# Neural Machine Translation



Thang Luong

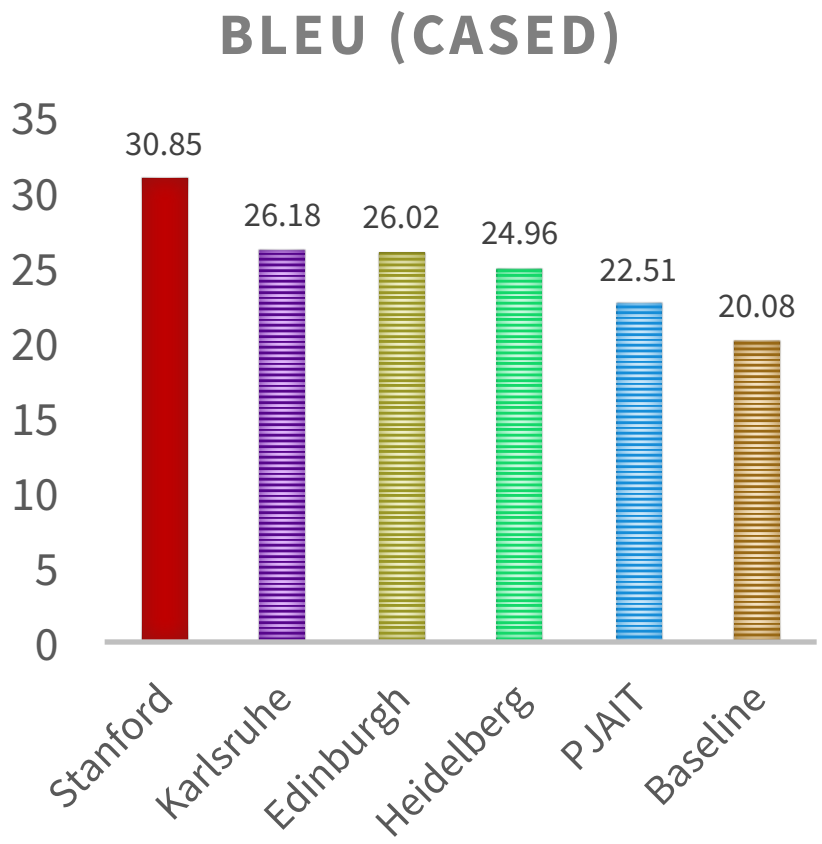
Kyunghyun Cho

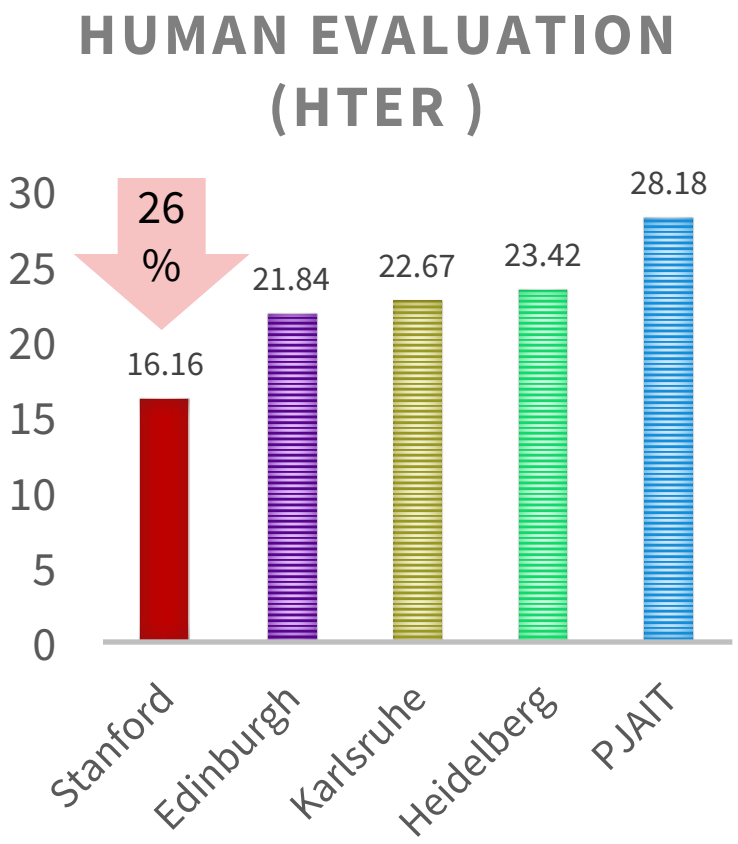
Christopher Manning

@lmthang · @kchonyc · @chrmanning

ACL 2016 tutorial · https://sites.google.com/site/acl16nmt/

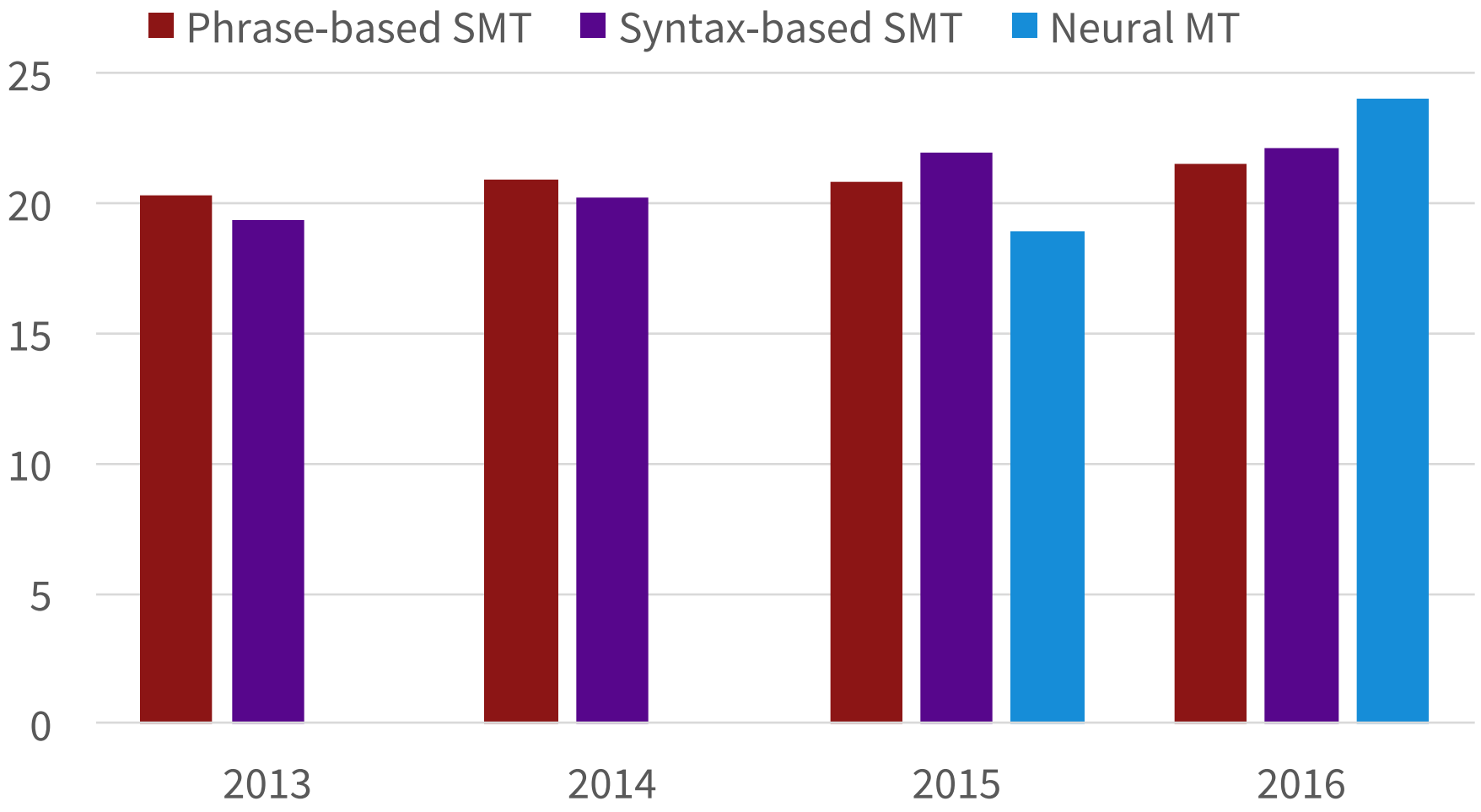
# IWSLT 2015, TED talk MT, English-German



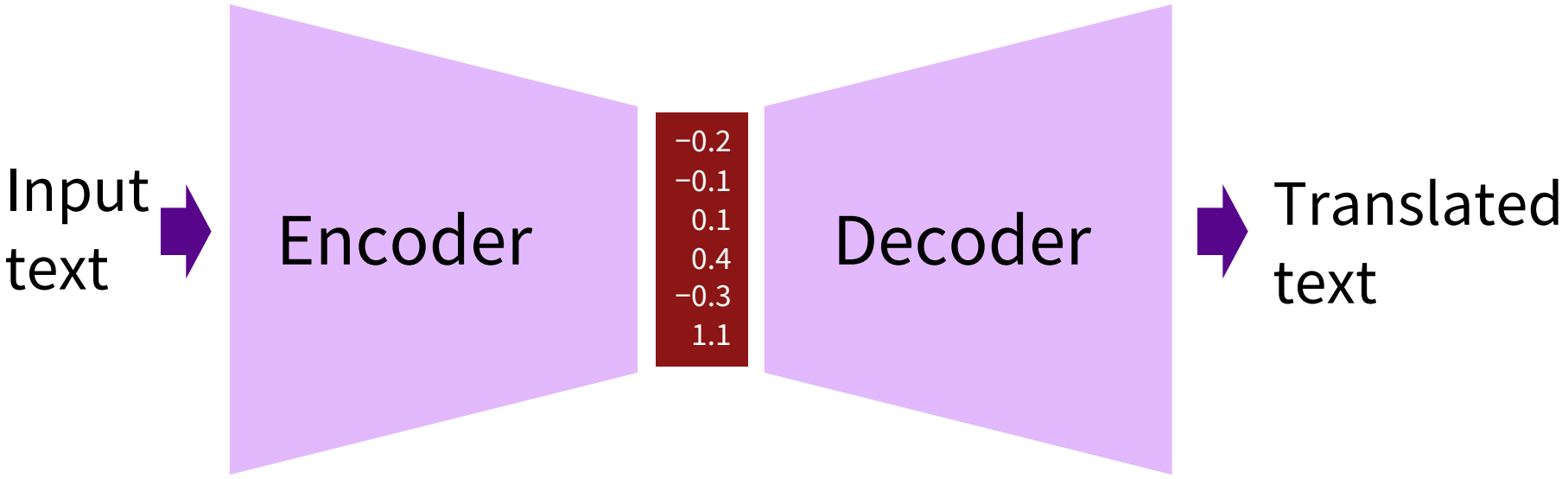
# Progress in Machine Translation

**[Edinburgh En-De WMT newstest2013 Cased BLEU; NMT 2015 from U. Montréal]**

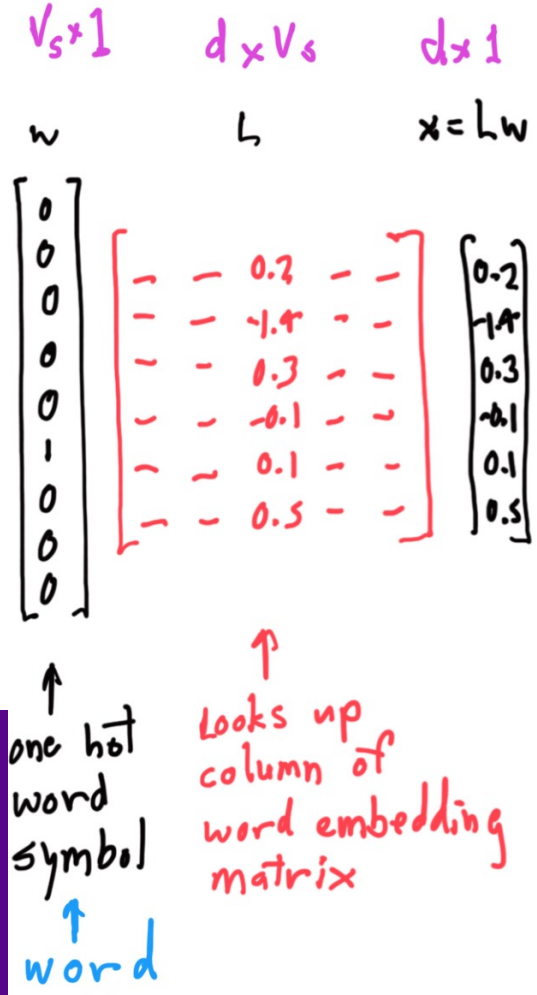



From [Sennrich 2016, http://www.meta-net.eu/events/meta-forum-2016/slides/09_sennrich.pdf]

