RNNs & ELMO

CS 685, Spring 2021

Advanced Topics in Natural Language Processing

http://brenocon.com/cs685

https://people.cs.umass.edu/~brenocon/cs685_s21/

Brendan O'Connor

College of Information and Computer Sciences

University of Massachusetts Amherst

Announcements

- I'll do OH tomorrow 10am-11am - at the usual course zoom link

Recurrent neural networks

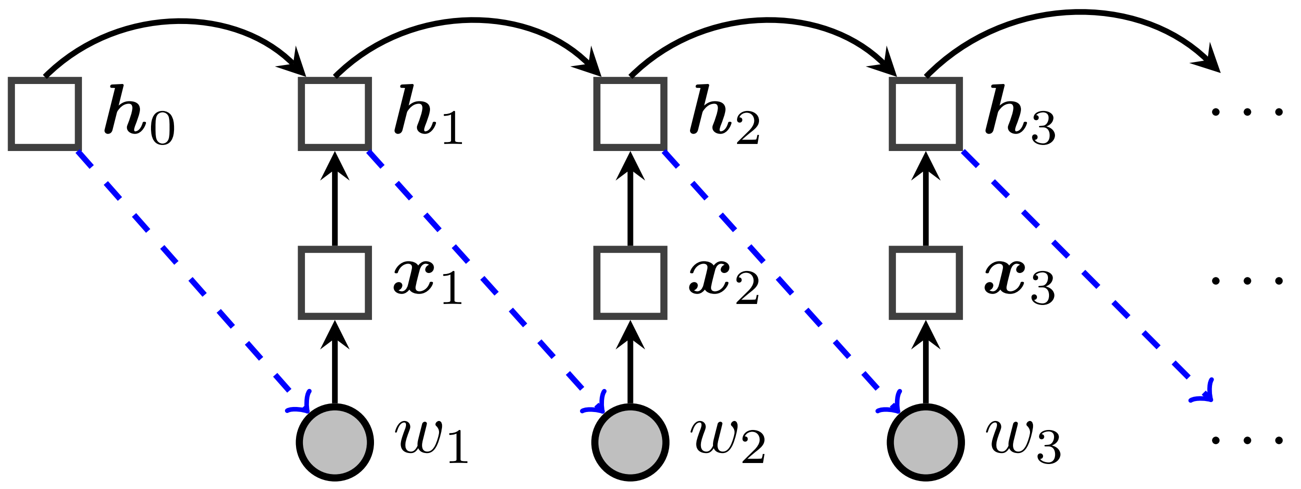


Figure 6.1: The recurrent neural network language model, viewed as an "unrolled" computation graph. Solid lines indicate direct computation, dotted blue lines indicate probabilistic dependencies, circles indicate random variables, and squares indicate computation nodes.

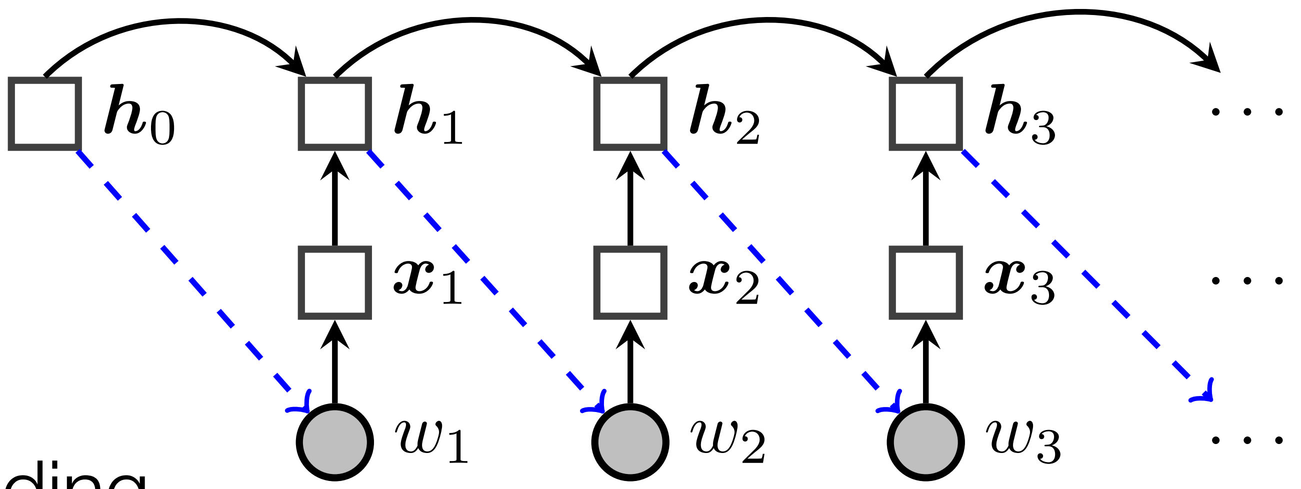
- Idea: extend a feedforward net to sequential data by iterating an NN at each position.
- Theoretically, an RNN can learn *any* update function. (Represent any Turing machine!)

[Diagram: Jacob Eisenstein]

Recurrent neural networks

h_m: hidden state at position m

ϕ_w: word embedding



$\sim$

$$x_m \triangleq \phi_{w_m} \quad h_m = \text{RNN}(x_m, h_{m-1}) \quad p(w_{m+1} \mid \underline{w_1, w_2, \ldots, w_m}) = \frac{\exp(\beta_{w_{m+1}} \cdot h_m)}{\sum_{w' \in \mathcal{V}} \exp(\beta_{w'} \cdot h_m)}$$

Elman ("vanilla") RNN unit: $\text{RNN}(x_m, h_{m-1}) \triangleq g(\Theta h_{m-1} + x_m)$

- Is this Markovian?
- What sort of information could $\mathbf{h_m}$ contain?
- Hyperparameter: $\mathbf{h_m}$, $\boldsymbol{\beta}_\mathbf{w}$ length K.

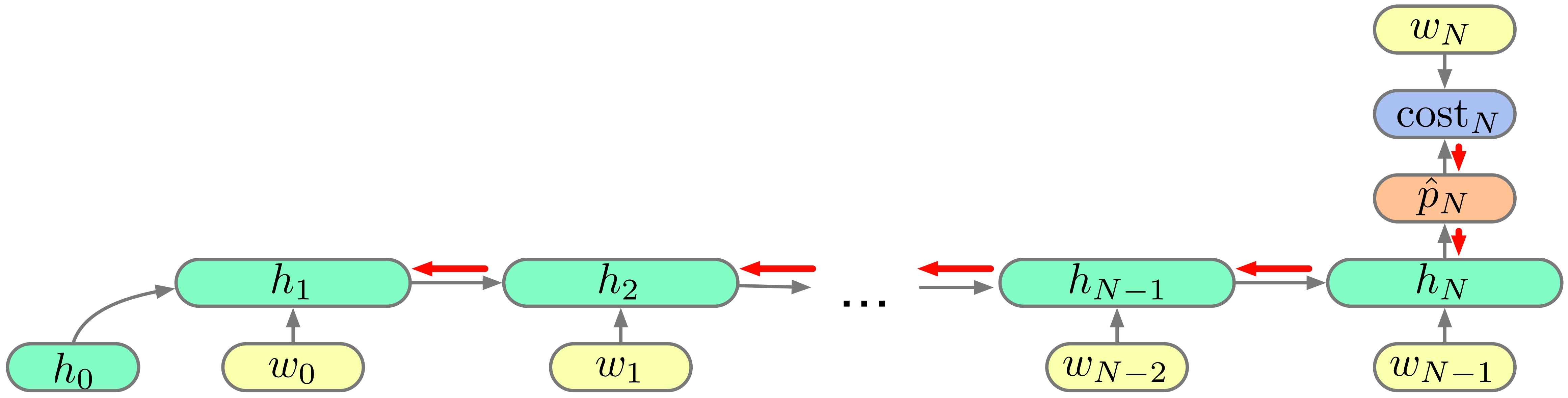
[Diagram: Jacob Eisenstein]

Capturing Long Range Dependencies

If an RNN Language Model is to outperform an n-gram model it must discover and represent long range dependencies:

$p(\text{sandcastle} \mid \text{Alice went to the beach. There she built a})$

While a simple RNN LM can represent such dependencies in theory, can it learn them?

