Overview of Natural Language Processing Part II & ACS L390

Lecture 6: Incrementality

Weiwei Sun

Department of Computer Science and Technology
University of Cambridge

Michaelmas 2024/25

Incrementality in human language comprehension

Self-paced reading: you press a button for each word

I

Incrementality in human language comprehension

Self-paced reading: you press a button for each word

convinced

Incrementality in human language comprehension

Self-paced reading: you press a button for each word

her

Incrementality in human language comprehension

Self-paced reading: you press a button for each word

children

Incrementality in human language comprehension

Self-paced reading: you press a button for each word

are

Incrementality in human language comprehension

Self-paced reading: you press a button for each word

noisy.

Incrementality in human language comprehension

Self-paced reading: you press a button for each word

I convinced her children are noisy.

at which word, do you stop for a significantly longer time?

Incrementality in human language comprehension

Self-paced reading: you press a button for each word

I convinced her children are noisy.

at which word, do you stop for a significantly longer time?

Linguistic performance

- Left-to-right, word-by-word
- Partially parsed results (history) constrain parsing of subsequent words

Lecture 6: Incrementality

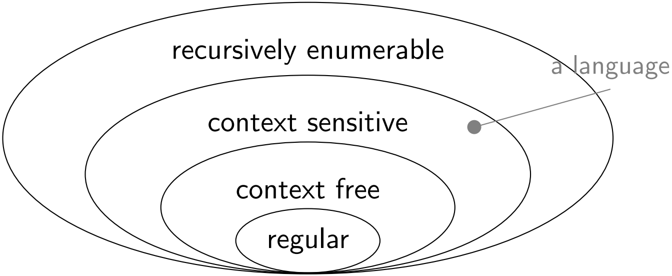
1. Rethink part-of-speech tagging
2. Sequence-to-sequence
3. Recurrent Neural Networks

Rethink Part-of-Speech Tagging

Chomsky Hierarchy

Grammar	Languages	Production rules
Type-0	Recursively enumerable	$\alpha \rightarrow \gamma$
Type-1	Context-sensitive	$\alpha A \beta \rightarrow \alpha \gamma \beta$
Type-2	Context-free	$A \rightarrow \gamma$
Type-3	Regular	$A \rightarrow a$
		$A \rightarrow aB$

$a \in N;\ \alpha, \beta, \gamma \in (N \cup \Sigma)^*$



An example

Max bit the cat [which chased the mouse [which died]].

A toy grammar

- VP$\rightarrow$ bit$|$chased$|\ldots$ DP
- VP$\rightarrow$ died
- DP$\rightarrow$ the$|$a$|$this$|\ldots$ NP
- NP$\rightarrow$ dog$|$cat$|$mouse$|\ldots$ RC
- RC$\rightarrow$ which$|$that$|\ldots$ VP

An example

Max bit the cat [which chased the mouse [which died]].

A toy grammar

- VP$\rightarrow$ bit|chased|... DP
- VP$\rightarrow$ died
- DP$\rightarrow$ the|a|this|... NP
- NP$\rightarrow$ dog|cat|mouse|... RC
- RC$\rightarrow$ which|that|... VP

<u>VP</u>

An example

Max bit the cat [which chased the mouse [which died]].

A toy grammar

- VP$\rightarrow$ bit|chased|... DP
- VP$\rightarrow$ died
- DP$\rightarrow$ the|a|this|... NP
- NP$\rightarrow$ dog|cat|mouse|... RC
- RC$\rightarrow$ which|that|... VP

$\underline{\text{VP}}$
$\Rightarrow$ *bit* $\underline{\textit{DP}}$

An example

Max bit the cat [which chased the mouse [which died]].

A toy grammar

- VP$\to$ bit|chased|... DP
- VP$\to$ died
- DP$\to$ the|a|this|... NP
- NP$\to$ dog|cat|mouse|... RC
- RC$\to$ which|that|... VP

$\underline{\text{VP}}$

$\Rightarrow$ *bit* $\underline{\textit{DP}}$ $\Rightarrow$ bit *the* $\underline{\textit{NP}}$

An example

Max bit the cat [which chased the mouse [which died]].