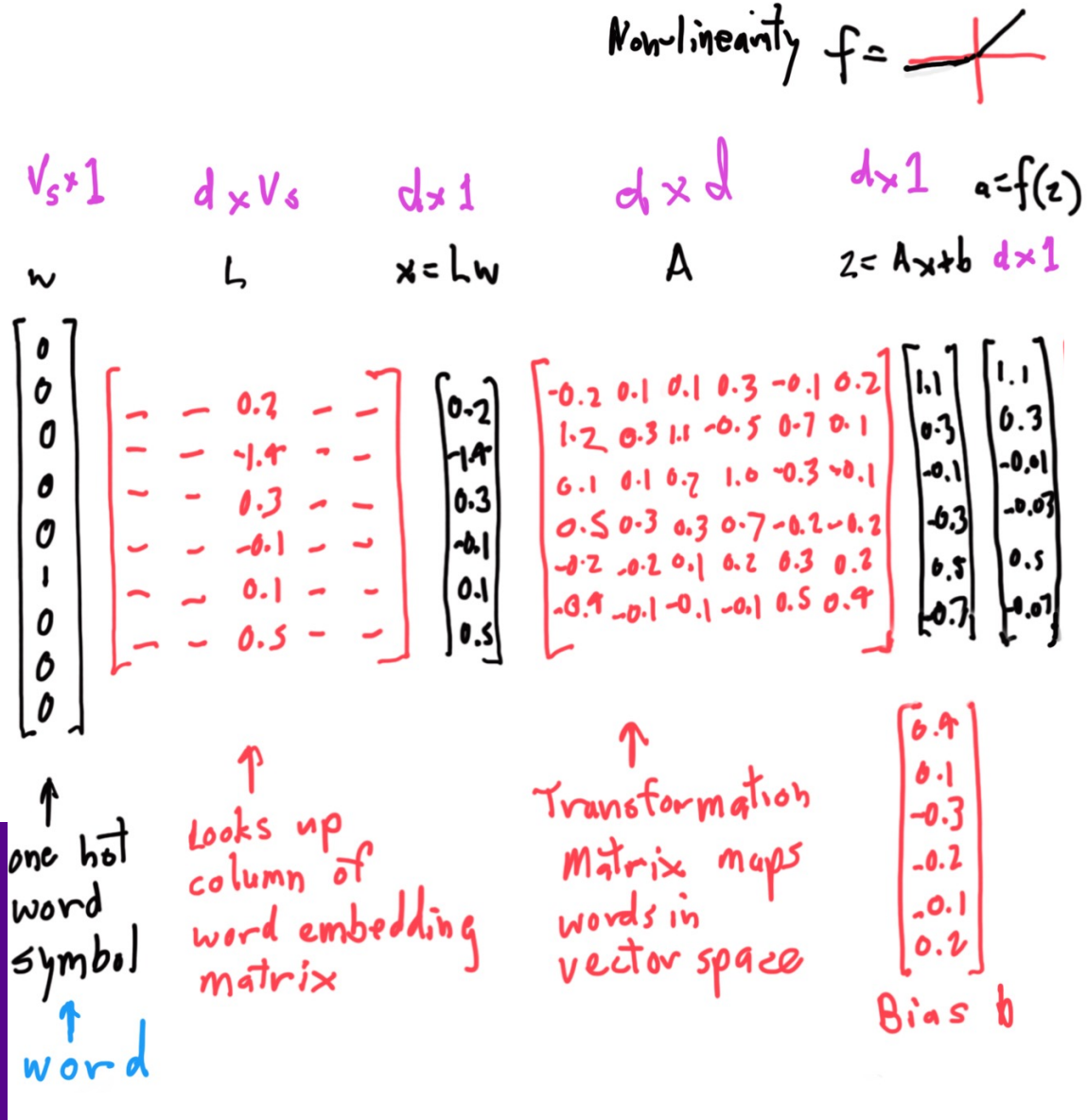
# Neural encoder-decoder architectures



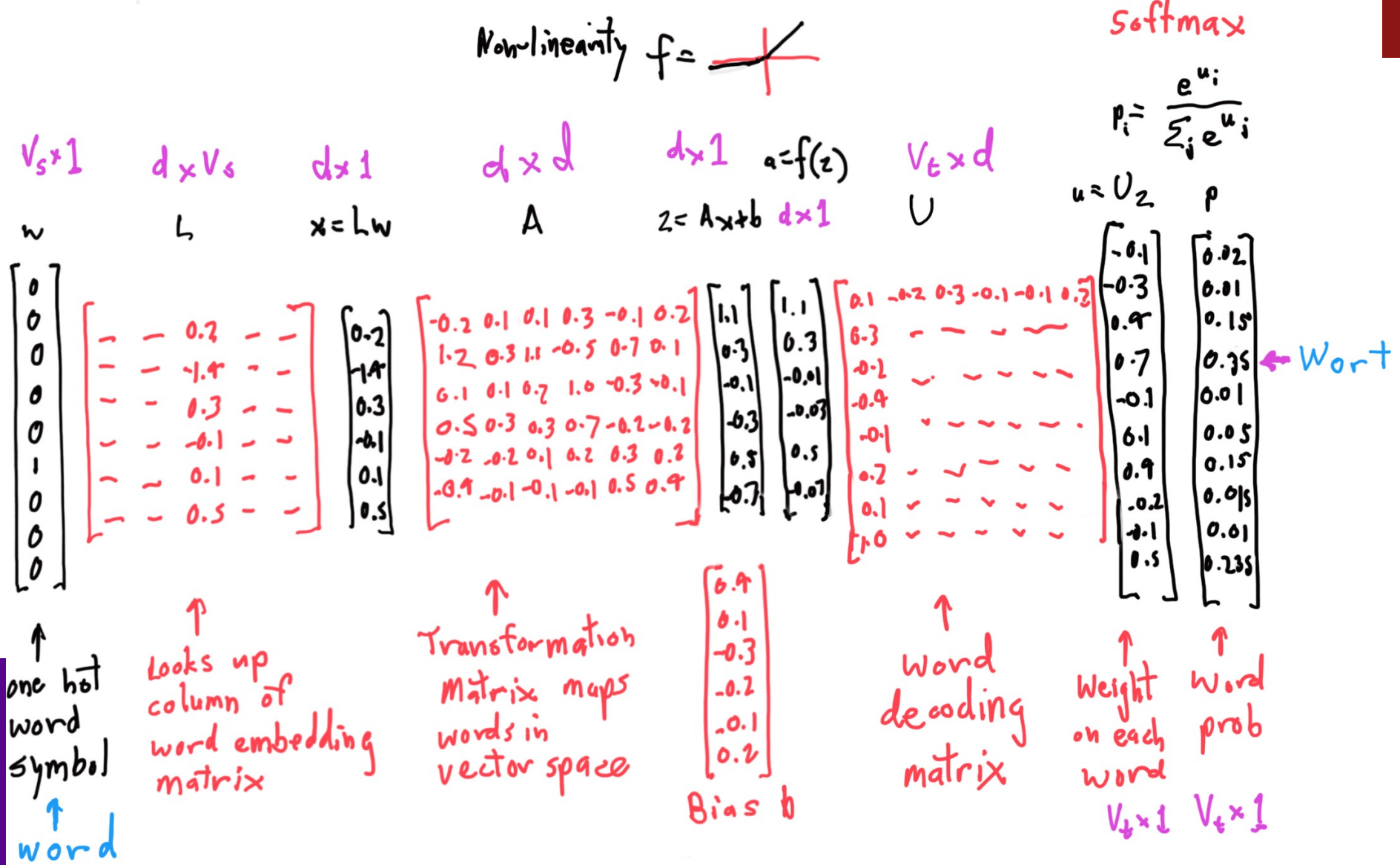
# NMT system for translating a single word



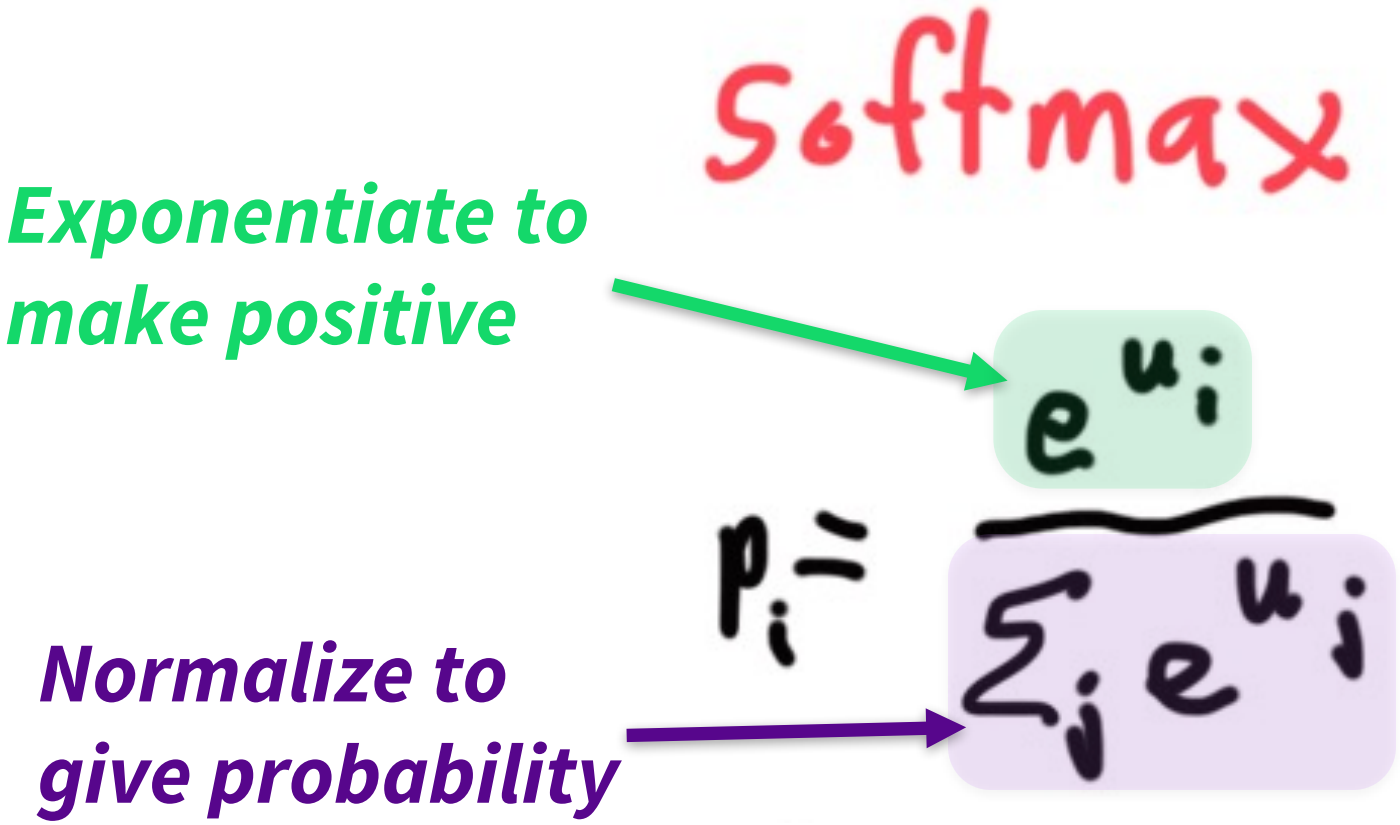
# NMT system for translating a single word



# NMT system for translating a single word



# Softmax function: Standard map from $\mathbb{R}^V$ to a probability distribution



# The three big wins of Neural MT

1. **End-to-end training**

   **All** parameters are simultaneously optimized to minimize a loss function on the network's output

2. **Distributed representations share strength**

   Better exploitation of word and phrase similarities

3. **Better exploitation of context**

   NMT can use a much bigger context – both source and partial target text – to translate more accurately

# A Non-Markovian Language Model

Can we directly model the true conditional probability?

$$p(x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1})$$

Can we build a neural language model for this?