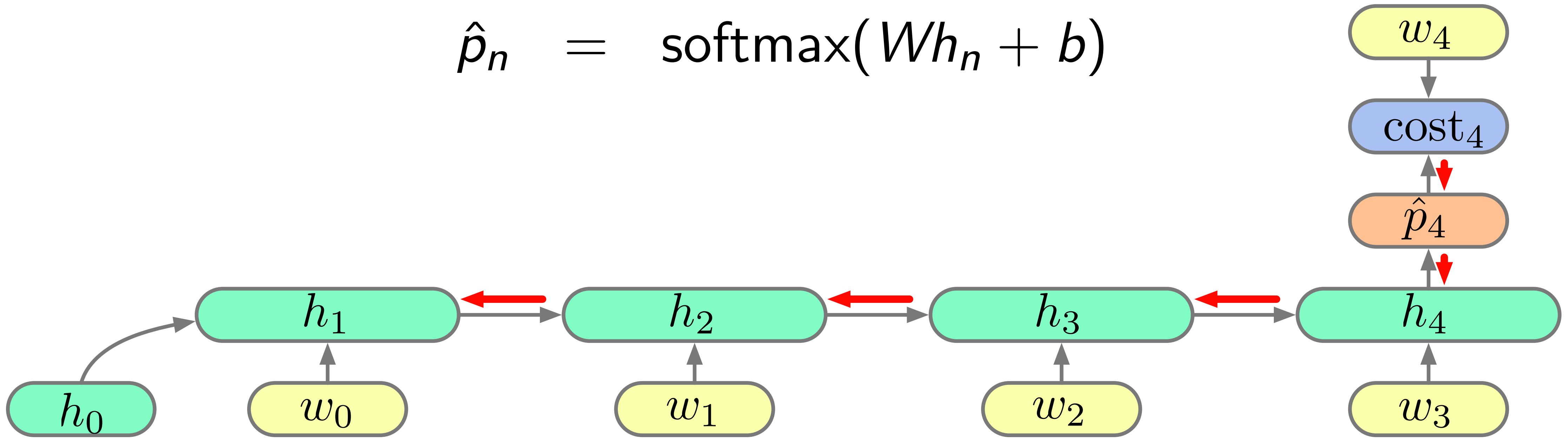


[Slide: Phil Blunsom]

RNNs: Exploding and Vanishing Gradients

Consider the path of partial derivatives linking a change in cost_4 to changes in h_1:

$$\begin{aligned} h_n &= g(V[x_n; h_{n-1}] + c) \\ \hat{p}_n &= \text{softmax}(W h_n + b) \end{aligned}$$



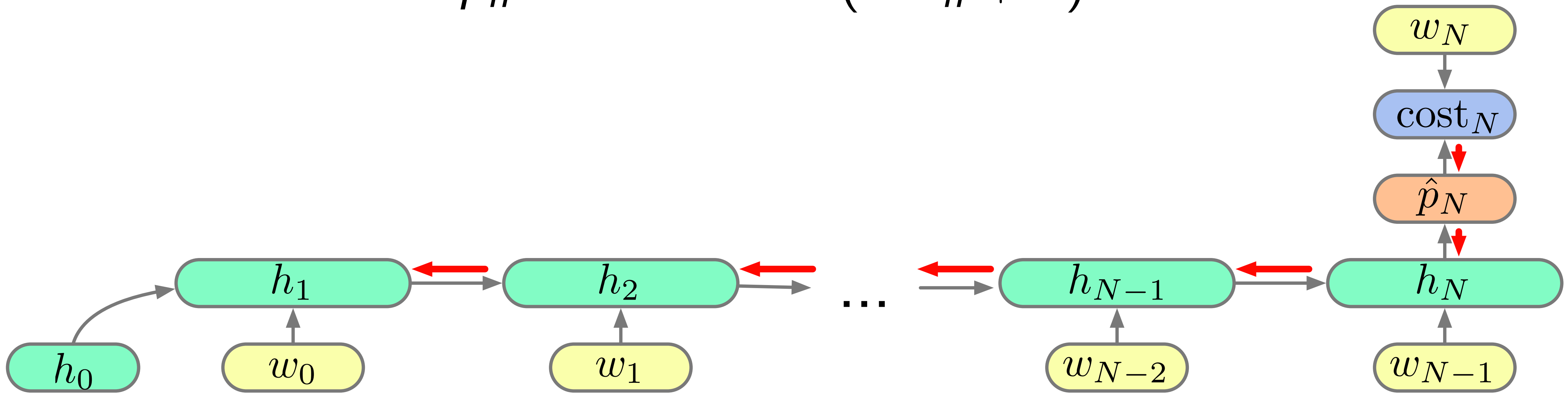
$$\frac{\partial \text{cost}_4}{\partial h_1} = \frac{\partial \text{cost}_4}{\partial \hat{p}_4} \frac{\partial \hat{p}_4}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1}$$

[Slide: Phil Blunsom]

RNNs: Exploding and Vanishing Gradients

Consider the path of partial derivatives linking a change in cost_N to changes in h_1:

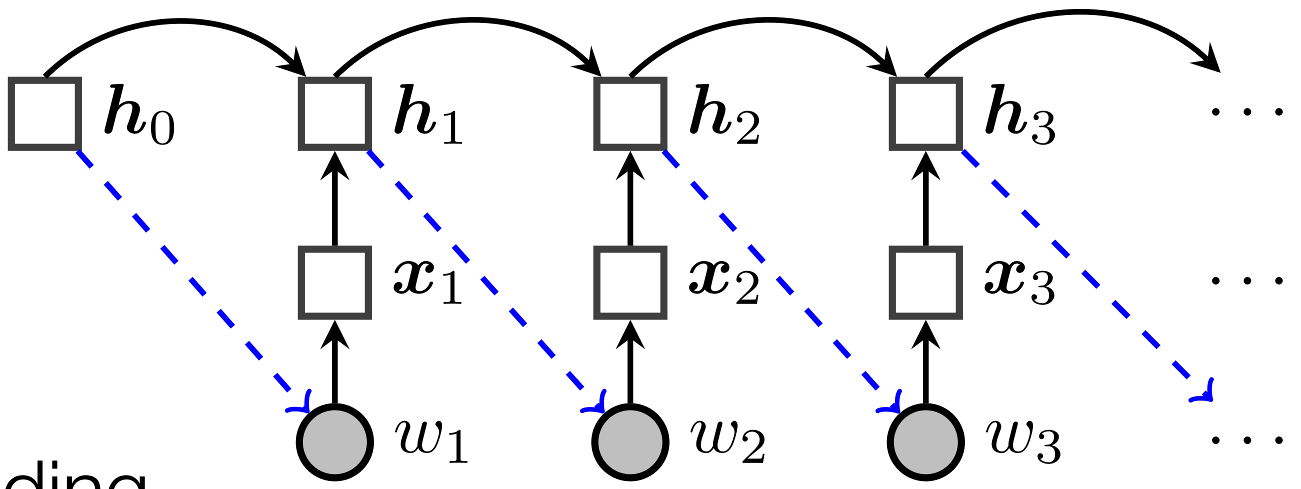
$$\begin{aligned} h_n &= g(V[x_n; h_{n-1}] + c) \\ \hat{p}_n &= \text{softmax}(W h_n + b) \end{aligned}$$



$$\frac{\partial \text{cost}_N}{\partial h_1} = \frac{\partial \text{cost}_N}{\partial \hat{p}_N} \frac{\partial \hat{p}_N}{\partial h_N} \left(\prod_{n \in \{N,\ldots,2\}} \frac{\partial h_n}{\partial h_{n-1}} \right)$$

[Slide: Phil Blunsom]

Recurrent neural networks



h_m: hidden state at position m

ϕ_w: word embedding

$$\boldsymbol{x}_m \triangleq \boldsymbol{\phi}_{w_m} \quad \boldsymbol{h}_m = \mathrm{RNN}(\boldsymbol{x}_m, \boldsymbol{h}_{m-1}) \quad \mathrm{p}(w_{m+1} \mid w_1, w_2, \ldots, w_m) = \frac{\exp(\boldsymbol{\beta}_{w_{m+1}} \cdot \boldsymbol{h}_m)}{\sum_{w' \in \mathcal{V}} \exp(\boldsymbol{\beta}_{w'} \cdot \boldsymbol{h}_m)}$$

Elman ("vanilla") RNN unit: $\mathrm{RNN}(\boldsymbol{x}_m, \boldsymbol{h}_{m-1}) \triangleq g(\boldsymbol{\Theta}\boldsymbol{h}_{m-1} + \boldsymbol{x}_m)$