A toy grammar

- VP$\rightarrow$ bit|chased|... DP
- VP$\rightarrow$ died
- DP$\rightarrow$ the|a|this|... NP
- NP$\rightarrow$ dog|cat|mouse|... RC
- RC$\rightarrow$ which|that|... VP

$\underline{\text{VP}}$
$\Rightarrow$ *bit* $\underline{\textit{DP}}$ $\Rightarrow$ bit *the* $\underline{\textit{NP}}$ $\Rightarrow$ bit the *cat* $\underline{\textit{RC}}$

An example

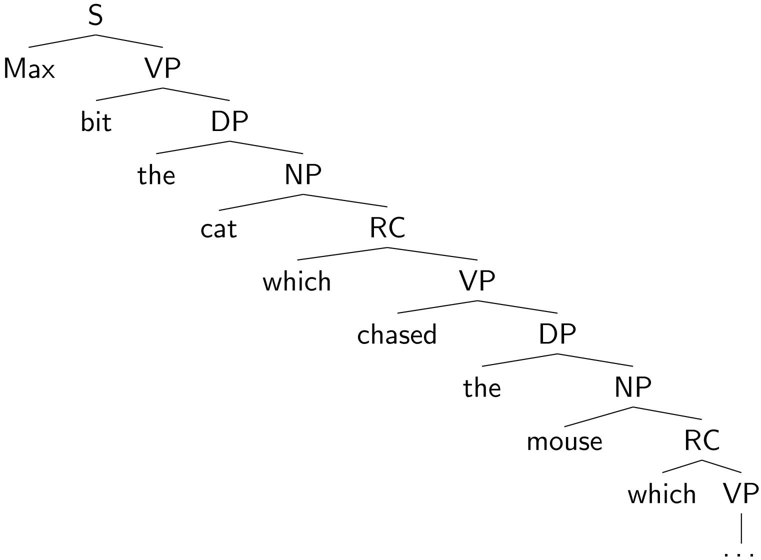
Max bit the cat [which chased the mouse [which died]].

A toy grammar

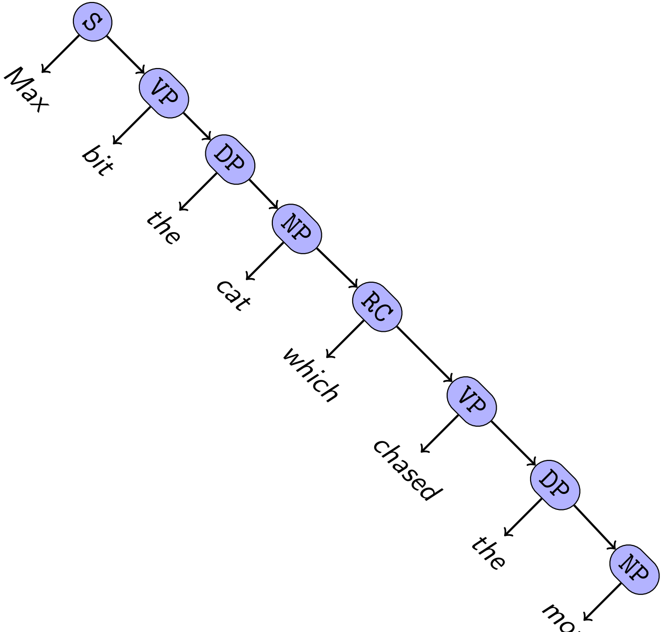
- VP$\rightarrow$ bit$|$chased$|\ldots$ DP
- VP$\rightarrow$ died
- DP$\rightarrow$ the$|$a$|$this$|\ldots$ NP
- NP$\rightarrow$ dog$|$cat$|$mouse$|\ldots$ RC
- RC$\rightarrow$ which$|$that$|\ldots$ VP

$\underline{\text{VP}}$
$\Rightarrow$ *bit* $\underline{\textit{DP}}$ $\Rightarrow$ bit *the* $\underline{\textit{NP}}$ $\Rightarrow$ bit the *cat* $\underline{\textit{RC}}$
$\Rightarrow$ bit the cat *which* $\underline{\textit{VP}}$

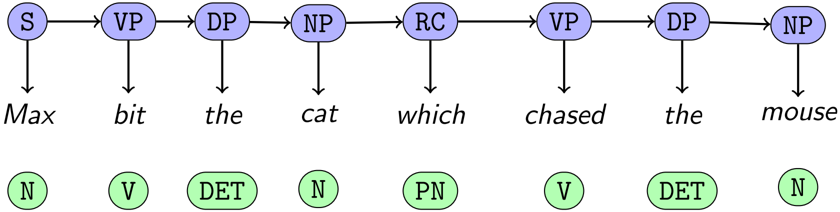
Finite state machines?



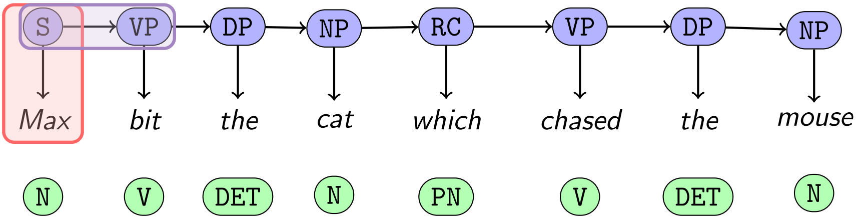
Finite state machines?