1. Feature extraction: $h_t = f(x_1, x_2, \ldots, x_t)$
2. Prediction: $p(x_{t+1} | x_1, \ldots, x_{t-1}) = g(h_t)$

How can $f$ take a variable-length input?

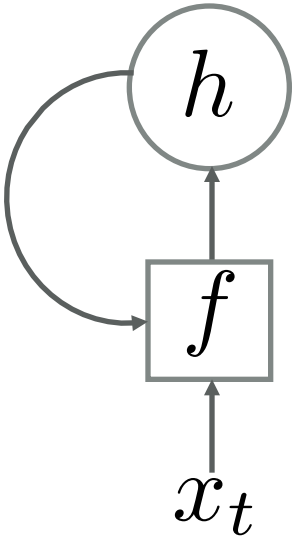
# A Non-Markovian Language Model

Can we directly model the true conditional probability?

$$p(x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1})$$

Recursive construction of $f$

1. Initialization $h_0 = 0$
2. Recursion $h_t = f(x_t, h_{t-1})$




We call $h_t$ a hidden state or memory

$h_t$ summarizes the history $(x_1, \ldots, x_t)$

# A Non-Markovian Language Model

*Example:* $p(\text{the}, \text{cat}, \text{is}, \text{eating})$

(1) Initialization: $h_0 = 0$

(2) Recursion with Prediction:

$$h_1 = f(h_0, \langle \text{bos} \rangle) \rightarrow p(\text{the}) = g(h_1)$$
$$h_2 = f(h_1, \text{cat}) \rightarrow p(\text{cat}|\text{the}) = g(h_2)$$
$$h_3 = f(h_2, \text{is}) \rightarrow p(\text{is}|\text{the}, \text{cat}) = g(h_3)$$
$$h_4 = f(h_3, \text{eating}) \rightarrow p(\text{eating}|\text{the}, \text{cat}, \text{is}) = g(h_4)$$

(3) Combination:

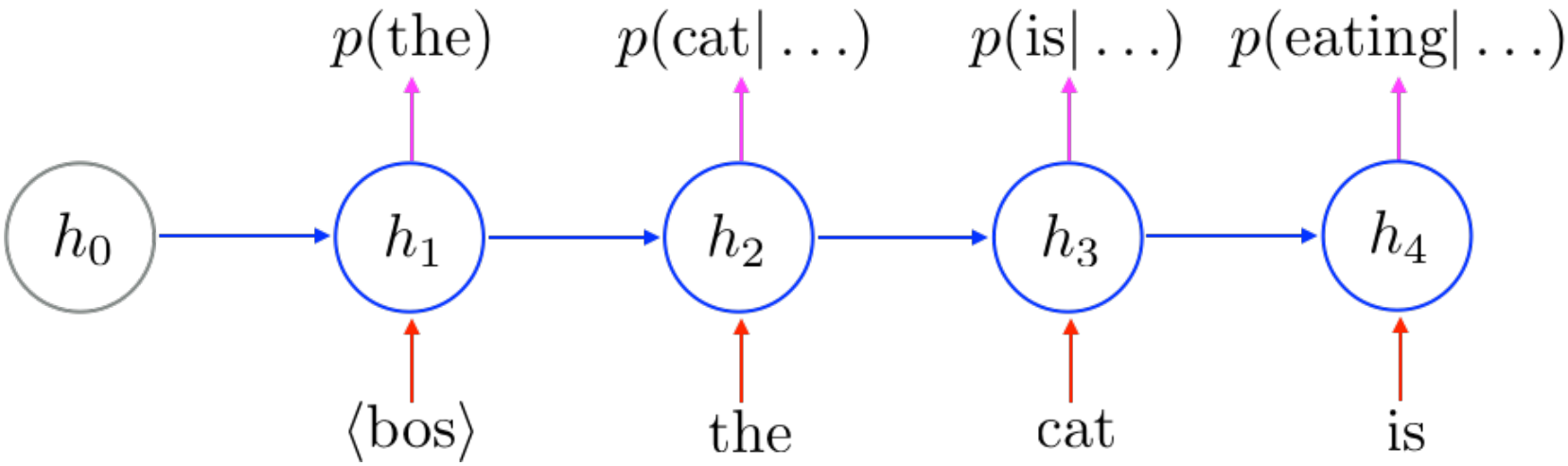
$$p(\text{the}, \text{cat}, \text{is}, \text{eating}) = g(h_1)g(h_2)g(h_3)g(h_4)$$



*Read, Update and Predict*

# A Recurrent Neural Network Language Model solves the second problem!

*Example:* $p(\text{the}, \text{cat}, \text{is}, \text{eating})$




*Read*, *Update* and *Predict*

# Building a Recurrent Language Model

*Transition Function* $h_t = f(h_{t-1}, x_t)$

Inputs

i. Current word $x_t \in \{1, 2, \ldots, |V|\}$

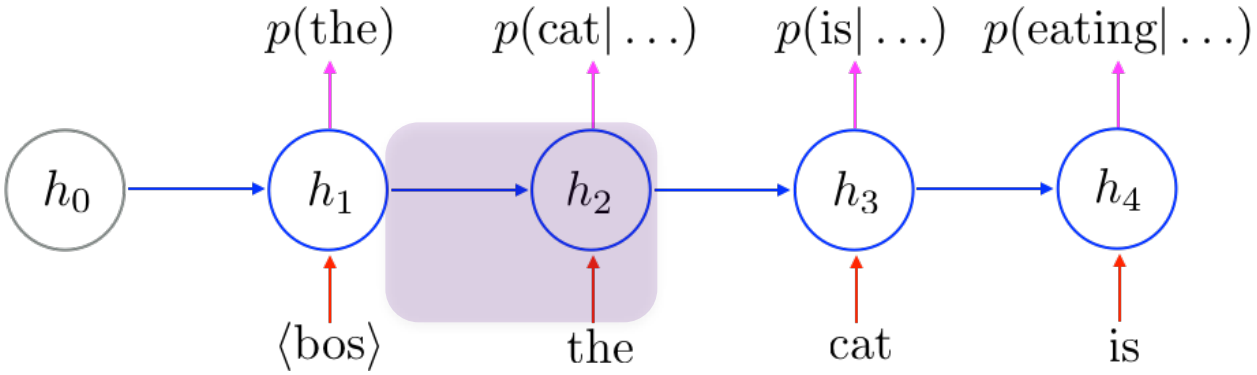
ii. Previous state $h_{t-1} \in \mathbb{R}^d$

Parameters

i. Input weight matrix $W \in \mathbb{R}^{|V| \times d}$

ii. Transition weight matrix $U \in \mathbb{R}^{d \times d}$

iii. Bias vector $b \in \mathbb{R}^d$



# Building a Recurrent Language Model

*Transition Function* $h_t = f(h_{t-1}, x_t)$

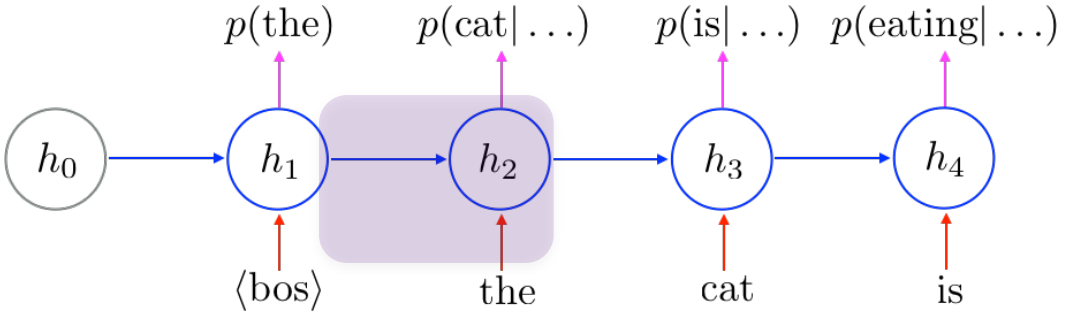
Naïve Transition Function

$$f(h_{t-1}, x_t) = \tanh(W\,[x_t] + Uh_{t-1} + b)$$

***Element-wise nonlinear transformation***

***Trainable word vector***

***Linear transformation of previous state***



# Building a Recurrent Language Model

*Prediction Function* $p(x_{t+1} = w | x_{\leq t}) = g_w(h_t)$

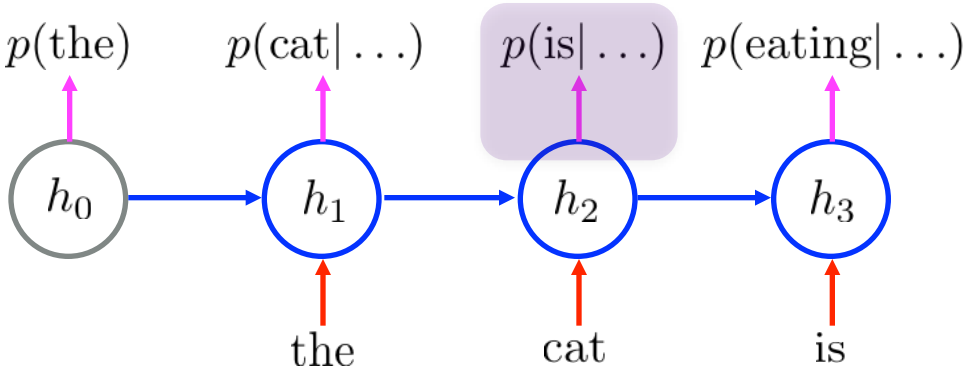
Inputs

i. Current state $h_t \in \mathbb{R}^d$

Parameters

i. Softmax matrix $R \in \mathbb{R}^{|V| \times d}$

ii. Bias vector $c \in \mathbb{R}^{|V|}$



# Building a Recurrent Language Model

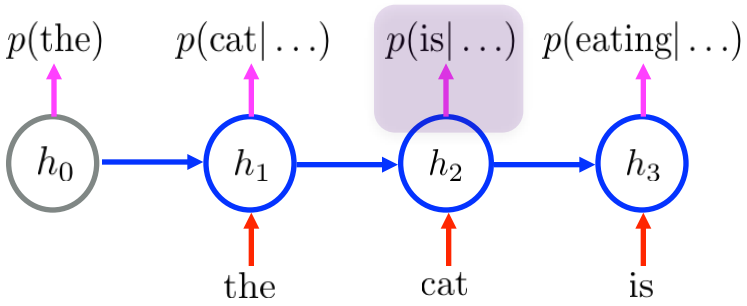
*Prediction Function* $p(x_{t+1} = w | x_{\leq t}) = g_w(h_t)$

$$p(x_{t+1} = w | x_{\leq t}) = g_w(h_t) = \frac{\exp(R[w]^\top h_t + c_w)}{\sum_{i=1}^{|V|} \exp(R[i]^\top h_t + c_i)}$$

***Compatibility between trainable word vector and hidden state***

***Exponentiate***

***Normalize***



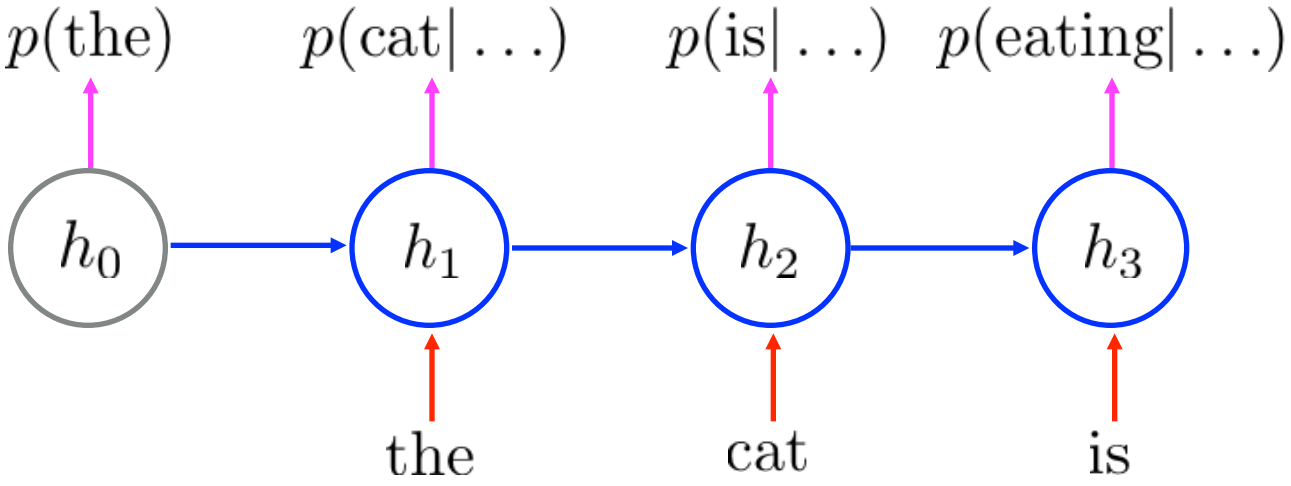
# Training a recurrent language model

Having determined the model form, we:

1. Initialize all parameters of the models, including the word representations with small random numbers
2. Define a loss function: how badly we predict actual next words [log loss or cross-entropy loss]
3. Repeatedly attempt to predict each next word
4. Backpropagate our loss to update **all** parameters
5. Just doing this learns good word representations and good prediction functions – it's almost magic

# Recurrent Language Model

*Example)* $p(\text{the}, \text{cat}, \text{is}, \text{eating})$




*Read*, *Update* and *Predict*

# Training a Recurrent Language Model

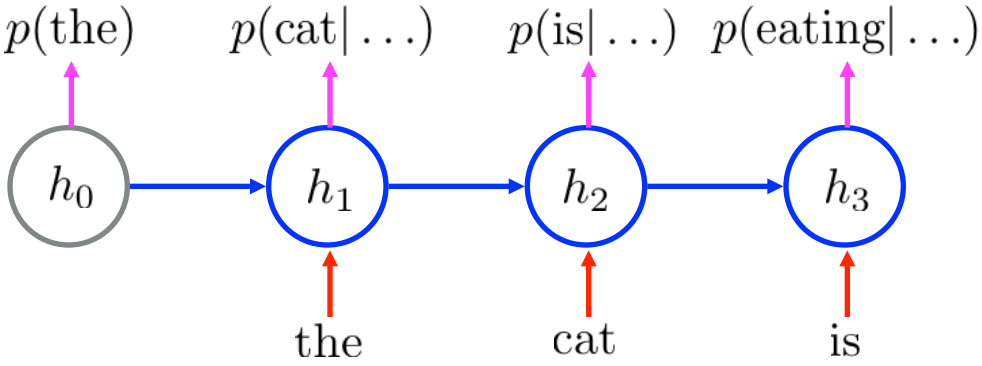
- Log-probability of one training sentence

$$\log p(x_1^n, x_2^n, \ldots, x_{T^n}^n) = \sum_{t=1}^{T^n} \log p(x_t^n | x_1^n, \ldots, x_{t-1}^n)$$

- Training set $D = \{X^1, X^2, \ldots, X^N\}$
- Log-likelihood Functional

$$\mathcal{L}(\theta, D) = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T^n} \log p(x_t^n | x_1^n, \ldots, x_{t-1}^n)$$

*Minimize* $-\mathcal{L}(\theta, D)$ *!!*

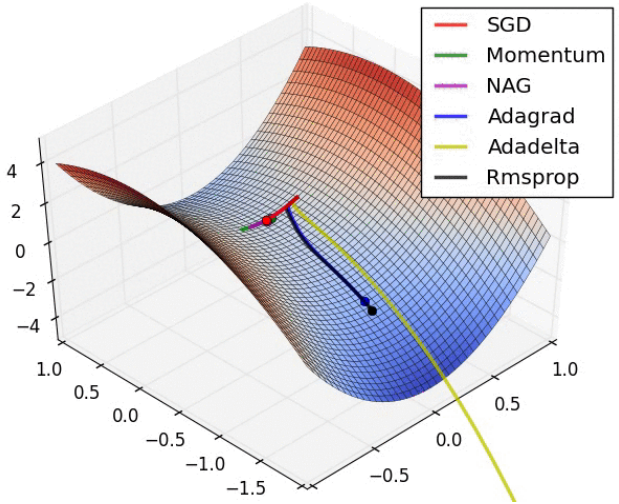


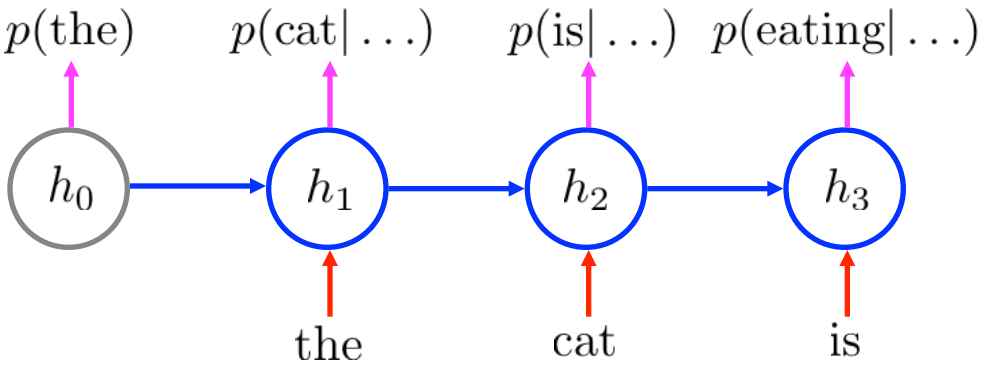
# Gradient Descent

- Move slowly in the steepest descent direction

$$\theta \leftarrow \theta - \eta \nabla \mathcal{L}(\theta, D)$$

- Computational cost of a single update: $O(N)$
- Not suitable for a large corpus



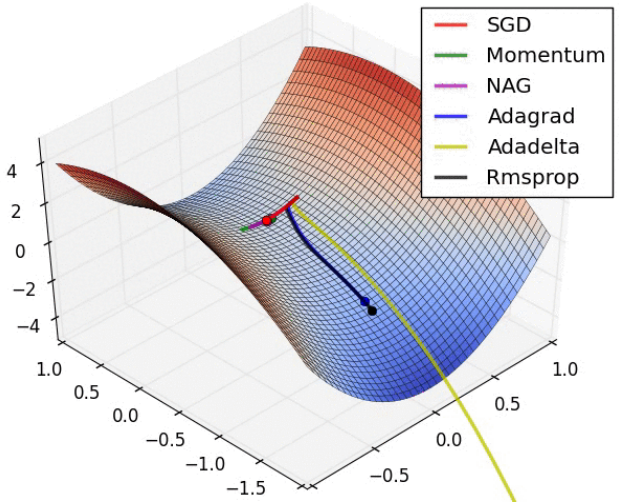
# Stochastic Gradient Descent

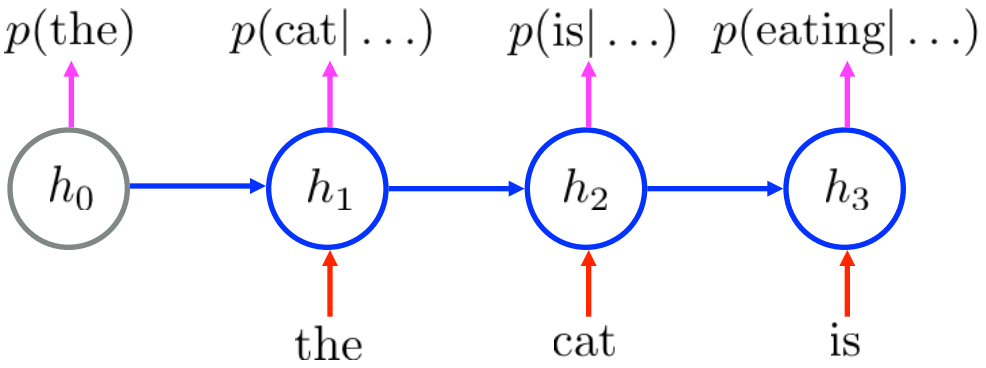
- Estimate the steepest direction with a minibatch

$$\nabla \mathcal{L}(\theta, D) \approx \nabla \mathcal{L}(\theta, \{X^1, \ldots, X^n\})$$

- Until the convergence (w.r.t. a validation set)

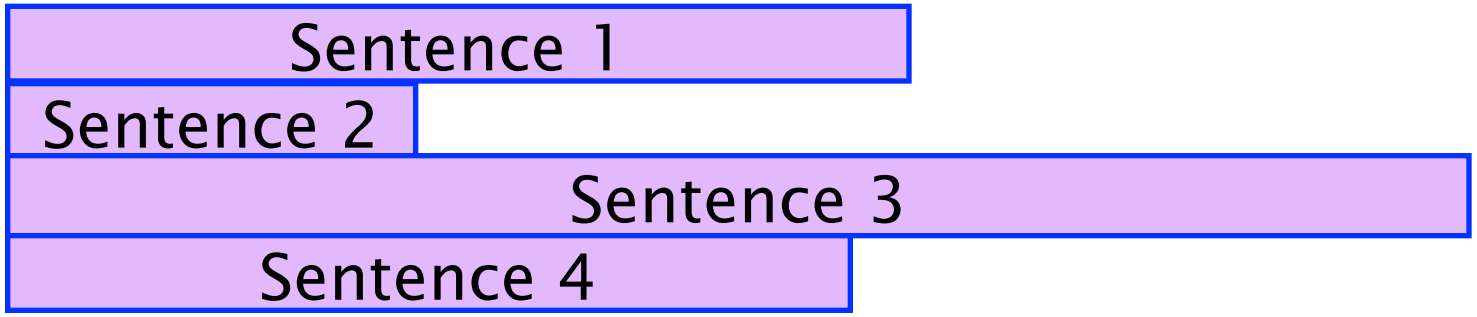
$$|\mathcal{L}(\theta, D_{\text{val}}) - \mathcal{L}(\theta - \eta \mathcal{L}(\theta, D), D_{\text{val}})| \leq \epsilon$$