Alternative RNN unit: a *gated* unit

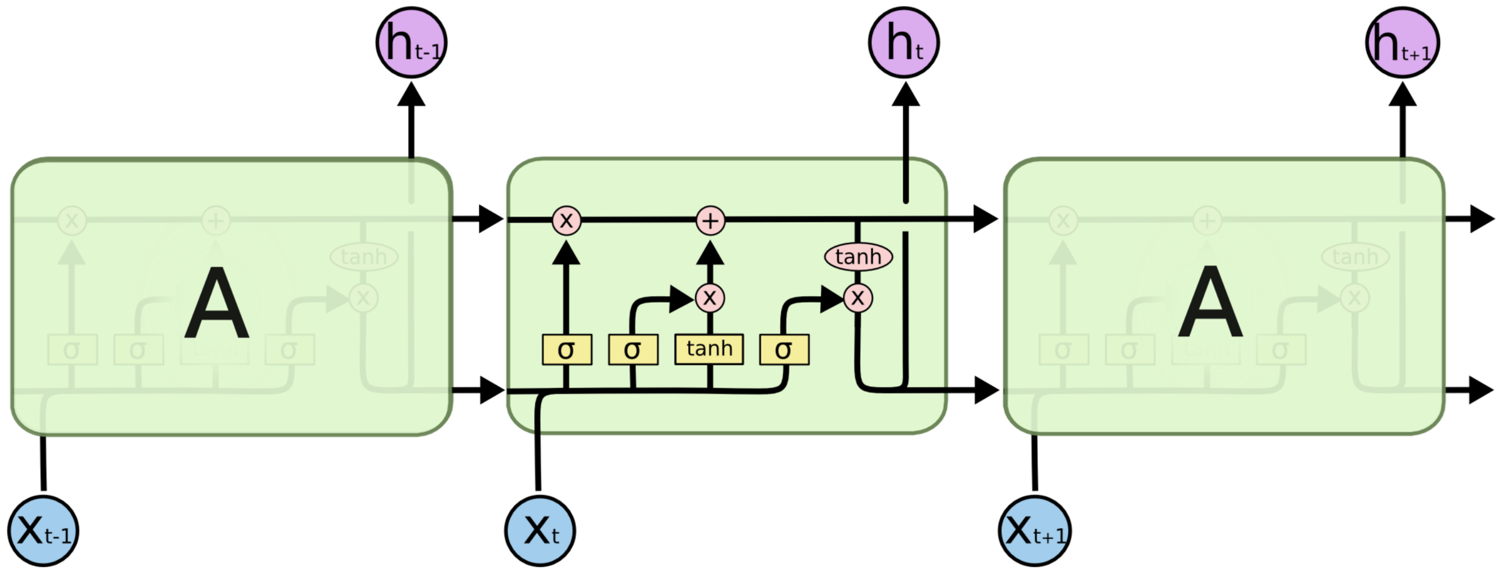
Most common: **LSTM**

$$\begin{aligned}
\boldsymbol{f}_{m+1} &= \sigma(\boldsymbol{\Theta}^{(h\to f)}\boldsymbol{h}_m + \boldsymbol{\Theta}^{(x\to f)}\boldsymbol{x}_{m+1} + \boldsymbol{b}_f) && \text{forget gate}\\
\boldsymbol{i}_{m+1} &= \sigma(\boldsymbol{\Theta}^{(h\to i)}\boldsymbol{h}_m + \boldsymbol{\Theta}^{(x\to i)}\boldsymbol{x}_{m+1} + \boldsymbol{b}_i) && \text{input gate}\\
\tilde{\boldsymbol{c}}_{m+1} &= \tanh(\boldsymbol{\Theta}^{(h\to c)}\boldsymbol{h}_m + \boldsymbol{\Theta}^{(w\to c)}\boldsymbol{x}_{m+1}) && \text{update candidate}\\
\boldsymbol{c}_{m+1} &= \boldsymbol{f}_{m+1} \odot \boldsymbol{c}_m + \boldsymbol{i}_{m+1} \odot \tilde{\boldsymbol{c}}_{m+1} && \text{memory cell update}\\
\boldsymbol{o}_{m+1} &= \sigma(\boldsymbol{\Theta}^{(h\to o)}\boldsymbol{h}_m + \boldsymbol{\Theta}^{(x\to o)}\boldsymbol{x}_{m+1} + \boldsymbol{b}_o) && \text{output gate}\\
\boldsymbol{h}_{m+1} &= \boldsymbol{o}_{m+1} \odot \tanh(\boldsymbol{c}_{m+1}) && \text{output.}
\end{aligned}$$

LSTM RNN (Long short-term memory)

- Goals:
  - 1. Be able to "remember" for longer distances
  - 2. Stable backpropagation during training
- Augment individual timesteps with a number of specialized vectors and gating functions. *(There are alternative gated RNNs, but at this point LSTM has won.)*
- LSTM RNNs are ~now? ~recently? a common baseline NN model in NLP

- Main state
  - **c**: Memory cell
  - **h**: Hidden state
- Update system
  - **g**: proposed new cell
  - **f, i, o**: Forget, Input, Output gates control acceptance of **g** into new cell & state



Christopher Olah: Understanding LSTM Networks
`colah.github.io/posts/2015-08-Understanding-LSTMs/`

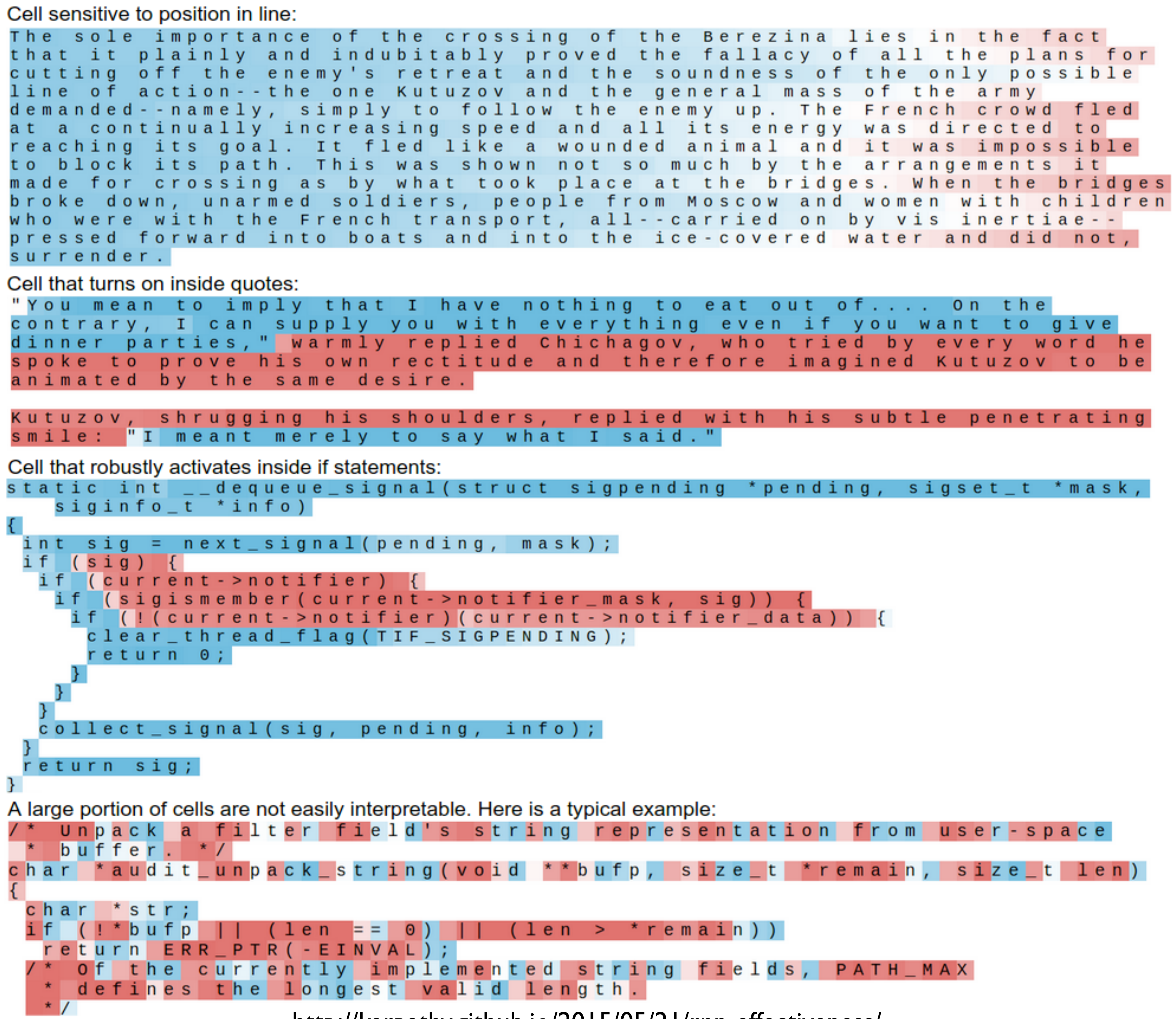
Character LMs comparison: LSTM vs. N-Gram

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.
```

```
First Citizen:
Nay, then, that was hers,
It speaks against your other service:
But since the
youth of the circumstance be spoken:
Your uncle and one Baptista's daughter.

SEBASTIAN:
Do I stand till the break off.

BIRON:
Hide thy head.