Word tagging is very powerful



Word tagging is very powerful



Generative models: Hidden Markov Models and PCFG

- $p_e(\textit{Max}|\text{S}) \times p_t(\text{VP}|\text{S})$
- $p(\text{S} \rightarrow \textit{Max}\ \text{VP})$

Probabilistic models for sequence pairs

- We have two sequences of random variables: $X_1, X_2, ..., X_n$ and $S_1, S_2, ..., S_n$
- Intuitively, each X_i corresponds to an observation and each S_i corresponds to an underlying state that generated the observation. Assume that each S_i is in $\{1, 2, ..., k\}$, and each X_i is in $\{1, 2, ..., o\}$.
- How do we model the joint distribution

$$P(X_1 = x_1, ..., X_n = x_n, S_1 = s_1, ..., S_n = s_n)$$

Hidden Markov Models

An HMM takes the following form

$$p(x_1 ... x_n, s_1 ... s_n; \theta) = p_t(s_1) \prod_{j=2}^{n} p_t(s_j | s_{j-1}) \prod_{j=1}^{n} p_e(x_j | s_j)$$

Parameters in the model

1. Initial state parameters ϕ_s for $s \in \{1, 2, ..., k\}$
2. Transition parameters $\phi_{s'|s}$ for $s, s' \in \{1, 2, ..., k\}$
3. Emission parameters $\phi_{e|s}$ for $s \in \{1, 2, ..., k\}$ and $e \in \{1, 2, ..., o\}$

If we use a specific symbol to denote *stop of a sequence*: $s_0 = *$

- Initial state parameters $\phi_{s|*}$
- Just look like transition parameters

Matrix representation for HMM parameters

- on whiteboard

Neural parameterisation

Chiu and Rush (2020)

https://arxiv.org/pdf/2011.04640

- Each state has an embedding: $\boldsymbol{E}_s \in \mathbb{R}^{|\mathcal{S}| \times h}$
- Each observation has an embedding: $\boldsymbol{E}_x \in \mathbb{R}^{|\mathcal{X}| \times h}$
- Intermediate representations for leaving and entering a state, as well as emitting a word:

$$\boldsymbol{H}_{\mathsf{out}}, \boldsymbol{H}_{\mathsf{in}}, \boldsymbol{H}_{\mathsf{emit}} = \mathrm{MLP}(\boldsymbol{E}_s)$$

- $\boldsymbol{H}_{\mathsf{out}}, \boldsymbol{H}_{\mathsf{in}}, \boldsymbol{H}_{\mathsf{emit}} \in \mathbb{R}^{|\mathcal{S}| \times h}$
- The HMM distributional parameters:

$$\boldsymbol{O} \propto \exp(\boldsymbol{H}_{\mathsf{emit}} \boldsymbol{E}_x^\top)$$

$$\boldsymbol{A} \propto \exp(\boldsymbol{H}_{\mathsf{in}} \boldsymbol{H}_{\mathsf{out}}^\top)$$

Harmonic word order

Morphology

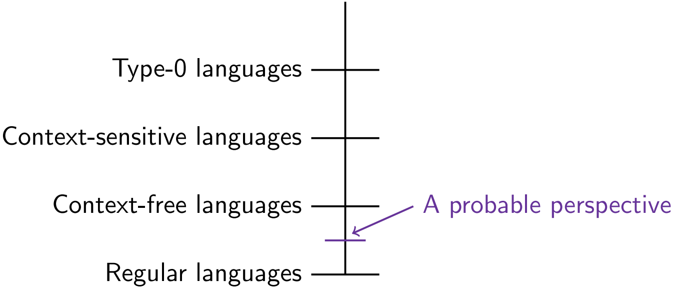
- Postpositional and head-final languages use suffixes and no prefixes.
- Prepositional and head-initial languages use not only prefixes but also suffixes.

Greenberg's word order universals

- Universal 3: Languages with dominant VSO order are always prepositional.
- Universal 4: With overwhelmingly greater than chance frequency, languages with normal SOV order are postpositional.
- Universal 5: If a language has dominant SOV order and the genitive follows the governing noun, then the adjective likewise follows the noun.
- Universal 17: With overwhelmingly more than chance frequency, languages with dominant order VSO have the adjective after the noun.

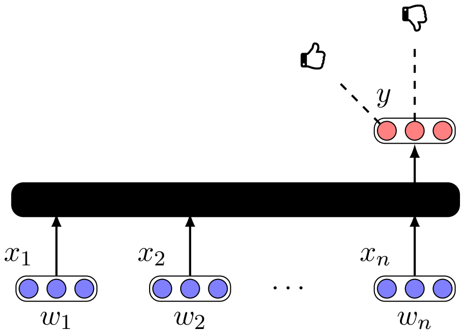
Empirical data can be found at `https://wals.info`.