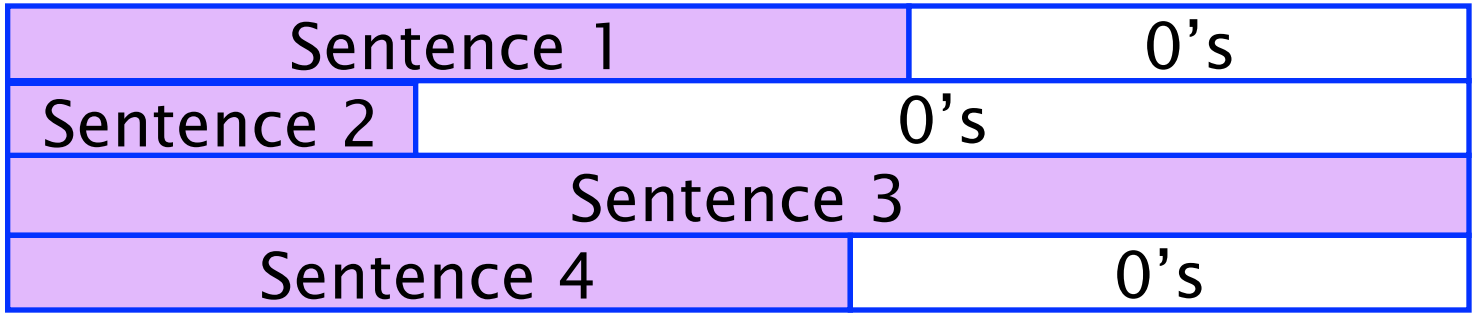



# Stochastic Gradient Descent

- *Not* trivial to build a minibatch

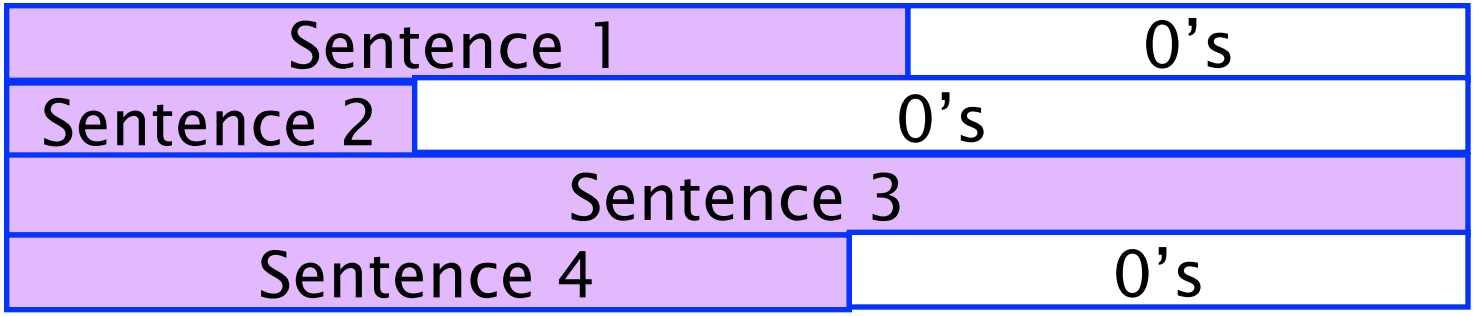



1. Padding and Masking: *suitable for GPU's, but wasteful*
   - *Wasted computation*

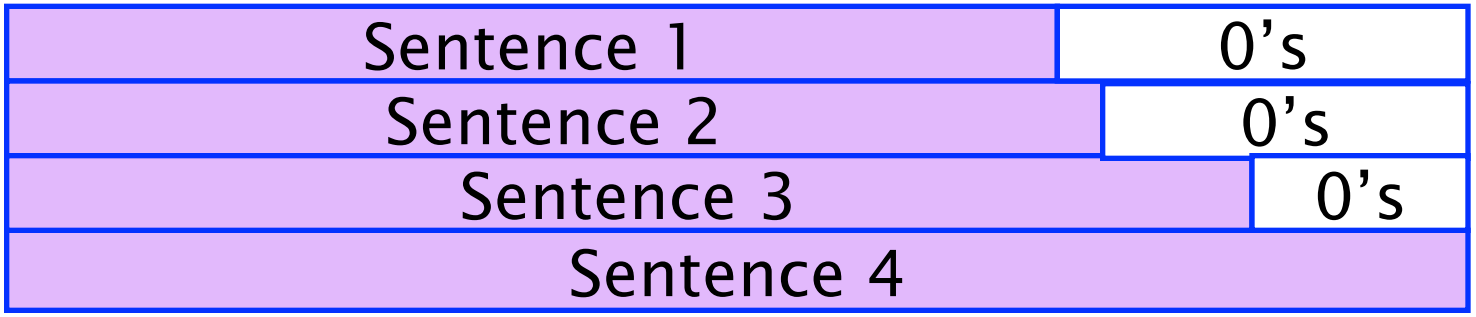


# Stochastic Gradient Descent

1. Padding and Masking: *suitable for GPU's, but wasteful*
   - *Wasted computation*




2. Smarter Padding and Masking: *minimize the waste*
   - *Ensure that the length differences are minimal.*
   - *Sort the sentences and sequentially build a minibatch*



# Backpropagation through Time

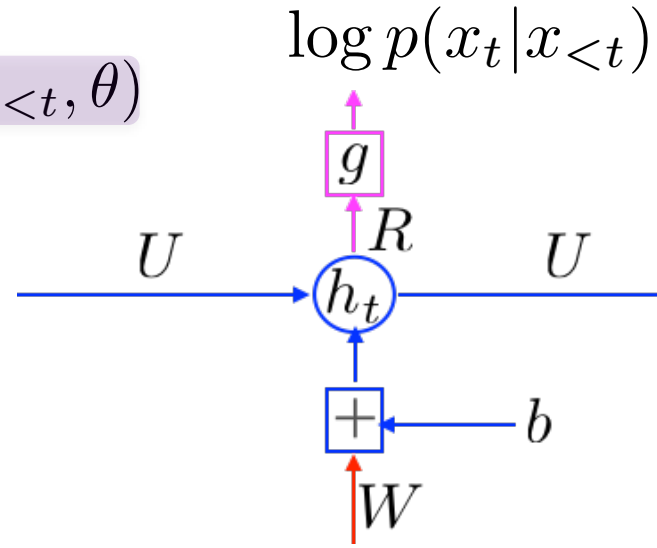
*How do we compute* $\nabla \mathcal{L}(\theta, D)$*?*

- Cost as a sum of per-sample cost function

$$\nabla \mathcal{L}(\theta, D) = \sum_{X \in D} \nabla \mathcal{L}(\theta, X)$$

- Per-sample cost as a sum of per-step cost functions

$$\nabla \mathcal{L}(\theta, X) = \sum_{t=1}^{T} \nabla \log p(x_t | x_{<t}, \theta)$$

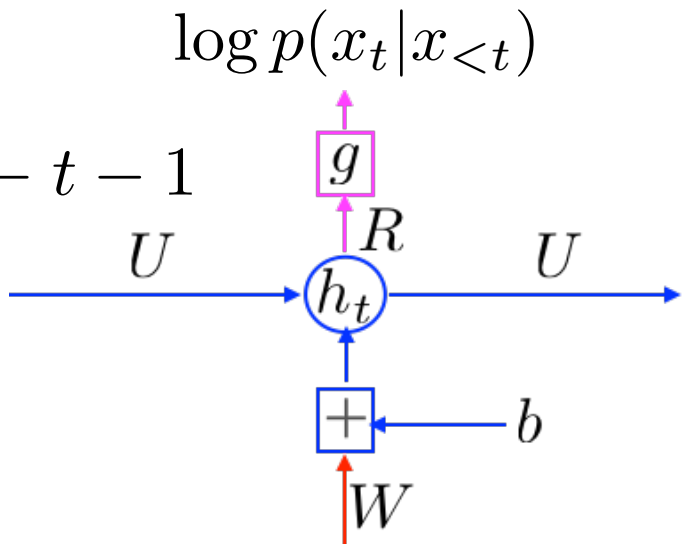


# Backpropagation through Time

*How do we compute* $\nabla \log p(x_t|x_{<t}, \theta)$*?*

- Compute per-step cost function from time $t = T$

1. Cost derivative $\partial \log p(x_t|x_{<t})/\partial g$
2. Gradient w.r.t. $R$ : $\times \partial g/\partial R$
3. Gradient w.r.t. $h_t$ : $\times \partial g/\partial h_t + \partial h_{t+1}/\partial h_t$
4. Gradient w.r.t. $U$ : $\times \partial h_t/\partial U$
5. Gradient w.r.t. $b$ and $W$ : $\times \partial h_t/\partial b$ and $\times \partial h_t/\partial W$
6. Accumulate the gradient and $t \leftarrow t - 1$




Note: I'm abusing math a lot here!!

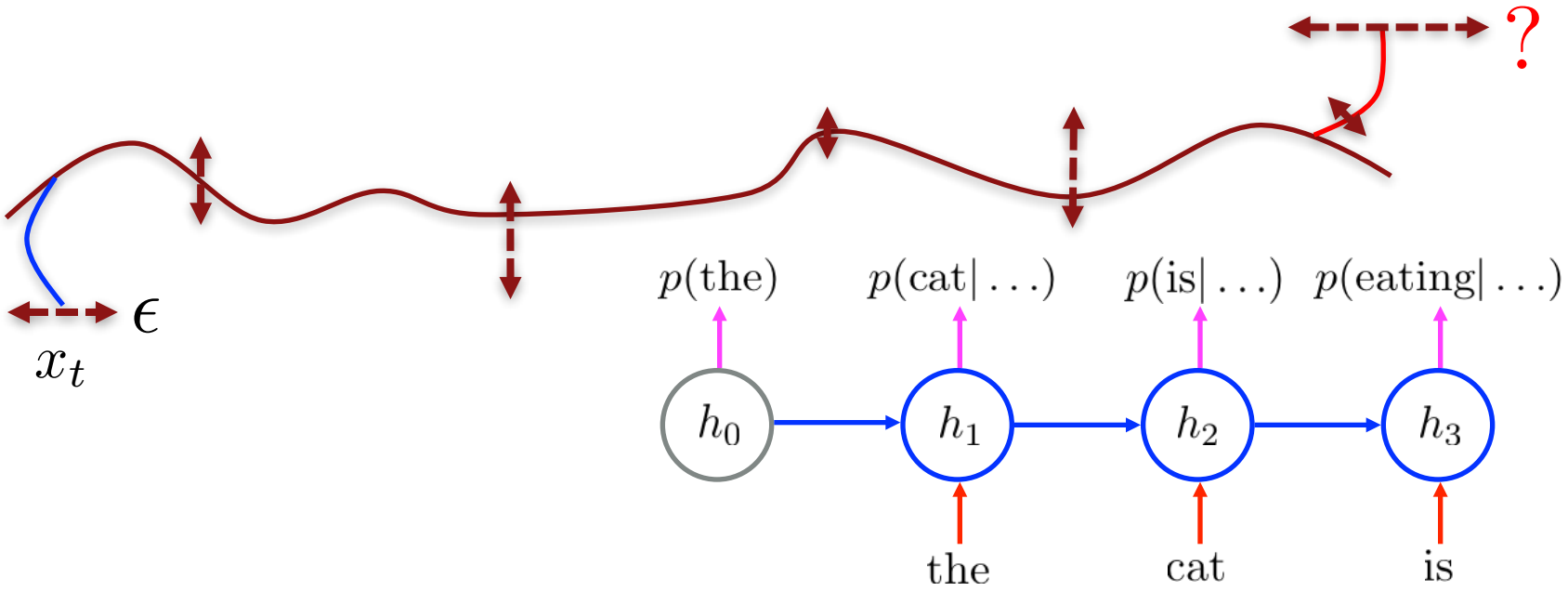
# Backpropagation through Time

*Intuitively, what's happening here?*

1. Measure the influence of the past on the future

$$\frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial h_t} = \frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial g} \frac{\partial g}{\partial h_{t+n}} \frac{\partial h_{t+n}}{\partial h_{t+n-1}} \cdots \frac{\partial h_{t+1}}{\partial h_t}$$

2. How does the perturbation at $t$ affect $p(x_{t+n}|x_{<t+n})$?



# Backpropagation through Time

*Intuitively, what's happening here?*

1. Measure the influence of the past on the future

$$\frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial h_t} = \frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial g} \frac{\partial g}{\partial h_{t+n}} \frac{\partial h_{t+n}}{\partial h_{t+n-1}} \cdots \frac{\partial h_{t+1}}{\partial h_t}$$

2. How does the perturbation at $t$ affect $p(x_{t+n}|x_{<t+n})$?


3. Change the parameters to maximize $p(x_{t+n}|x_{<t+n})$

# Backpropagation through Time

*Intuitively, what's happening here?*

1. Measure the influence of the past on the future

$$\frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial h_t} = \frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial g} \frac{\partial g}{\partial h_{t+n}} \frac{\partial h_{t+n}}{\partial h_{t+n-1}} \cdots \frac{\partial h_{t+1}}{\partial h_t}$$

2. With a naïve transition function

$$f(h_{t-1}, x_{t-1}) = \tanh(W\,[x_{t-1}] + U h_{t-1} + b)$$

We get $$\frac{\partial J_{t+n}}{\partial h_t} = \frac{\partial J_{t+n}}{\partial g} \frac{\partial g}{\partial h_{t+N}} \underbrace{\prod_{n=1}^{N} U^\top \operatorname{diag}\left(\frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}}\right)}$$

***Problematic!***



[Bengio, IEEE 1994]

# Backpropagation through Time

*Gradient either **vanishes** or **explodes***

- What happens?

$$\frac{\partial J_{t+n}}{\partial h_t} = \frac{\partial J_{t+n}}{\partial g} \frac{\partial g}{\partial h_{t+N}} \underbrace{\prod_{n=1}^{N} U^\top \operatorname{diag}\left(\frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}}\right)}$$

1. The gradient *likely* explodes if

$$e_{\max} \geq \frac{1}{\max \tanh'(x)} = 1$$

2. The gradient *likely* vanishes if

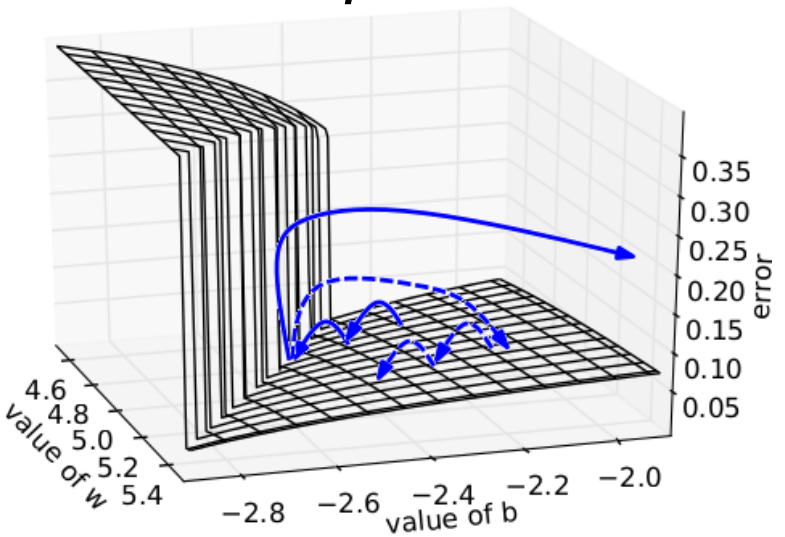
$$e_{\max} < \frac{1}{\max \tanh'(x)} = 1, \text{where} \quad e_{\max} : \text{largest eigenvalue of } U$$