VENTIDIUS:
He purposeth to Athens: whither, with the vow
I made to handle you.
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/
http://nbviewer.jupyter.org/gist/yoavg/d76121dfde2618422139

Structure awareness



Cell sensitive to position in line:

```
The sole importance of the crossing of the Berezina lies in the fact
that it plainly and indubitably proved the fallacy of all the plans for
cutting off the enemy's retreat and the soundness of the only possible
line of action--the one Kutuzov and the general mass of the army
demanded--namely, simply to follow the enemy up. The French crowd fled
at a continually increasing speed and all its energy was directed to
reaching its goal. It fled like a wounded animal and it was impossible
to block its path. This was shown not so much by the arrangements it
made for crossing as by what took place at the bridges. When the bridges
broke down, unarmed soldiers, people from Moscow and women with children
who were with the French transport, all--carried on by vis inertiae--
pressed forward into boats and into the ice-covered water and did not,
surrender.
```

Cell that turns on inside quotes:

```
"You mean to imply that I have nothing to eat out of.... On the
contrary, I can supply you with everything even if you want to give
dinner parties," warmly replied Chichagov, who tried by every word he
spoke to prove his own rectitude and therefore imagined Kutuzov to be
animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating
smile: "I meant merely to say what I said."
```

Cell that robustly activates inside if statements:

```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
   siginfo_t *info)
{
 int sig = next_signal(pending, mask);
 if (sig) {
  if (current->notifier) {
   if (sigismember(current->notifier_mask, sig)) {
    if (!(current->notifier)(current->notifier_data)) {
     clear_thread_flag(TIF_SIGPENDING);
     return 0;
    }
   }
  }
  collect_signal(sig, pending, info);
 }
 return sig;
}
```

A large portion of cells are not easily interpretable. Here is a typical example:

```
/* Unpack a filter field's string representation from user-space
 * buffer. */
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
 char *str;
 if (!*bufp || (len == 0) || (len > *remain))
  return ERR_PTR(-EINVAL);
 /* Of the currently implemented string fields, PATH_MAX
  * defines the longest valid length.
  */
```

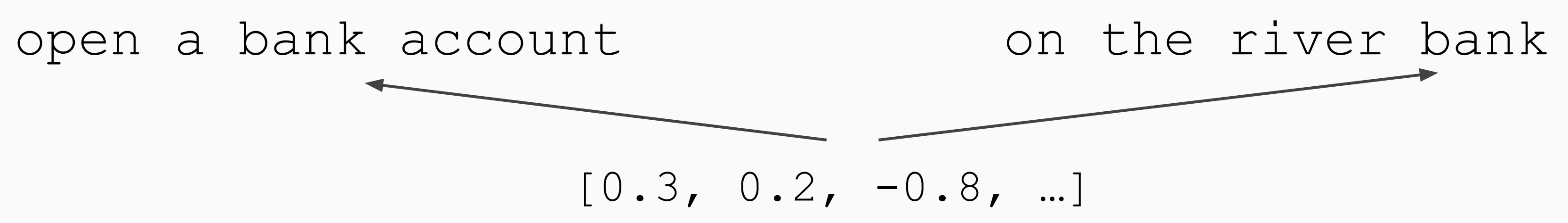
http://karpathy.github.io/2015/05/21/rnn-effectiveness/

can we use language models to produce word embeddings?

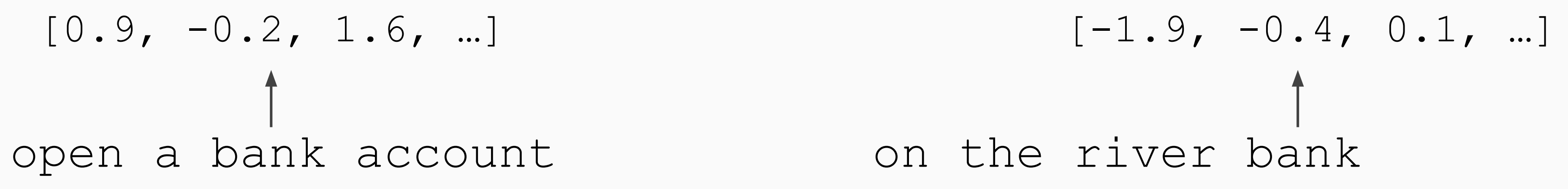
Deep contextualized word representations. Peters et al., NAACL 2018

Contextual Representations

- **Problem**: Word embeddings are applied in a context free manner

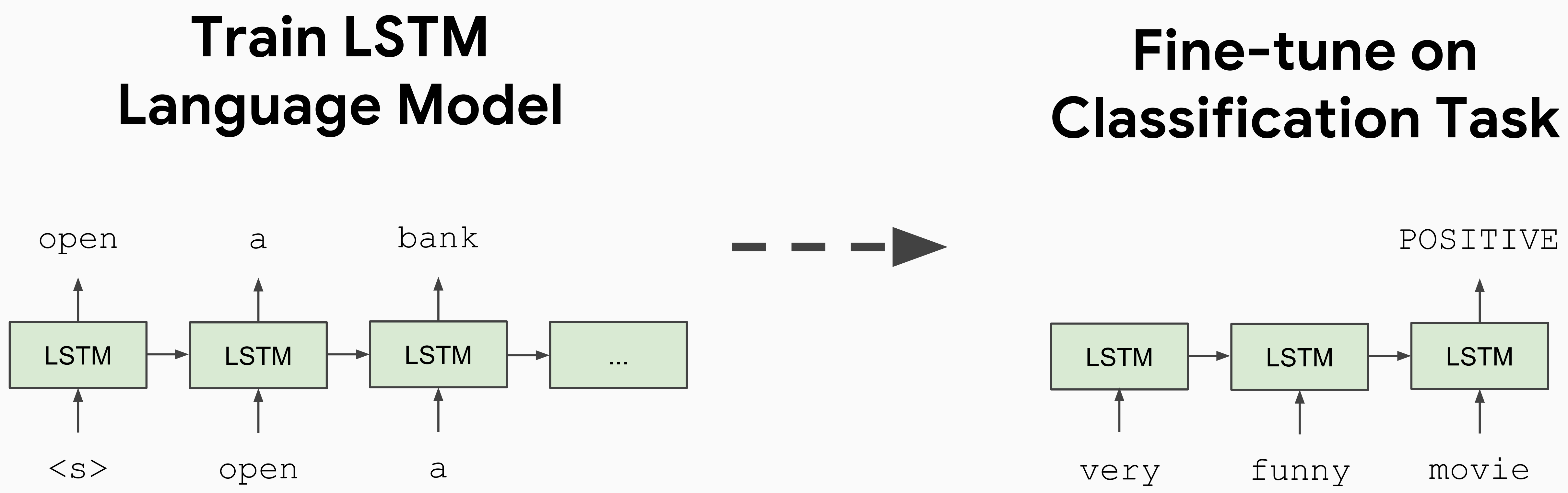


- **Solution**: Train *contextual* representations on text corpus



History of Contextual Representations

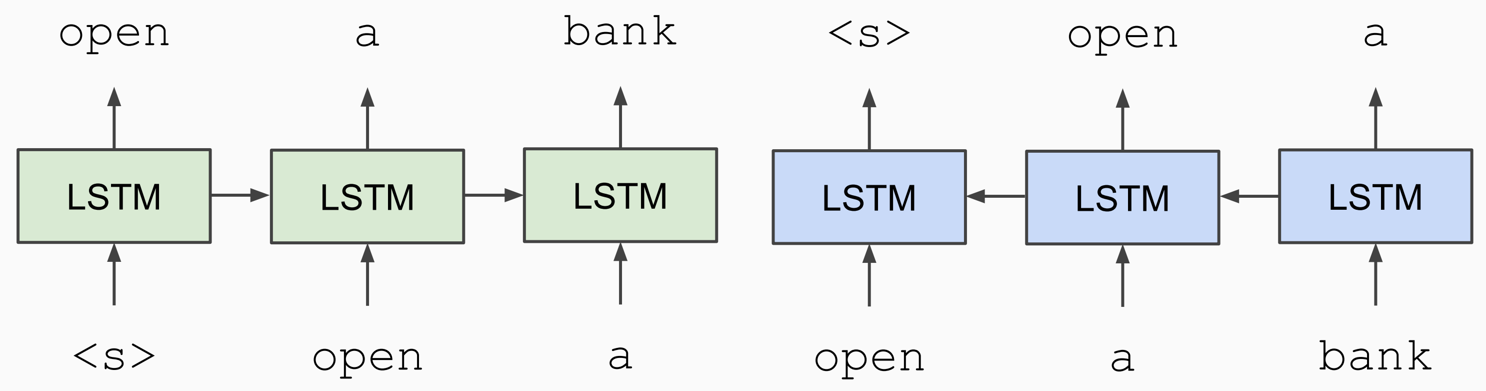
- *Semi-Supervised Sequence Learning*, Google, 2015