With a *probable* perspective



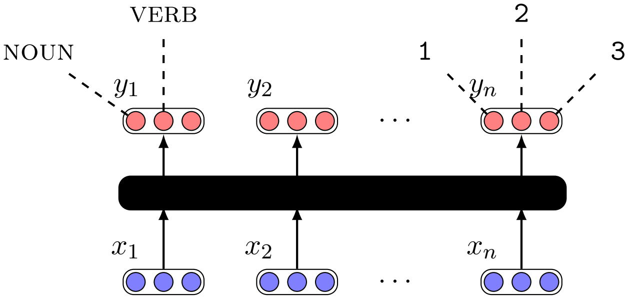
Sequence-to-Sequence

Many input tokens; one output token



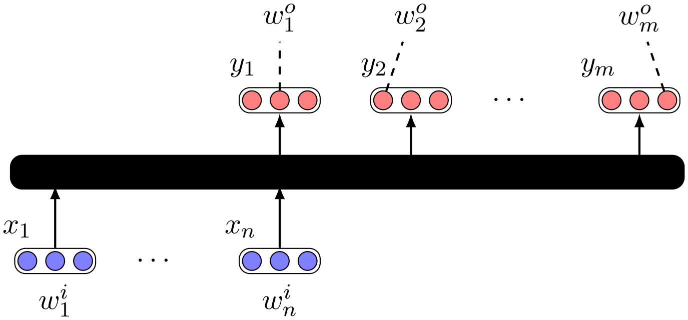
- Sentiment classification
- Document classification
- Automatic essay scoring
- ...

Many input tokens; many output token



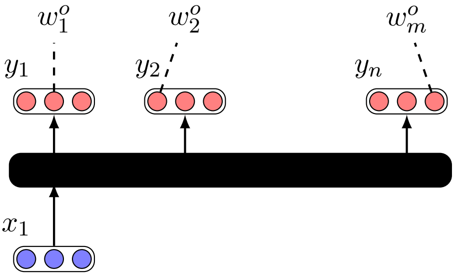
- POS tagging
- Segmentation and chunking
- Information extraction
- Dependency parsing
- ...

Many input tokens; many output token



- Machine translation
- Textual summarization
- ChatBot?
- ...

One input token; many output tokens



- Image captioning
- ...

Linguistic structure prediction

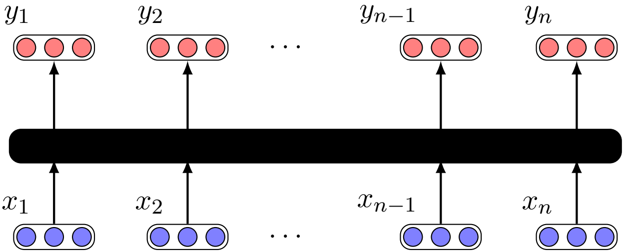
As a structured prediction problem

- Search space: Is this analysis possible?
- Measurement: Is this analysis *good*?
- Decode: find the analysis that obtains the highest score
- Parameter estimation: find good parameters

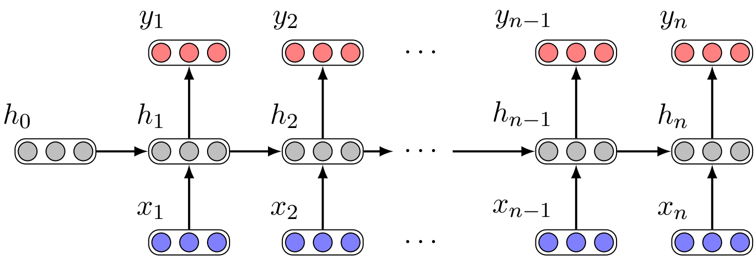
$$
\begin{aligned}
\mathbf{y}^*(\mathbf{x};\theta) &= \arg\max_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \textsc{Score}(\mathbf{x},\mathbf{y}) \\
&= \arg\max_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \arg\max_{\mathbf{d}:\textsc{Deriv}(\mathbf{d})=\mathbf{y}} \sum_{s=1}^{S} \textsc{StepScore}(\mathbf{x},\mathbf{d}_s)
\end{aligned}
$$

