac{\partial s_3}{\partial w} + \frac{\partial s_4}{\partial s_3} \frac{\partial s_3}{\partial s_2} \frac{\partial s_2}{\partial s_w} + \ldots \right)$

Figure: Goldberg, Yoav. Neural network methods for natural language processing

Vanishing gradient problem

Chain rule: $\frac{\partial L}{\partial w} = \frac{\partial L}{\partial p(\hat{y}_5)} \frac{\partial p(\hat{y}_5)}{\partial s_4} \left(\frac{\partial s_4}{\partial w} + \frac{\partial s_4}{\partial s_3} \frac{\partial s_3}{\partial w} + \frac{\partial s_4}{\partial s_3} \frac{\partial s_3}{\partial s_2} \frac{\partial s_2}{\partial s_w} + \ldots \right)$

g – sigmoid

1. Many sigmoids near 0 and 1
   - Gradients $\to 0$
   - Not training for long term dependencies
2. Many sigmoids > 1
   - Gradients $\to +\text{inf}$
   - Not training again

Solution: gated architectures (LSTM and GRU)

Controlled memory access

- Entire memory vector is changed: $s_{i+1} = R(x_i, s_i)$
- Controlled memory access: $s_{i+1} = g \odot R(x_i, s_i) + (1 - g)s_i$
  $g \in [0, 1]^d, s, x \in \mathbb{R}^d$
- Differential gates: $\sigma(g), g' \in \mathbb{R}^d$
- This controllable gating mechanism is the basis of the LSTM and the GRU architectures

Long short term memory

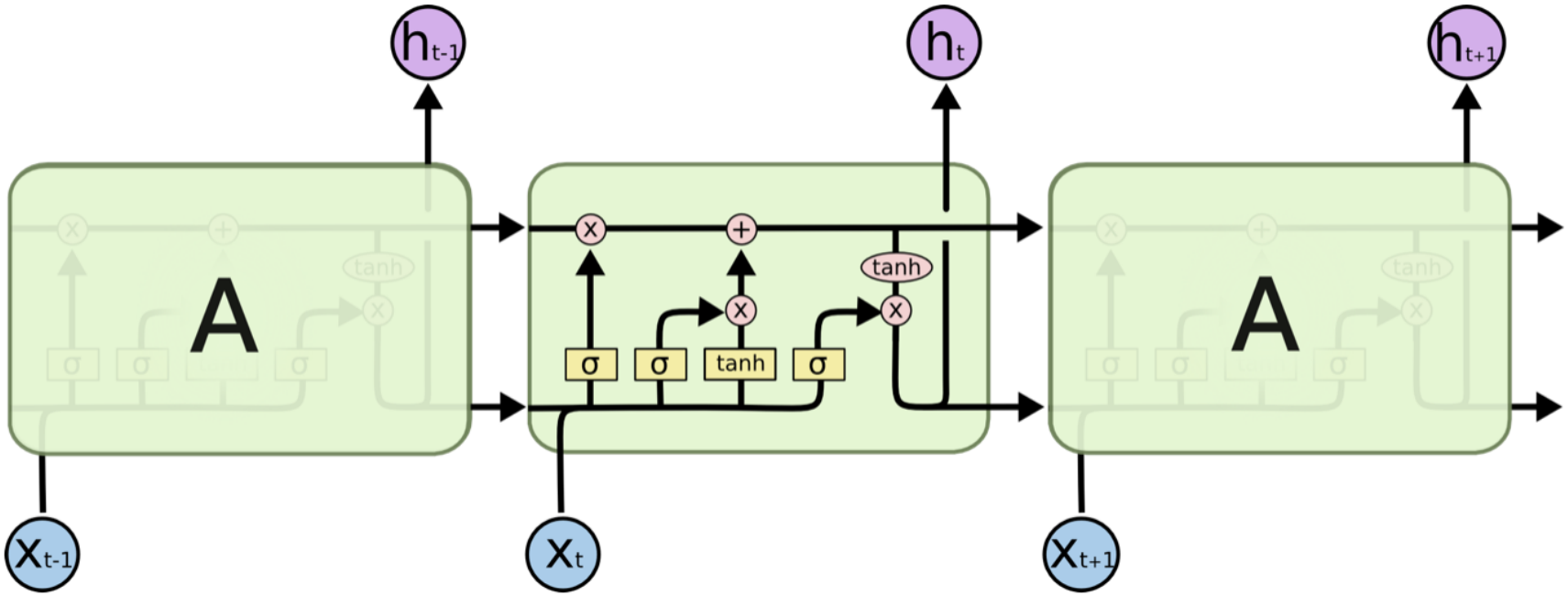
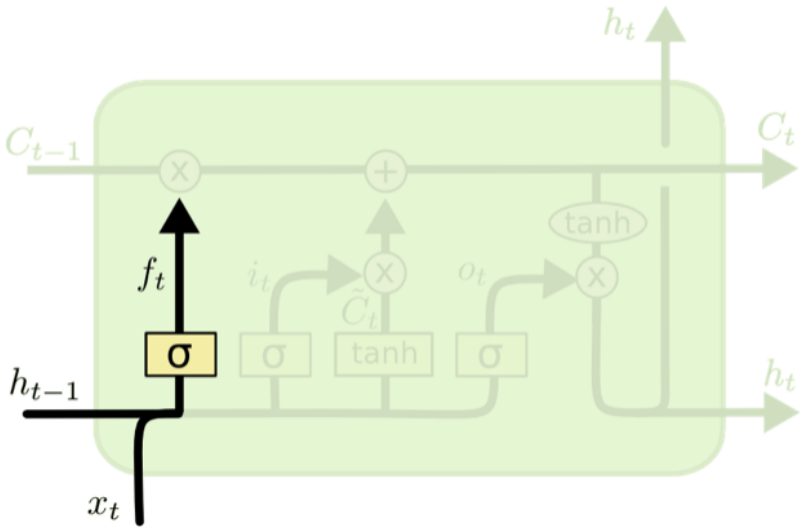