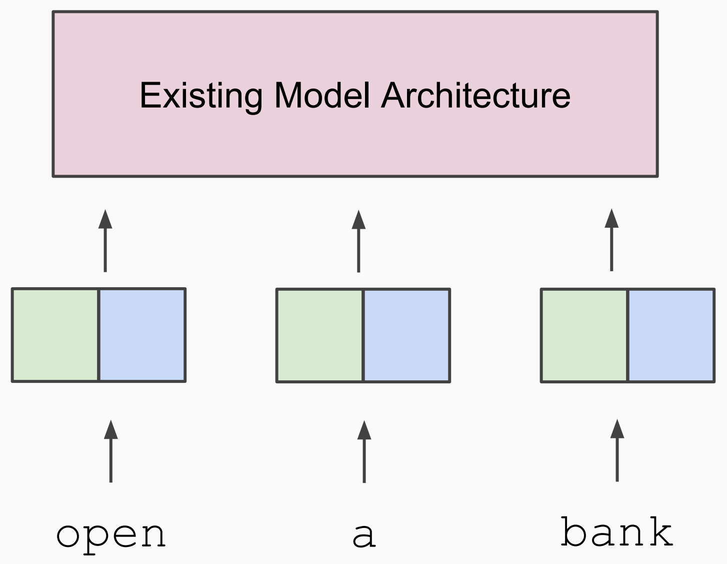
History of Contextual Representations

- *ELMo: Deep Contextual Word Embeddings*, AI2 & University of Washington, 2017

Train Separate Left-to-Right and Right-to-Left LMs



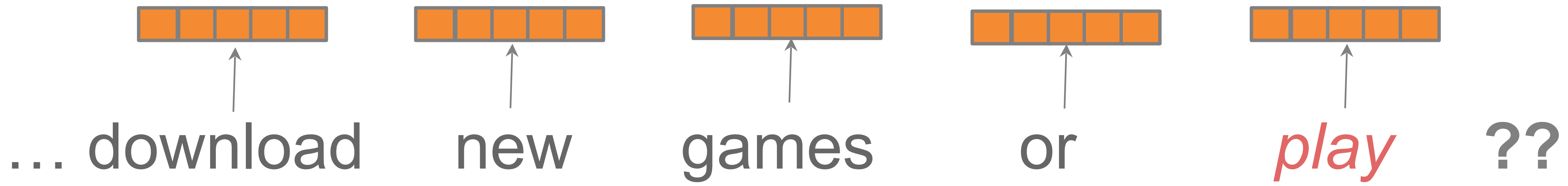
Apply as "Pre-trained Embeddings"



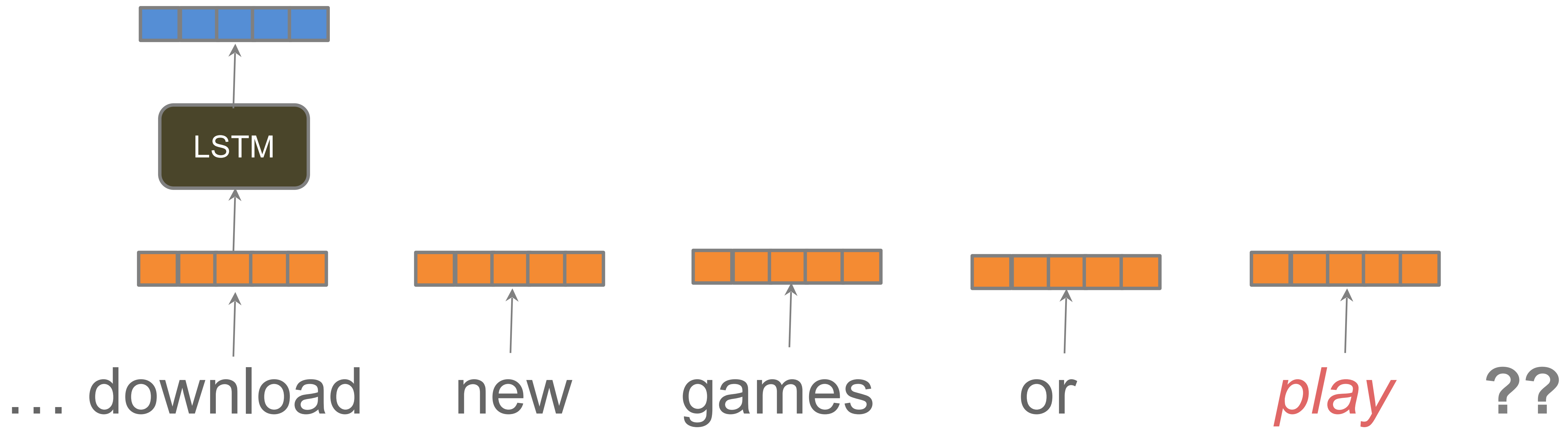
Deep bidirectional language model

… download new games or *play* ??