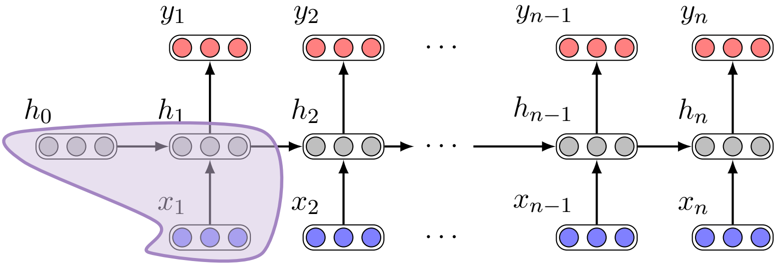
generate a structure step by step

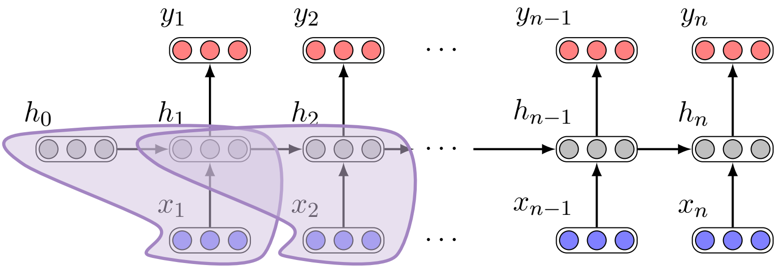
Recurrent Neural Networks

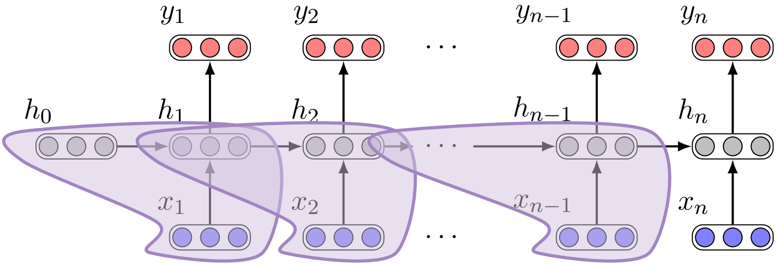


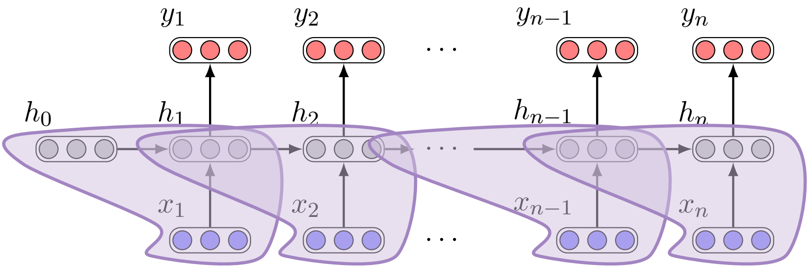
y_1
y_2
y_{n-1}
y_n
$\cdots$
x_1
x_2
x_{n-1}
x_n
$\cdots$


y_1
y_2
$\cdots$
y_{n-1}
y_n
h_0
h_1
h_2
$\cdots$
h_{n-1}
h_n
x_1
x_2
$\cdots$
x_{n-1}
x_n