[Bengio, Simard, Frasconi, TNN1994; Hochreiter, Bengio, Frasconi, Schmidhuber, 2001]

# Backpropagation through Time

*Addressing Exploding Gradient*

- *"when gradients explode so does the curvature along v, leading to a wall in the error surface"*




- Gradient Clipping
  1. Norm clipping

$$\tilde{\nabla} \leftarrow \begin{cases} \frac{c}{\|\nabla\|}\nabla & \text{,if } \|\nabla\| \geq c \\ \nabla & \text{,otherwise} \end{cases}$$

  2. Element-wise clipping

$$\nabla_i \leftarrow \min(c, |\nabla_i|)\text{sgn}(\nabla_i), \text{ for all } i \in \{1, \ldots, \dim \nabla\}$$



[Pascanu, Mikolov, Bengio, ICML 2013]

# Backpropagation through Time

*Vanishing gradient is super-problematic*

- When we only observe

$$\left\| \frac{\partial h_{t+N}}{\partial h_t} \right\| = \left\| \prod_{n=1}^{N} U^\top \operatorname{diag}\left( \frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}} \right) \right\| \to 0 \ ,$$

- We cannot tell whether
  1. No dependency between *t* and *t+n* in data, or
  2. Wrong configuration of parameters:

$$e_{\max}(U) < \frac{1}{\max \tanh'(x)}$$

# Gated Recurrent Unit

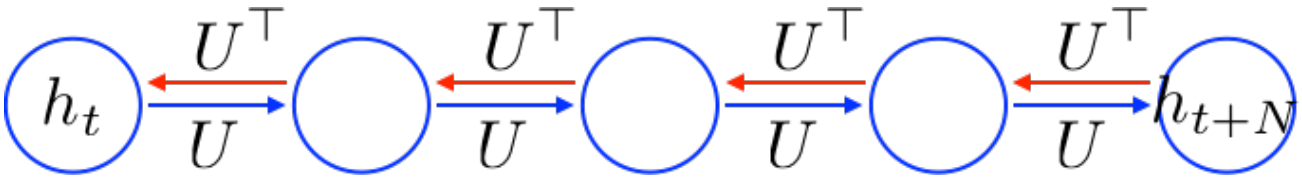
- Is the problem with the naïve transition function?

$$f(h_{t-1}, x_t) = \tanh(W\,[x_t] + U h_{t-1} + b)$$

- With it, the temporal derivative is

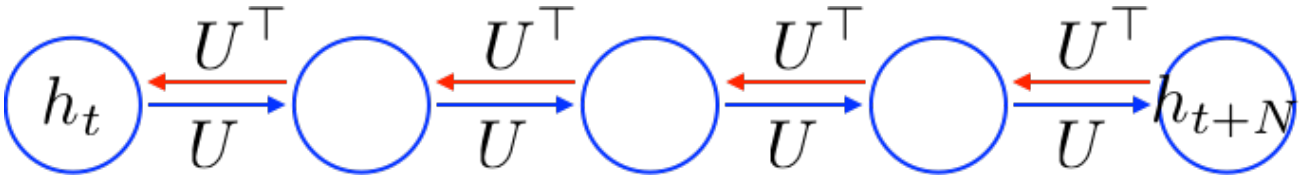
$$\frac{\partial h_{t+1}}{\partial h_t} = U^\top \frac{\partial \tanh(a)}{\partial a}$$

- It implies that the error must be backpropagated through all the intermediate nodes:

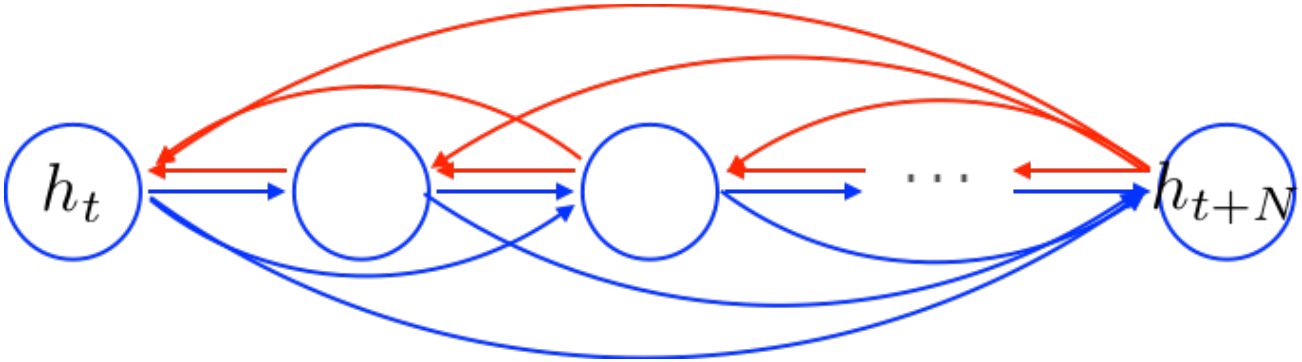


# Gated Recurrent Unit

- It implies that the error must backpropagate through all the intermediate nodes:

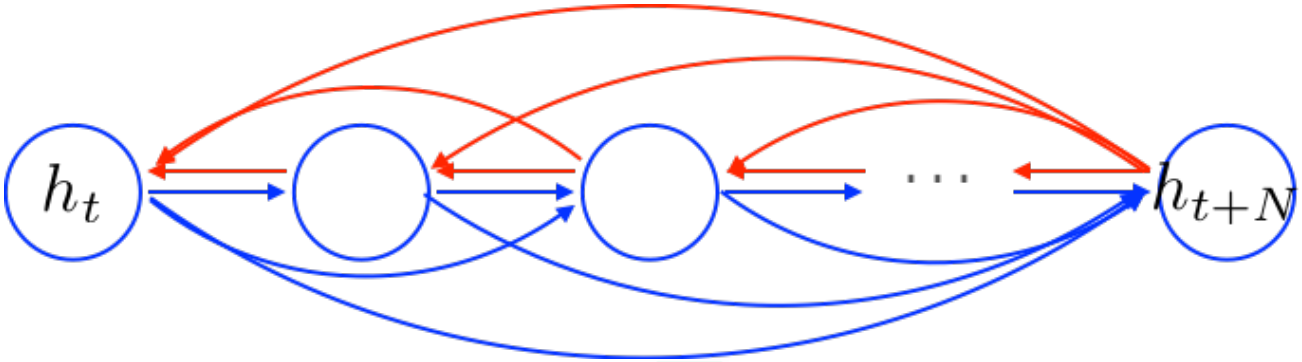



- Perhaps we can create shortcut connections.



# Gated Recurrent Unit

- Perhaps we can create *adaptive* shortcut connections.




$$f(h_{t-1}, x_t) = u_t \odot \tilde{h}_t + (1 + u_t) \odot h_{t-1}$$

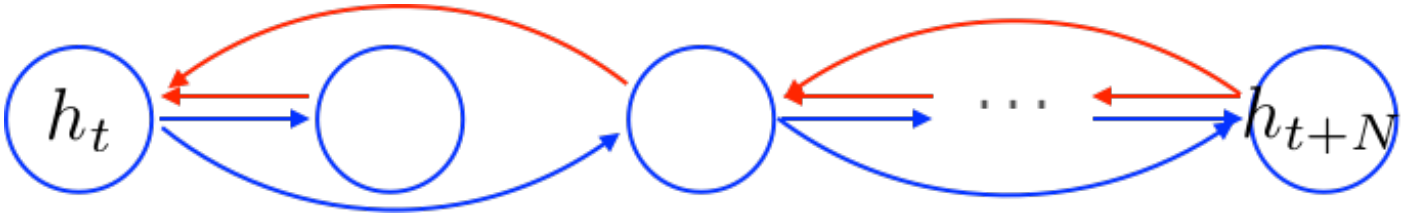
- Candidate Update $\tilde{h}_t = \tanh(W\,[x_t] + U h_{t-1} + b)$
- Update gate $u_t = \sigma(W_u\,[x_t] + U_u h_{t-1} + b_u)$



$\odot$: element-wise multiplication

# Gated Recurrent Unit

- Let the net prune unnecessary connections *adaptively*.




$$f(h_{t-1}, x_t) = u_t \odot \tilde{h}_t + (1 + u_t) \odot h_{t-1}$$

- Candidate Update $\tilde{h}_t = \tanh(W\,[x_t] + U(r_t \odot h_{t-1}) + b)$
- Reset gate $r_t = \sigma(W_r\,[x_t] + U_r h_{t-1} + b_r)$
- Update gate $u_t = \sigma(W_u\,[x_t] + U_u h_{t-1} + b_u)$

# Gated Recurrent Unit

*Two most widely used gated recurrent units*

## Gated Recurrent Unit

**[Cho et al., EMNLP2014; Chung, Gulcehre, Cho, Bengio, DLUFL2014]**

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

$$\tilde{h} = \tanh(W\,[x_t] + U(r_t \odot h_{t-1}) + b)$$

$$u_t = \sigma(W_u\,[x_t] + U_u h_{t-1} + b_u)$$

$$r_t = \sigma(W_r\,[x_t] + U_r h_{t-1} + b_r)$$

## Long Short-Term Memory

**[Hochreiter&Schmidhuber, NC1999; Gers, Thesis2001]**

$$h_t = o_t \odot \tanh(c_t)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

$$\tilde{c}_t = \tanh(W_c\,[x_t] + U_c h_{t-1} + b_c)$$

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

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

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

# Training an RNN

*A few well-established + my personal wisdoms*

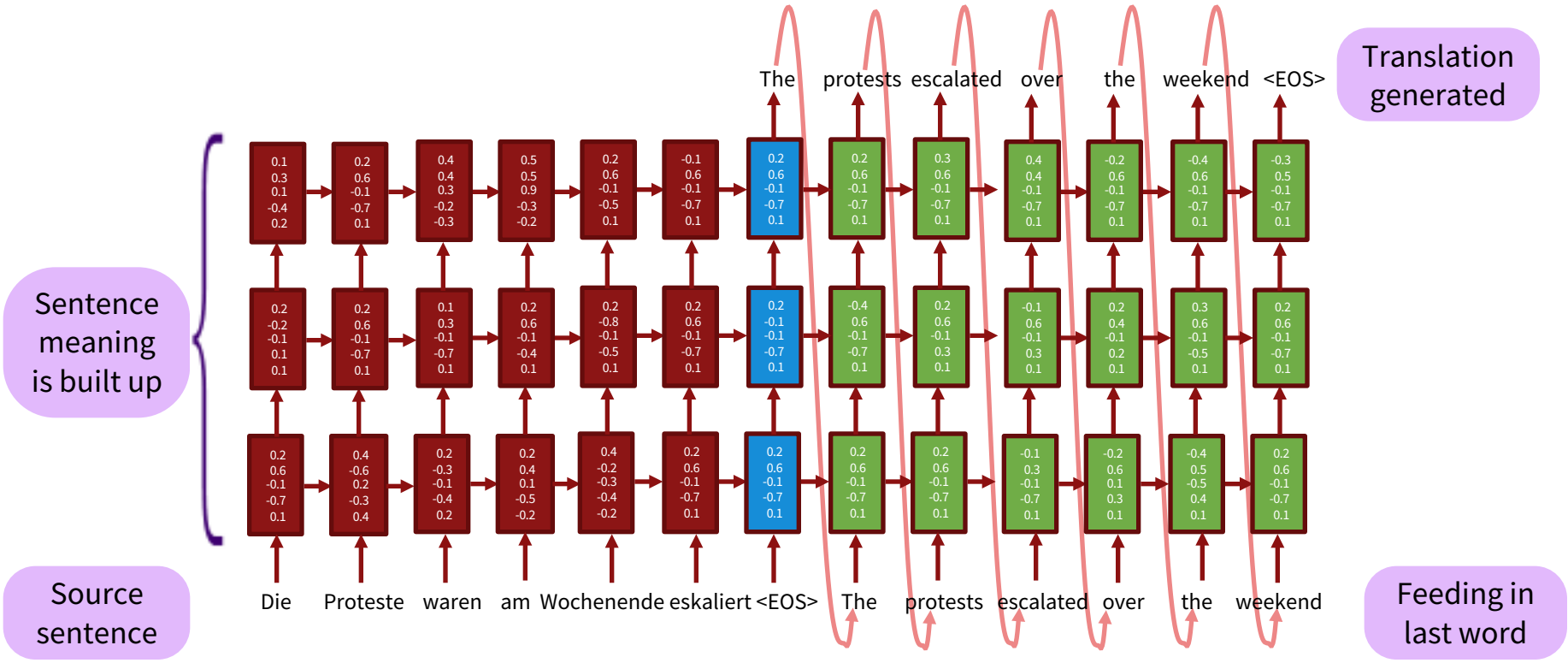
1. Use LSTM or GRU: *makes your life so much simpler*
2. Initialize recurrent matrices to be orthogonal
3. Initialize other matrices with a sensible scale
4. Use adaptive learning rate algorithms: *Adam, Adadelta, …*
5. Clip the norm of the gradient: "*1" seems to be a reasonable threshold when used together with adam or adadelta.*
6. *Be patient!*