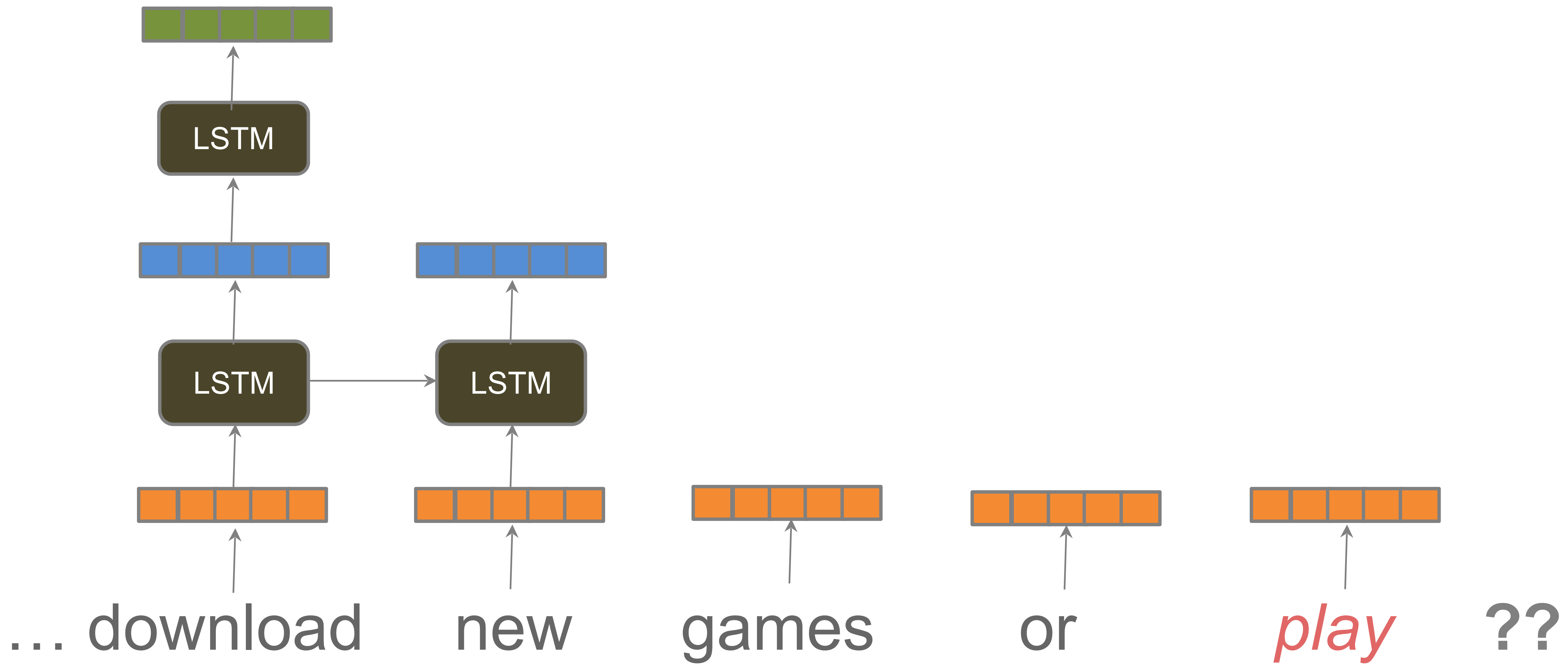
Deep bidirectional language model



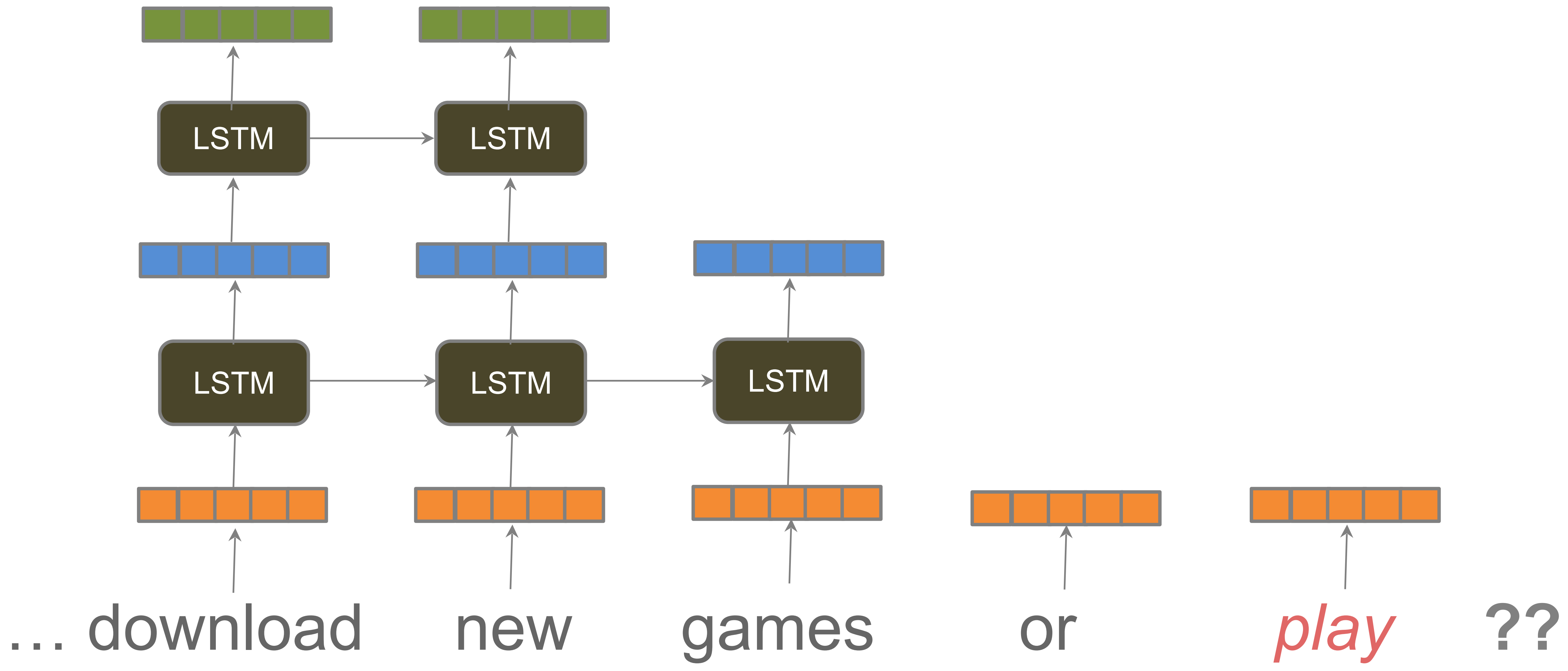
Deep bidirectional language model



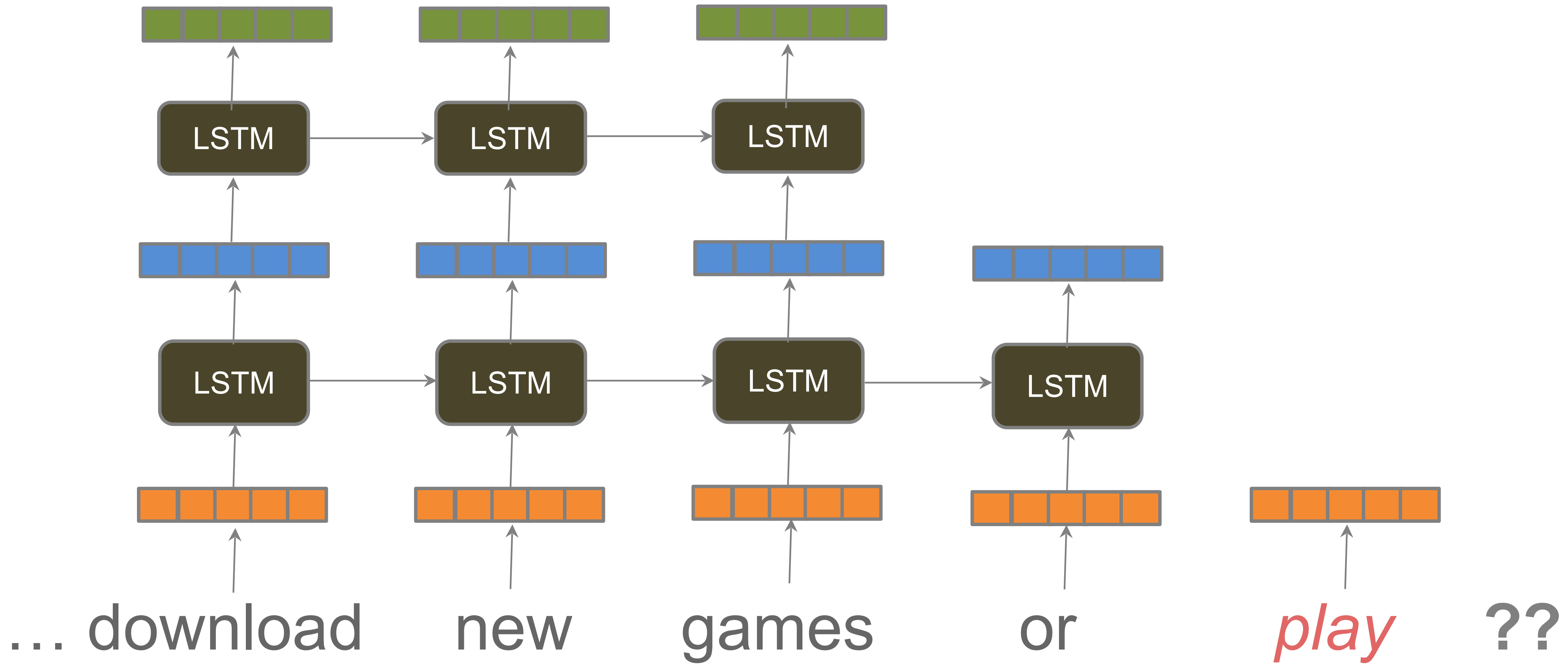
Deep bidirectional language model



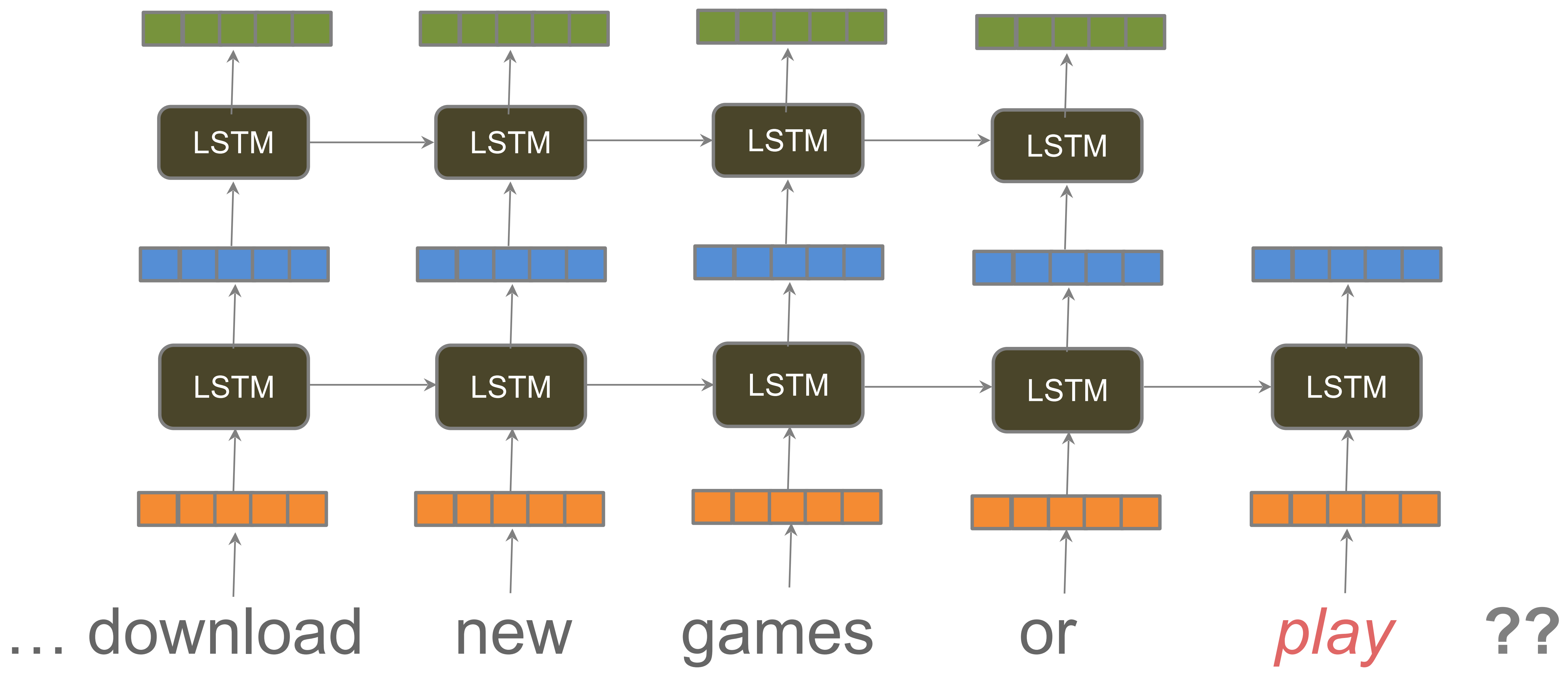
Deep bidirectional language model



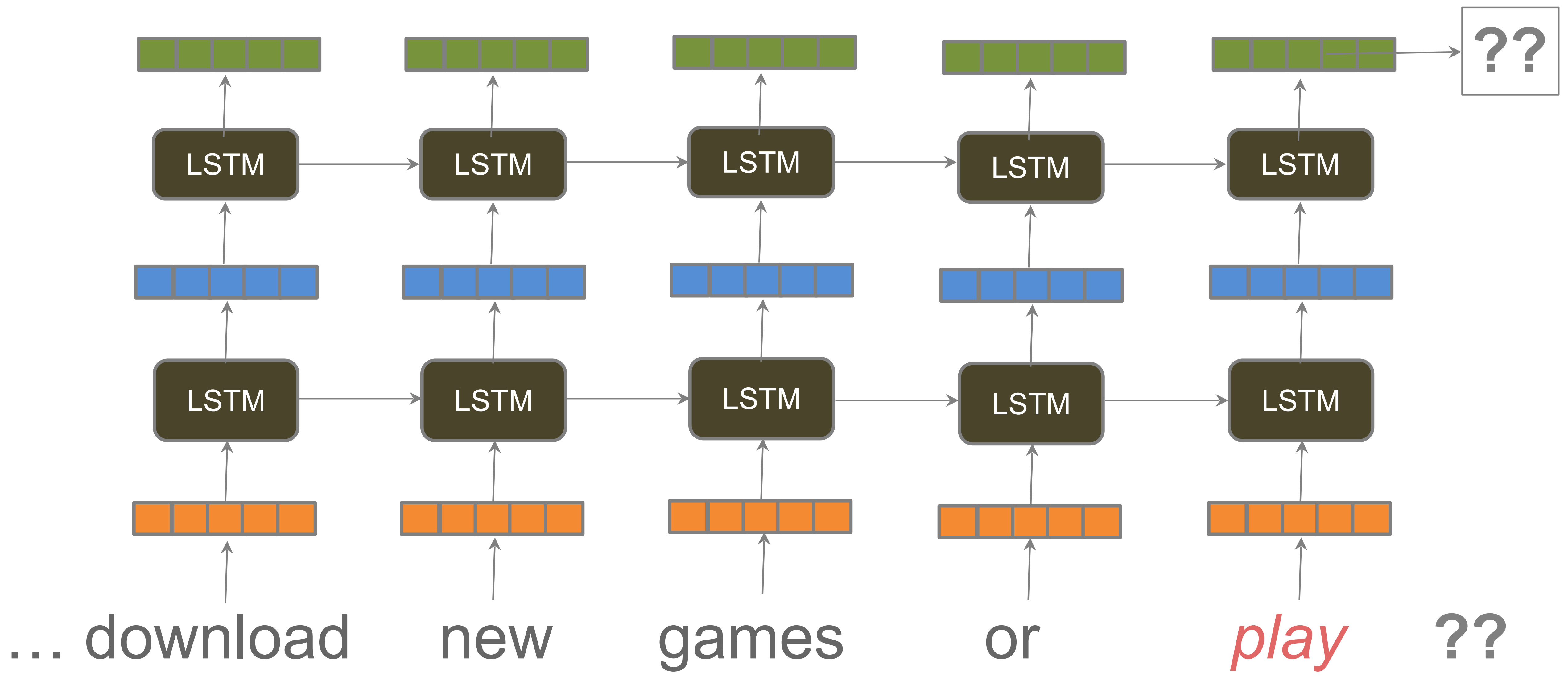