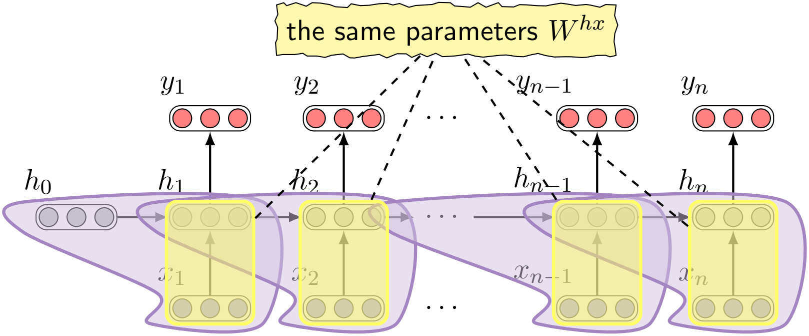

y_1
y_2
$\cdots$
y_{n-1}
y_n
h_0
h_1
h_2
h_{n-1}
h_n
x_1
x_2
x_{n-1}
x_n


y_1
y_2
y_{n-1}
y_n
h_0
h_1
h_2
h_{n-1}
h_n
x_1
x_2
x_{n-1}
x_n

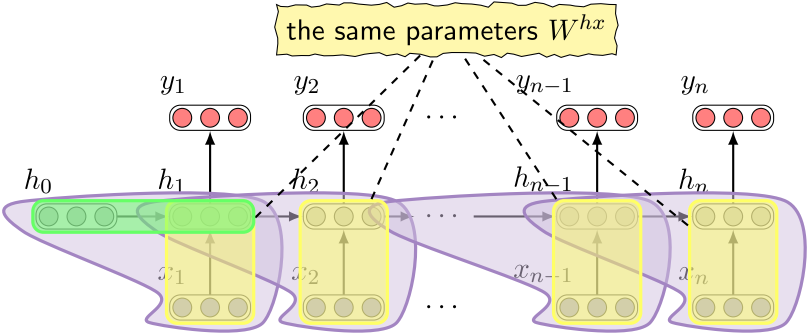

y_1
y_2
y_{n-1}
y_n
h_0
h_1
h_2
h_{n-1}
h_n
x_1
x_2
x_{n-1}
x_n



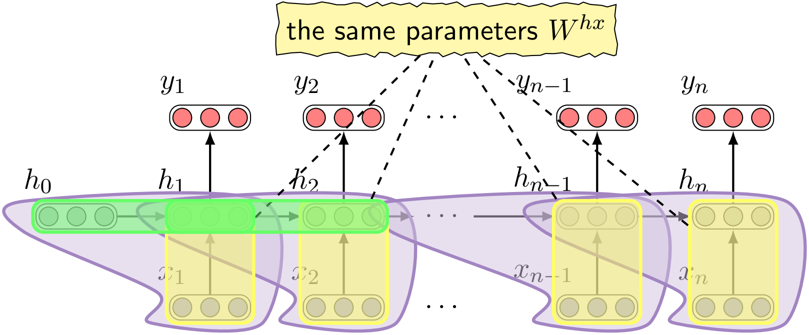
$$h_m = \text{RNN}(x_m, h_{m-1})$$



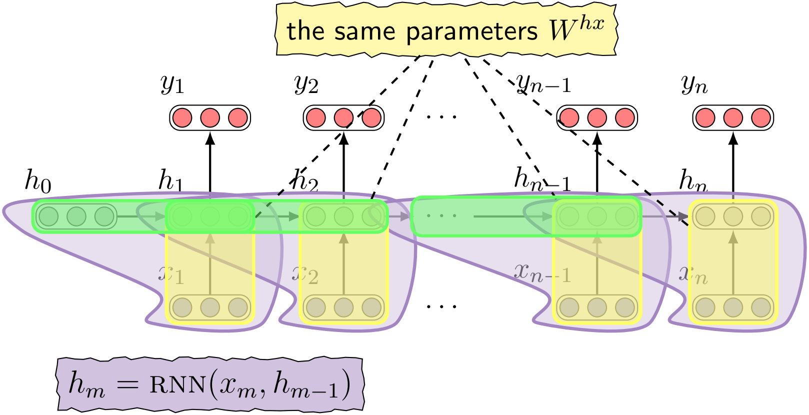
$$h_m = \mathrm{RNN}(x_m, h_{m-1})$$



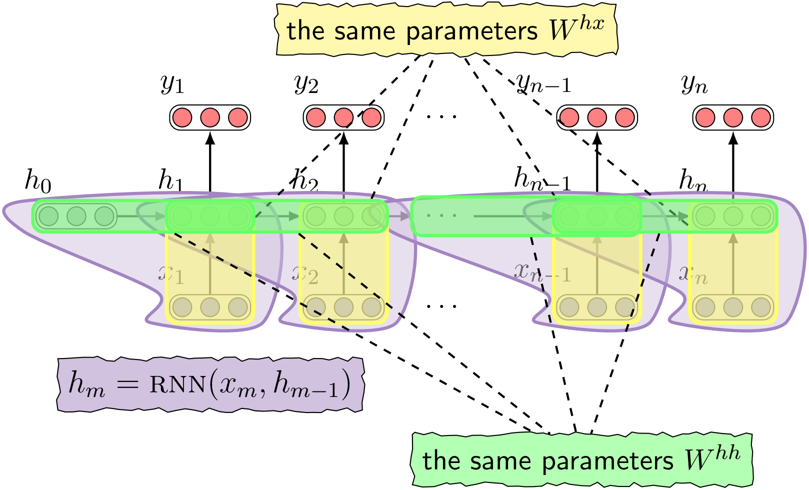
$$h_m = \text{RNN}(x_m, h_{m-1})$$

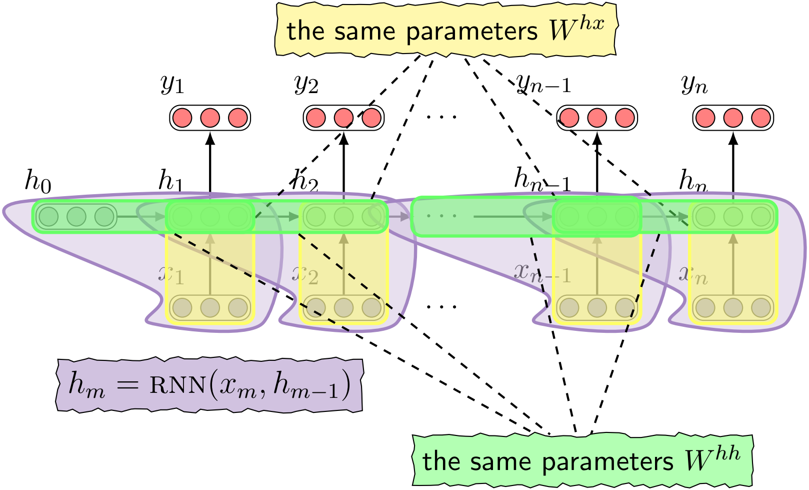


$$h_m = \mathrm{RNN}(x_m, h_{m-1})$$



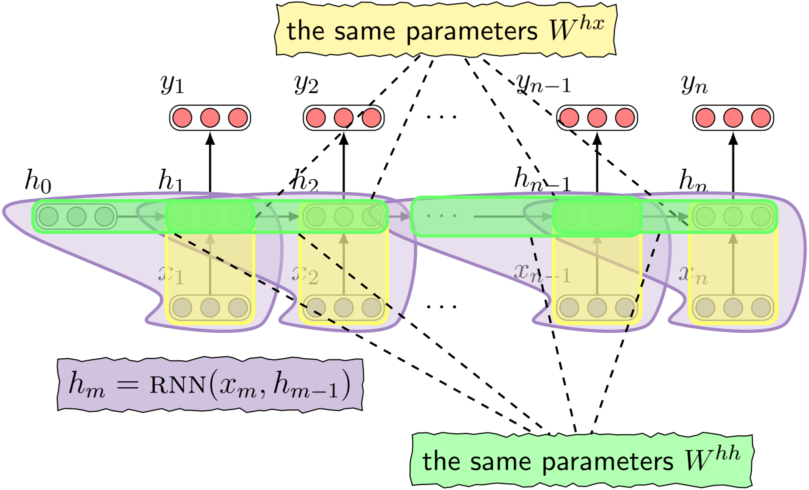
the same parameters W^{hx}
y_1
y_2
y_{n-1}
y_n
h_0
h_1
h_2
h_{n-1}
h_n
x_1
x_2
x_{n-1}
x_n
$h_m = \mathrm{RNN}(x_m, h_{m-1})$


the same parameters W^{hx}
y_1
y_2
y_{n-1}
y_n
h_0
h_1
h_2
h_{n-1}