Figure: colahbloghttp://colah.github.io/posts/2015-08-Understanding-LSTMs/

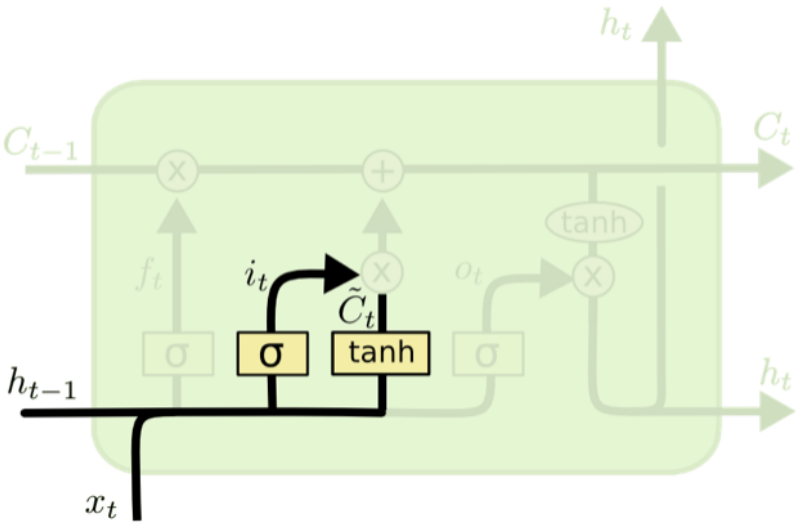
Long short term memory



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long short term memory

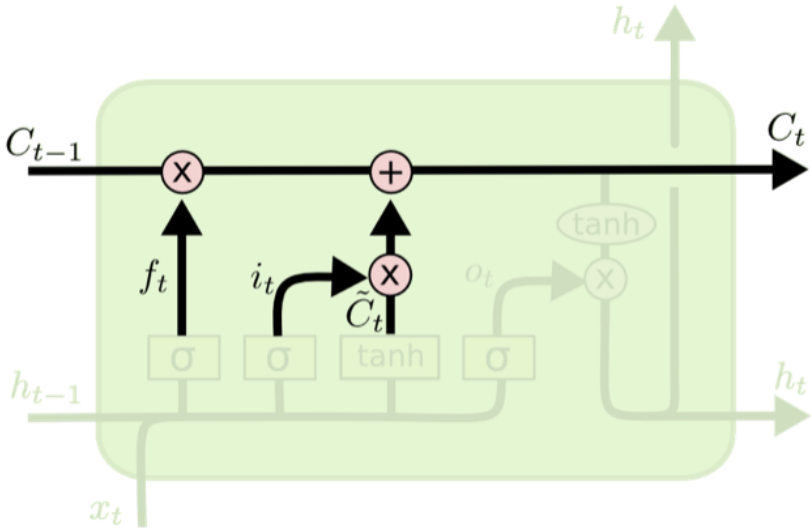


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure: colahbloghttp://colah.github.io/posts/2015-08-Understanding-LSTMs/

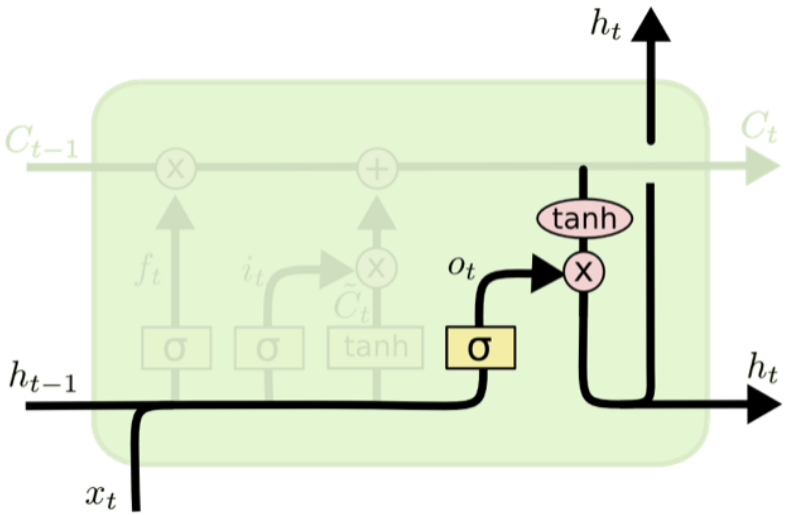
Long short term memory



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure: colahbloghttp://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long short term memory