[Saxe et al., ICLR2014;
Ba, Kingma, ICLR2015;
Zeiler, arXiv2012;
Pascanu et al., ICML2013]

# Modern Sequence Models for NMT

## [Sutskever et al. 2014, Bahdanau et al. 2014, et seq.]

**following [Jordan 1986] and more closely [Elman 1990]**




A deep recurrent neural network

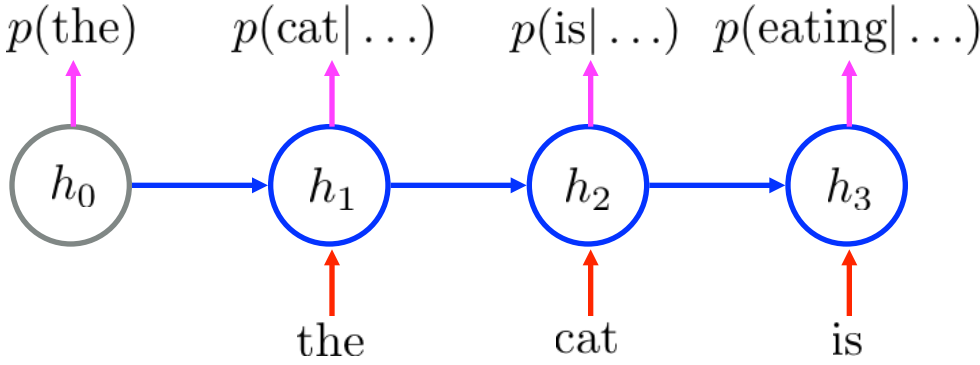
# Recurrent Language Model *can*

1. Score a given sentence very well

$$\log p(\text{the}, \text{cat}, \text{is}, \text{sitting}, \text{on}, \text{a}, \text{couch}, .)$$

   - Mere reranking significantly improves machine translation and speech recognition quality [Schwenk, 2007; Schwenk, 2012]
   - Very good at sentence completion without much task-specific engineering [Tran, …, Monz, NAACL 2016]

2. Generate a long, coherent text
   - Observed earlier by Mikolov [2010, in his thesis] and Sutskever et al. [2011]



# Conditional Recurrent Language Model

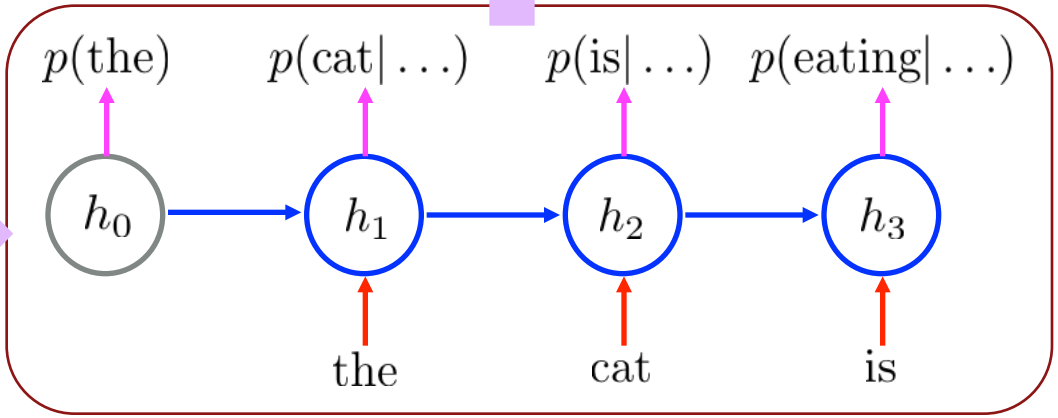
Le chat assis sur le tapis.

?

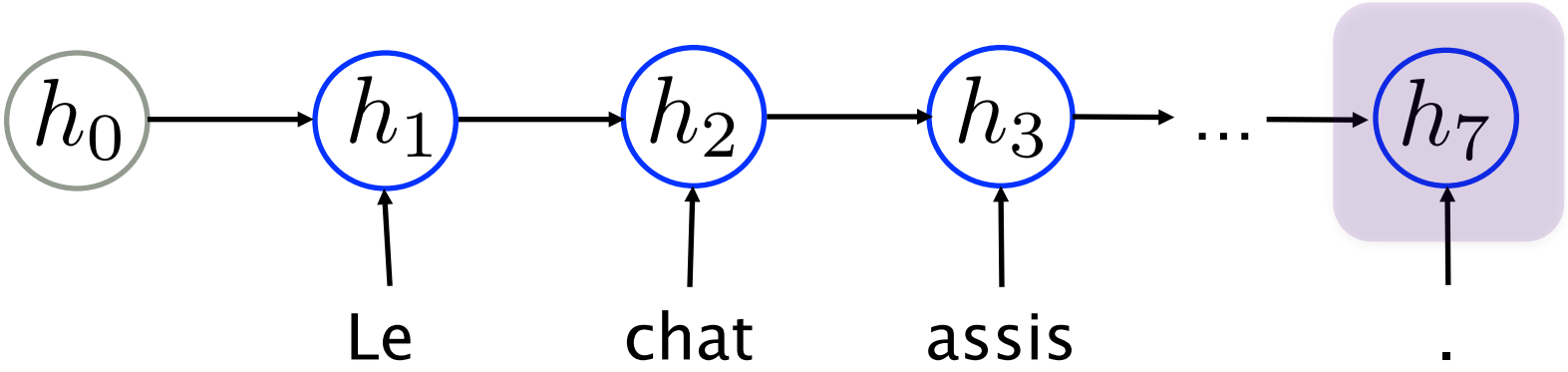
The cat sat on the mat.

Encoder

$Y$

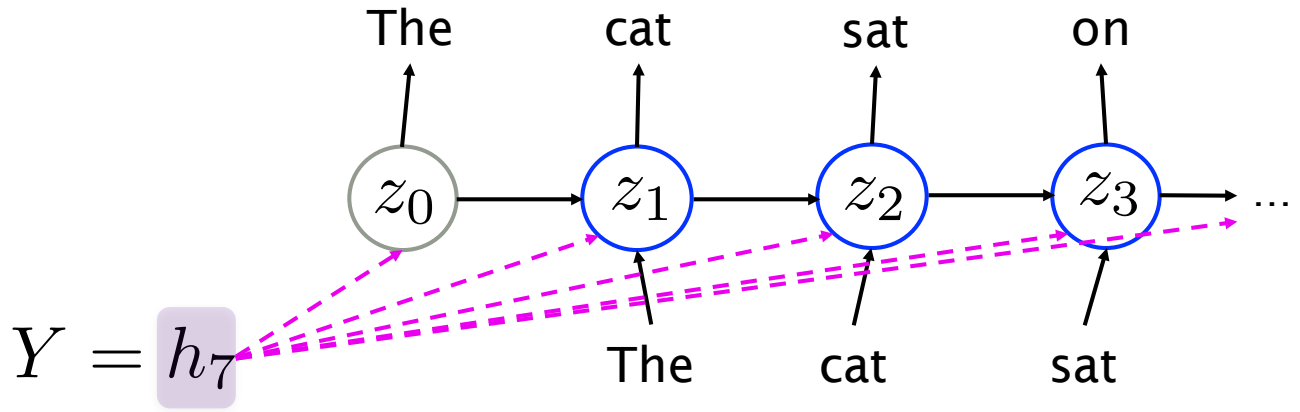


# Recurrent Neural Network Encoder




- Read a source sentence one symbol at a time.
- The last hidden state $Y$ summarizes the entire source sentence.
- Any recurrent activation function can be used:
  - Hyperbolic tangent $\tanh$
  - Gated recurrent unit [Cho et al., 2014]
  - Long short-term memory [Sutskever et al., 2014]
  - Convolutional network [Kalchbrenner&Blunsom, 2013]

# Decoder: Recurrent Language Model



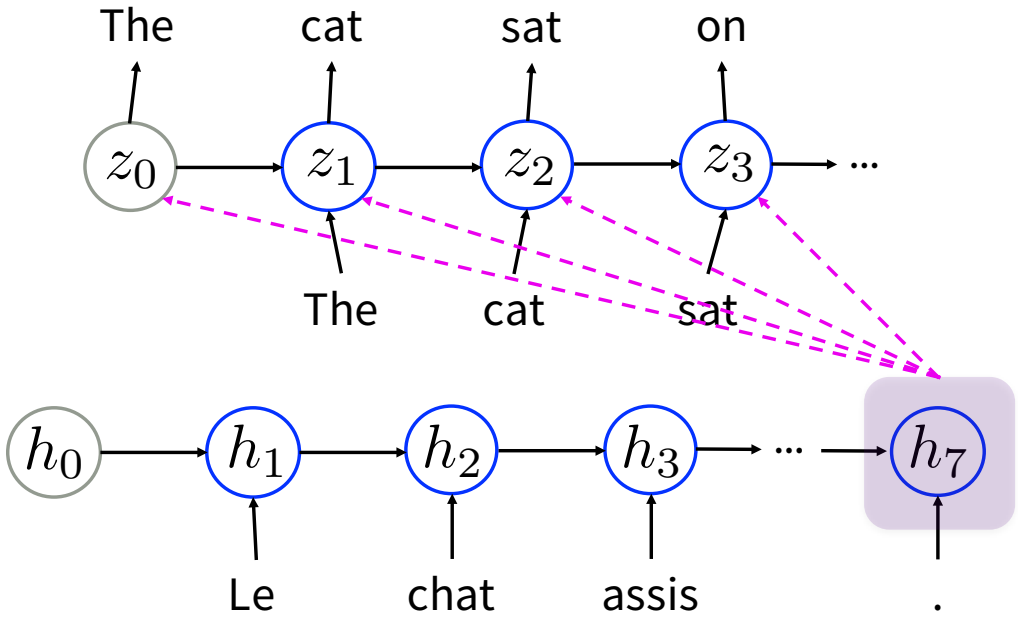

- Usual recurrent language model, except
  1. Transition $z_t = f(z_{t-1}, x_t, Y)$
  2. Backpropagation $\sum_t \partial z_t / \partial Y$
- Same learning strategy as usual: MLE with SGD

$$\mathcal{L}(\theta, D) = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T^n} \log p(x_t^n | x_1^n, \ldots, x_{t-1}^n, Y)$$

# Decoding (0) – Exhaustive Search

- Simple and exact decoding algorithm
- Score each and every possible translation
- Pick the best one