h_n
x_1
x_2
x_{n-1}
x_n
$h_m = \mathrm{RNN}(x_m, h_{m-1})$
the same parameters W^{hh}



The context at token m is summarized by a recurrently-updated vector:

$$h_m = g(W^{hx}x_m + W^{hh}h_{m-1} + b^h)$$

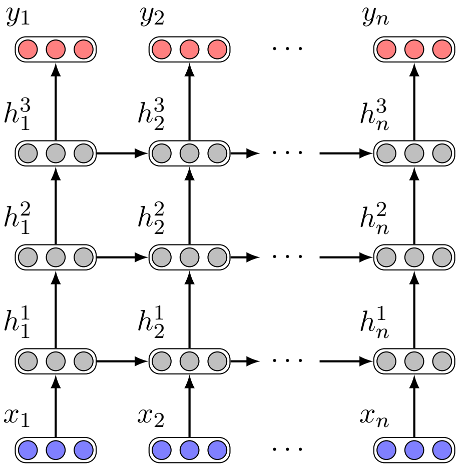


The context at token m is summarized by a recurrently-updated vector:

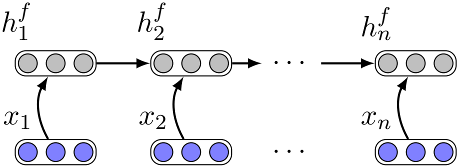
$$h_m = g(W^{hx}x_m + W^{hh}h_{m-1} + b^h)$$

A straightforward application to sequence labelling is to score each tag y_m with a log-linear function.

Multiple hidden layers

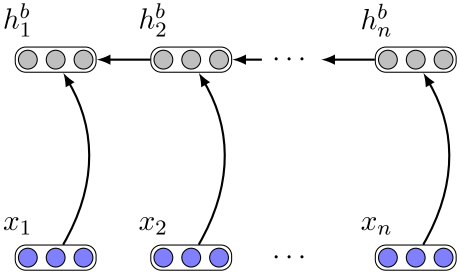


Bidirectional RNN



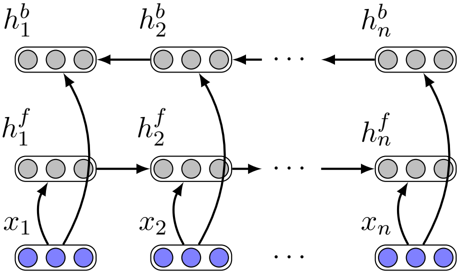
- $h_i^f = \text{RNN}_{\textsf{forward}}(x, i)$

Bidirectional RNN



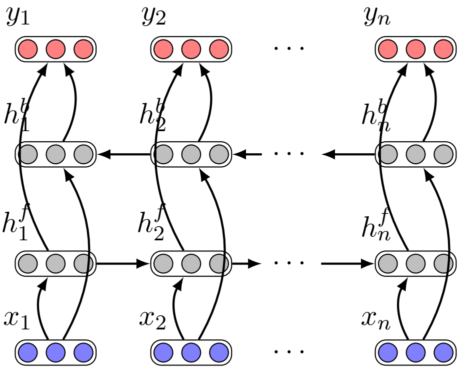
- $h_i^f = \text{RNN}_{\text{forward}}(x, i)$
- $h_i^b = \text{RNN}_{\text{backward}}(x, i)$