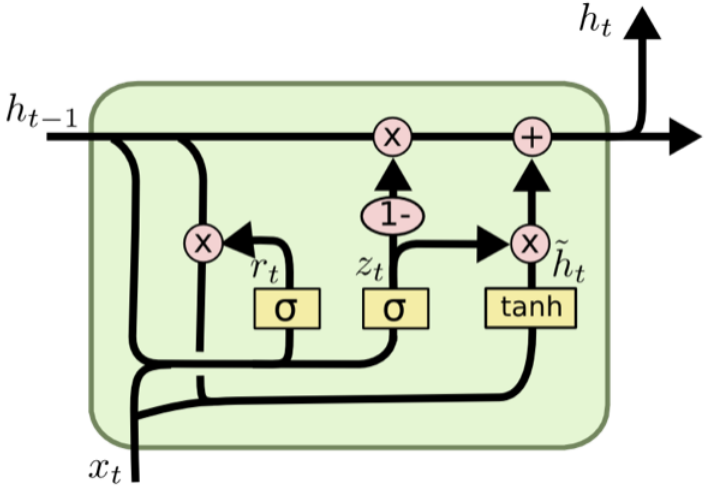


$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Figure: colahbloghttp://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated recurrent unit



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

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

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure: colahbloghttp://colah.github.io/posts/2015-08-Understanding-LSTMs/

Today

Language model

1. Compute the probability of a sequence of words:

$$P(w_1, w_2, \ldots, w_n)$$

2. Predict next word:

$$P(w_n | w_1, w_2, \ldots, w_{n-1})$$

Perplexity

$$2^{H(p)} = 2^{\frac{1}{|V|} - \sum_x \log_2 p(x)}$$

Sequence generation

Teacher forcing: $x := < s > x, y := x < /s >$

$x : < s > x_1 x_2 \ldots x_n$

$y : x_1 x_2 \ldots x_n < /s >$

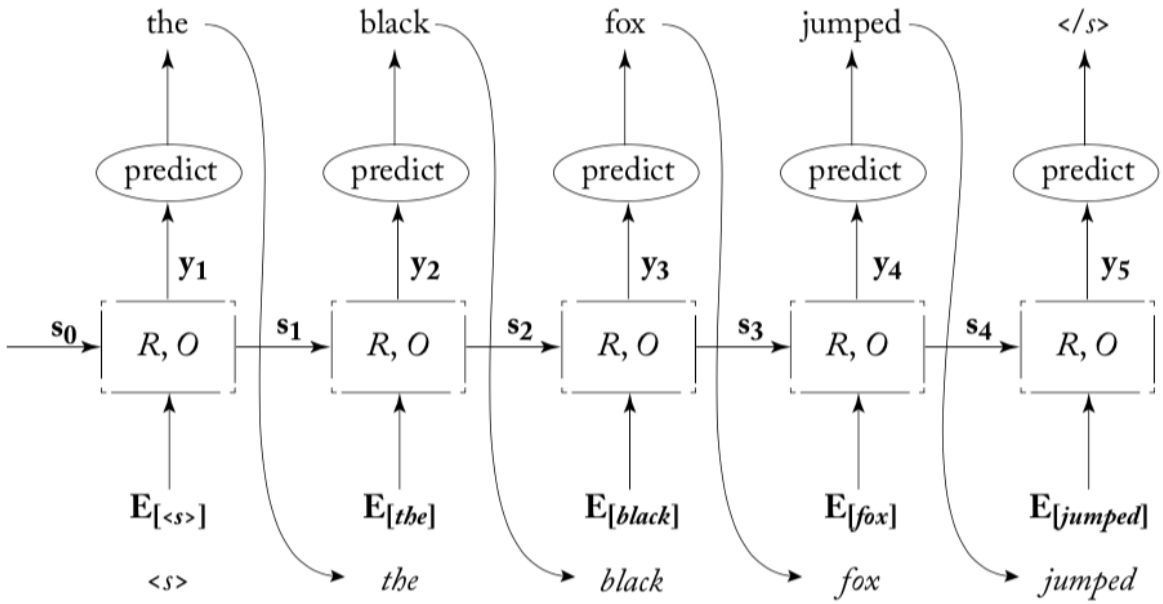


Figure: Goldberg, Yoav. Neural network methods for natural language processing