Deep bidirectional language model



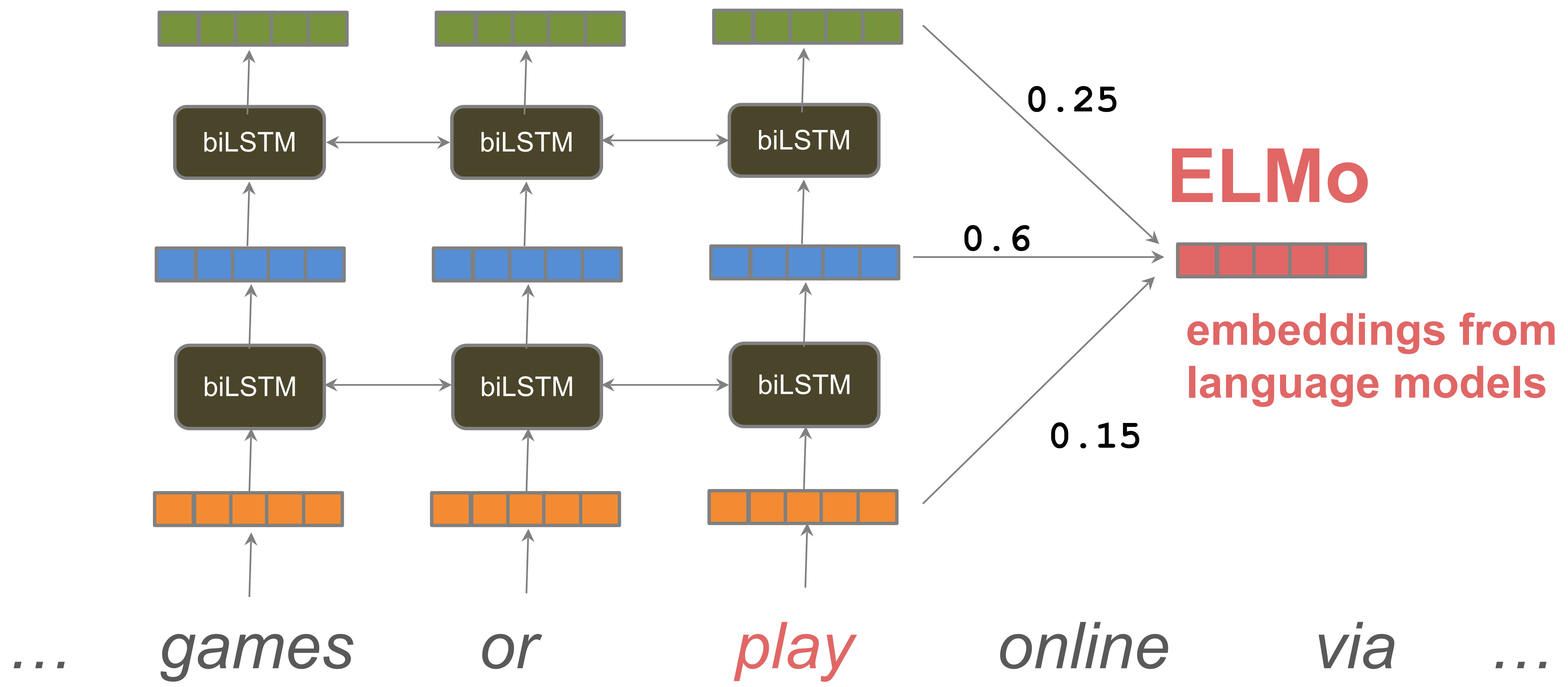
Deep bidirectional language model



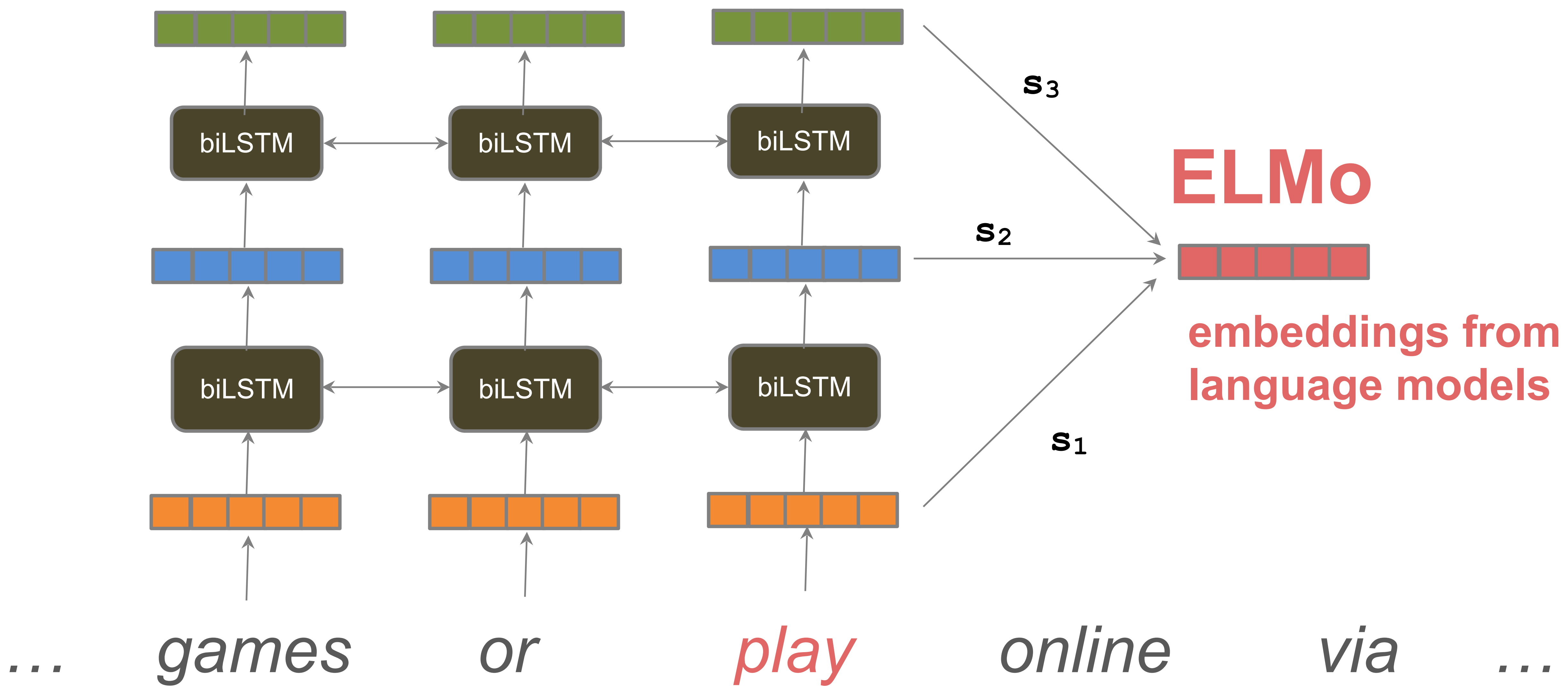
Deep bidirectional language model



Use all layers of language model



Learned task-specific combination of layers



The biLM produces $2L + 1$ intermediate representations:

$$\begin{aligned} R_k =& \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ =& \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\} \end{aligned}$$

where $\mathbf{h}_{k,0}^{LM} = \mathbf{x}_k^{LM}$ is the token layer and $\mathbf{h}_{k,j}^{LM} = [\overrightarrow{\mathbf{h}}_{k,j}^{LM}; \overleftarrow{\mathbf{h}}_{k,j}^{LM}]$, for each biLSTM layer.
ELMo: A task specific combination of these features:

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}.$$

where $\mathbf{s}^{task}$ are softmax-normalized weights and γ^{task} is a scaling parameter.

The biLM produces $2L+1$ intermediate representations:

$$\begin{aligned} R_k =& \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ =& \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\} \end{aligned}$$

where $\mathbf{h}_{k,0}^{LM} = \mathbf{x}_k^{LM}$ is the token layer and $\mathbf{h}_{k,j}^{LM} = [\overrightarrow{\mathbf{h}}_{k,j}^{LM}; \overleftarrow{\mathbf{h}}_{k,j}^{LM}]$, for each biLSTM layer.

ELMo: A task specific combination of these features:

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} \overset{\text{layer weights}}{\boxed{s_j^{task}}} \mathbf{h}_{k,j}^{LM}.$$

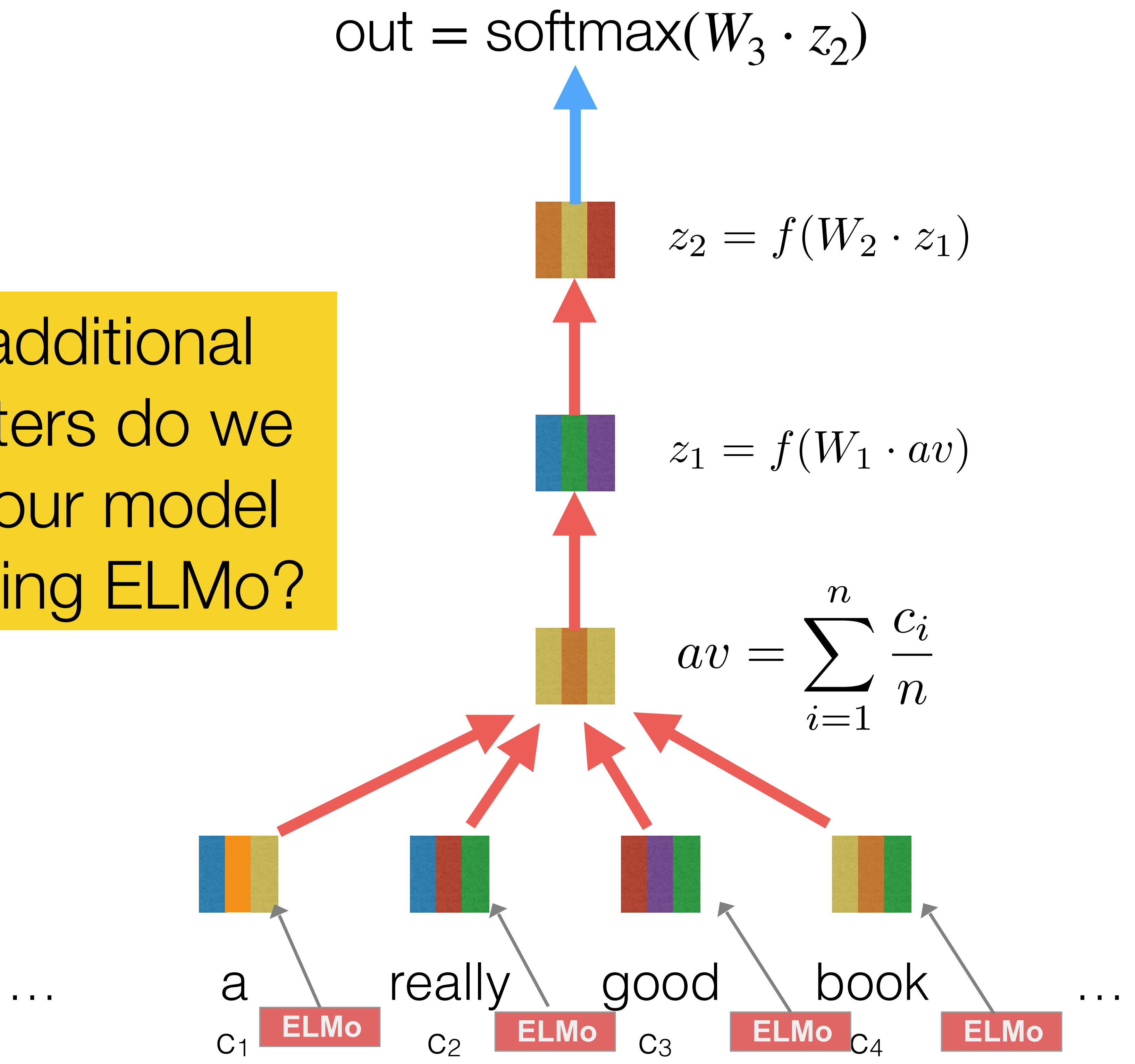
where $\mathbf{s}^{task}$ are softmax-normalized weights and γ^{task} is a scaling parameter.

Contextual representations

ELMo representations are **contextual** – they depend on the entire sentence in which a word is used.

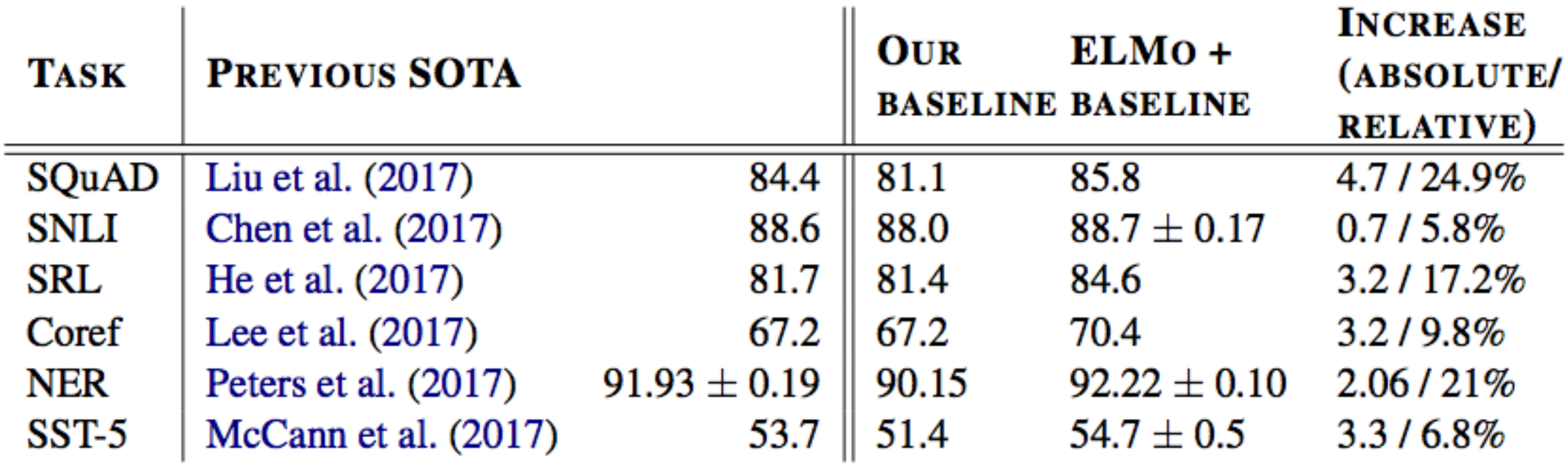
how many different embeddings does ELMo compute for a given word?

how to use ELMo in NLP tasks?



What additional parameters do we add to our model when using ELMo?

ELMo improves NLP tasks



Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/relative)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%