Bidirectional RNN



- $h_i^f = \text{RNN}_{\textsf{forward}}(x, i)$
- $h_i^b = \text{RNN}_{\textsf{backward}}(x, i)$
- $h_i = h_i^f \oplus h_i^b$

Bidirectional RNN



- $h_i^f = \text{RNN}_{\textsf{forward}}(x, i)$
- $h_i^b = \text{RNN}_{\textsf{backward}}(x, i)$
- $h_i = h_i^f \oplus h_i^b$

Difficulties

- Despite having access to the entire preceding sequence, the information encoded in hidden states tends to be fairly local, more relevant to the most recent parts of the input sequence and recent decisions.
- It is often the case, however, that distant information is critical to many language applications.

I convinced her children are noisy.
I convinced her children who are noisy are noisy.

Difficulties

- Despite having access to the entire preceding sequence, the information encoded in hidden states tends to be fairly local, more relevant to the most recent parts of the input sequence and recent decisions.
- It is often the case, however, that distant information is critical to many language applications.

I convinced her children are noisy.
I convinced her children who are noisy are noisy.

- A second difficulty with training simple RNNs arises from the need to backpropagate the error signal back through time.
- A frequent result of this process is that the gradients are eventually driven to zero — the so-called vanishing gradients problem.

Difficulties

- Despite having access to the entire preceding sequence, the information encoded in hidden states tends to be fairly local, more relevant to the most recent parts of the input sequence and recent decisions.
- It is often the case, however, that distant information is critical to many language applications.

I convinced her children are noisy.
I convinced her children who are noisy are noisy.

- A second difficulty with training simple RNNs arises from the need to backpropagate the error signal back through time.
- A frequent result of this process is that the gradients are eventually driven to zero — the so-called vanishing gradients problem.

How can the two problems be solved with syntax?

Long Short-Term Memory

- RNNs have short-term memory
- Aim: lengthen the short-term memory

Long Short-Term Memory

- RNNs have short-term memory
- Aim: lengthen the short-term memory

Long short-term memory (LSTM) networks

- remove information no longer needed from the context
- add information likely to be needed for later decision making

Long Short-Term Memory

- RNNs have short-term memory
- Aim: lengthen the short-term memory

Long short-term memory (LSTM) networks

- remove information no longer needed from the context
- add information likely to be needed for later decision making

The *key* is to learn how to manage the context rather than hard-coding a strategy: *adding gates* to control the flow of information into and out of the units.

Long Short-Term Memory

basic computation

$$g_i = \tanh(W^{hh} h_{i-1} + W^{hx} x_i)$$

context vector

$$c_i = j_i + k_i$$

input gate: select the information to add to the current context.

$$\begin{aligned} i_i &= \sigma(W^{i,h} h_{i-1} + W^{i,x} x_i) \\ j_i &= g_i \odot i_i \end{aligned}$$

forget gate: delete information from the context

$$\begin{aligned} f_i &= \sigma(W^{f,h} h_{i-1} + W^{f,x} x_i) \\ k_i &= c_{i-1} \odot f_i \end{aligned}$$

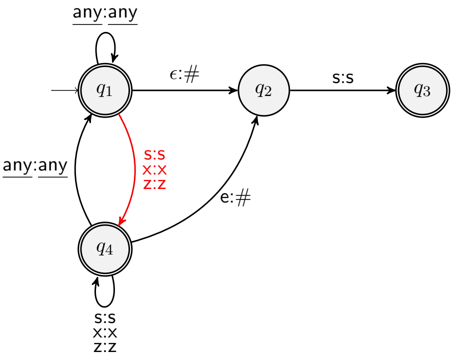
output gate: decide the information for the current hidden state.

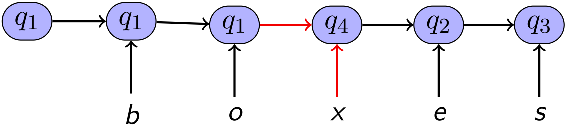
$$\begin{aligned} o_i &= \sigma(W^{o,h} h_{i-1} + W^{o,x} x_i) \\ h_i &= o_i \odot \tanh(c_i) \end{aligned}$$


```
rnn = nn.LSTM(10, 20, 2)
input = torch.randn(5, 3, 10)
h0 = torch.randn(2, 3, 20)
c0 = torch.randn(2, 3, 20)
output, (hn, cn) = rnn(input, (h0, c0))
```

photo idea from Hung-yi Lee

Connection to Finite State Machines





Reading

D Jurafsky and J Martin. *Speech and Language Processing*.

- Chapter 8. RNNs and LSTMs.
  `https://web.stanford.edu/~jurafsky/slp3/8.pdf`