***DO NOT EVEN THINK of TRYING IT OUT!****

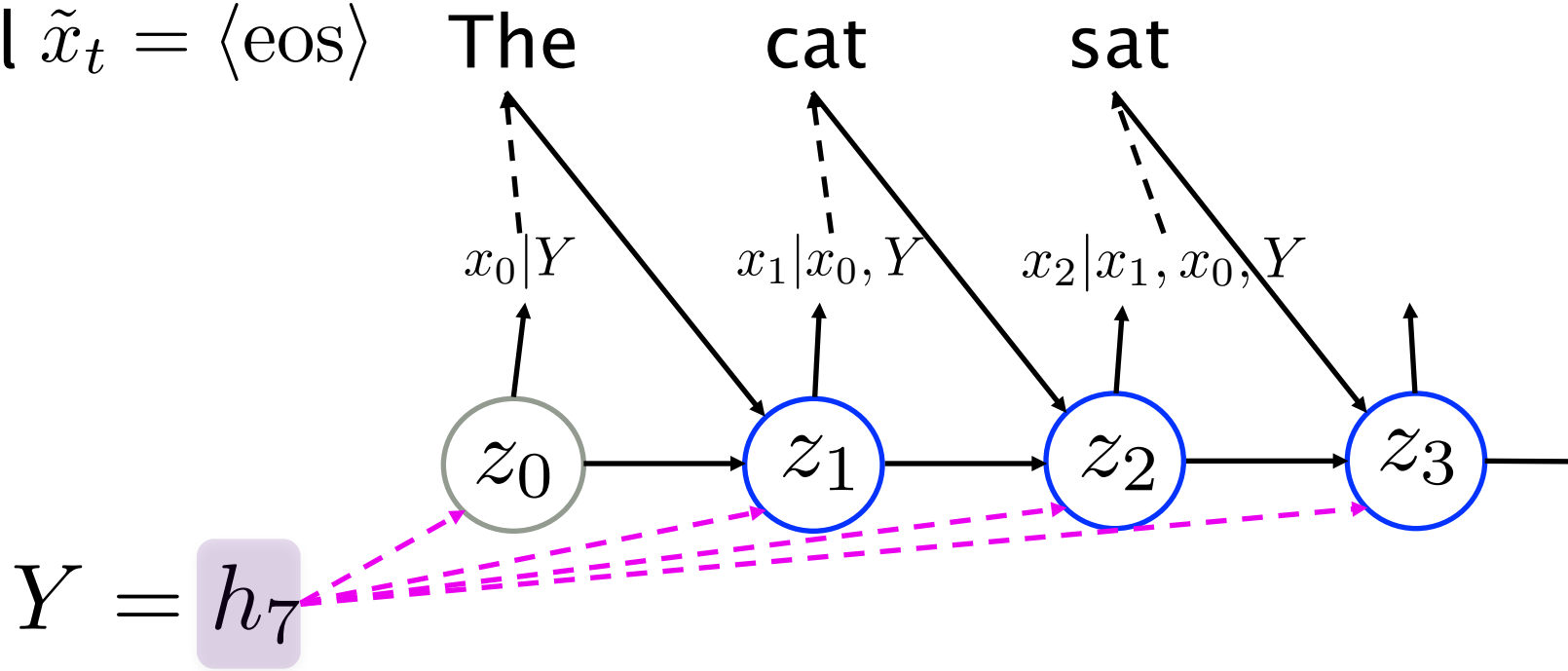





* Perhaps with quantum computer and quantum annealing?

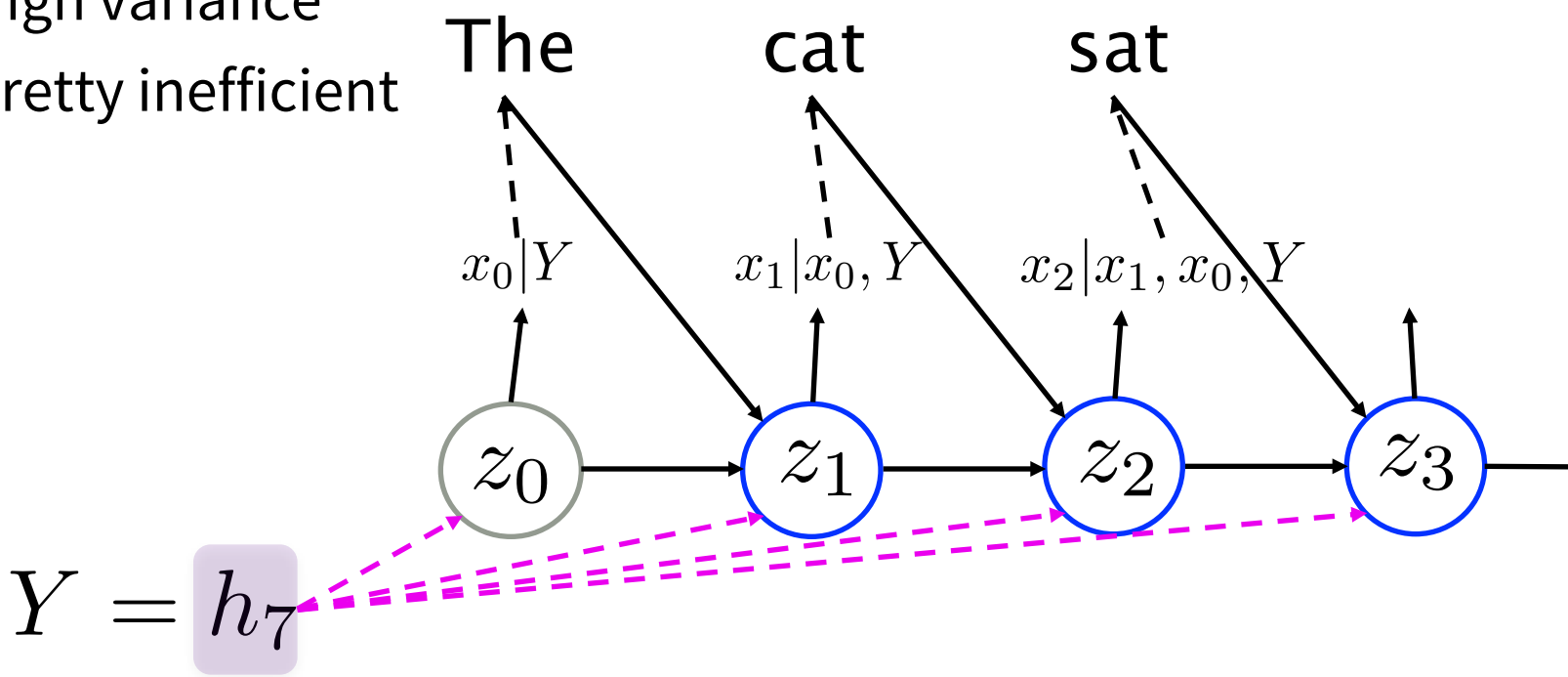
# Decoding (1) – Ancestral Sampling

- Efficient, unbiased sampling
- One symbol at a time from $\tilde{x}_t \sim x_t | x_{t-1}, \ldots, x_1, Y$
- Until $\tilde{x}_t = \langle \text{eos} \rangle$



# Decoding (1) – Ancestral Sampling

- Pros:
  1. Unbiased (asymptotically exact)
- Cons:
  1. High variance
  2. Pretty inefficient

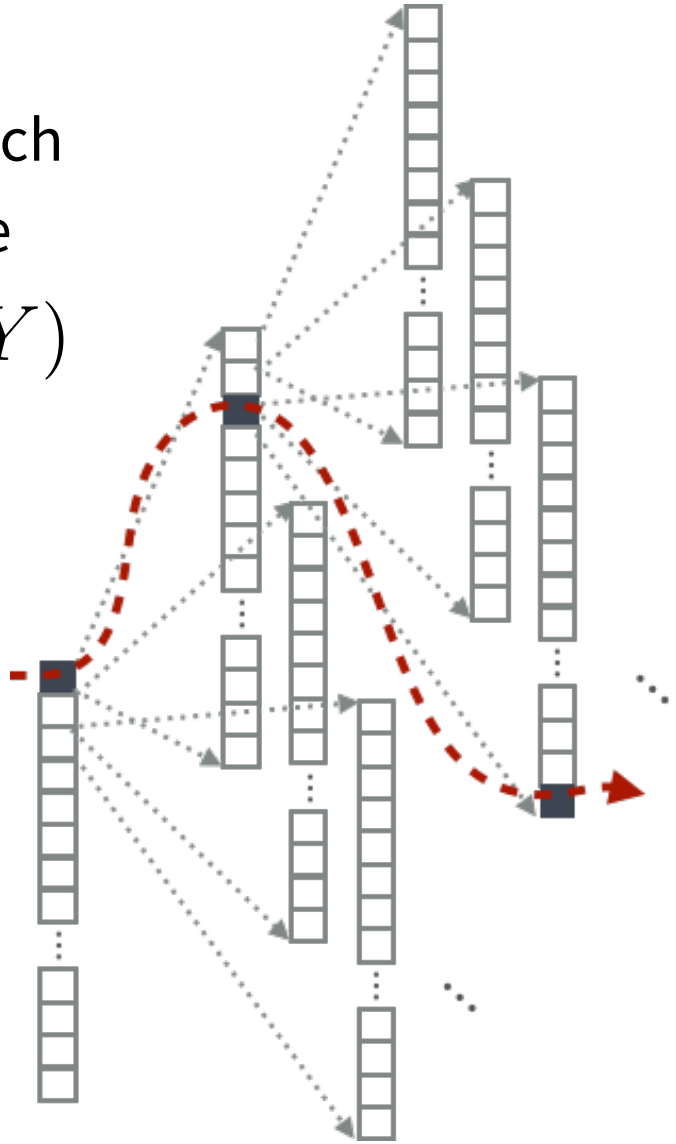


# Decoding (2) – Greedy Search

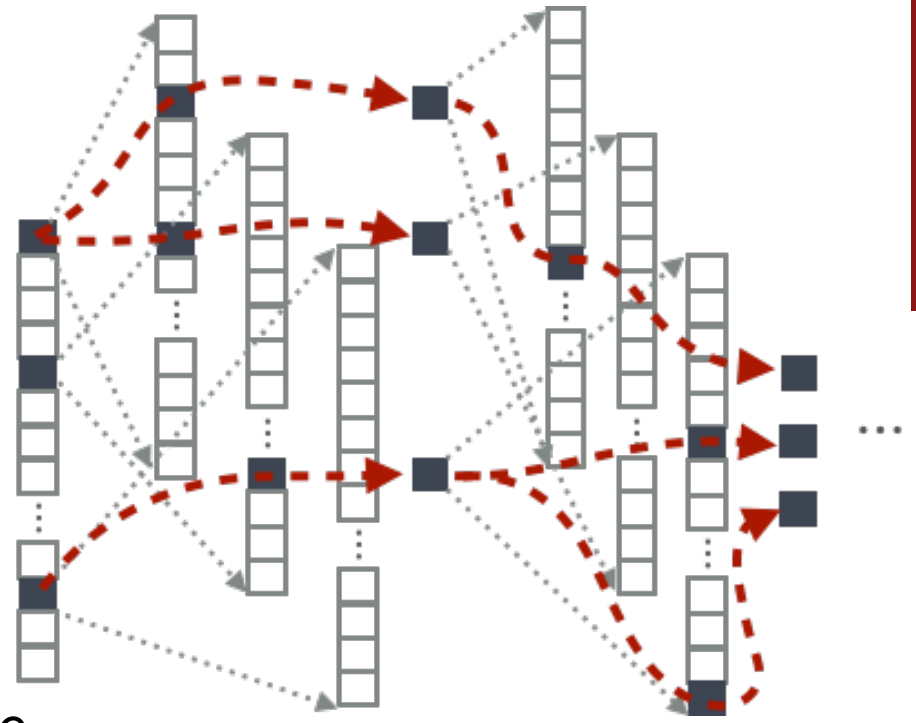
- Efficient, but heavily suboptimal search
- Pick the most likely symbol each time

$$\tilde{x}_t = \arg\max_x \log p(x|x_{<t}, Y)$$

- Until $\tilde{x}_t = \langle \text{eos} \rangle$
- Pros:
  1. Super-efficient
     - Both computation and memory
- Cons:
  1. Heavily suboptimal



# Decoding (3) – Beam Search



- Pretty, but *not quite* efficient
- Maintain *K* hypotheses at a time

$$\mathcal{H}_{t-1} = \{(\tilde{x}_1^1, \tilde{x}_2^1, \ldots, \tilde{x}_{t-1}^1), (\tilde{x}_1^2, \tilde{x}_2^2, \ldots, \tilde{x}_{t-1}^2), \ldots, (\tilde{x}_1^K, \tilde{x}_2^K, \ldots, \tilde{x}_{t-1}^K)\}$$

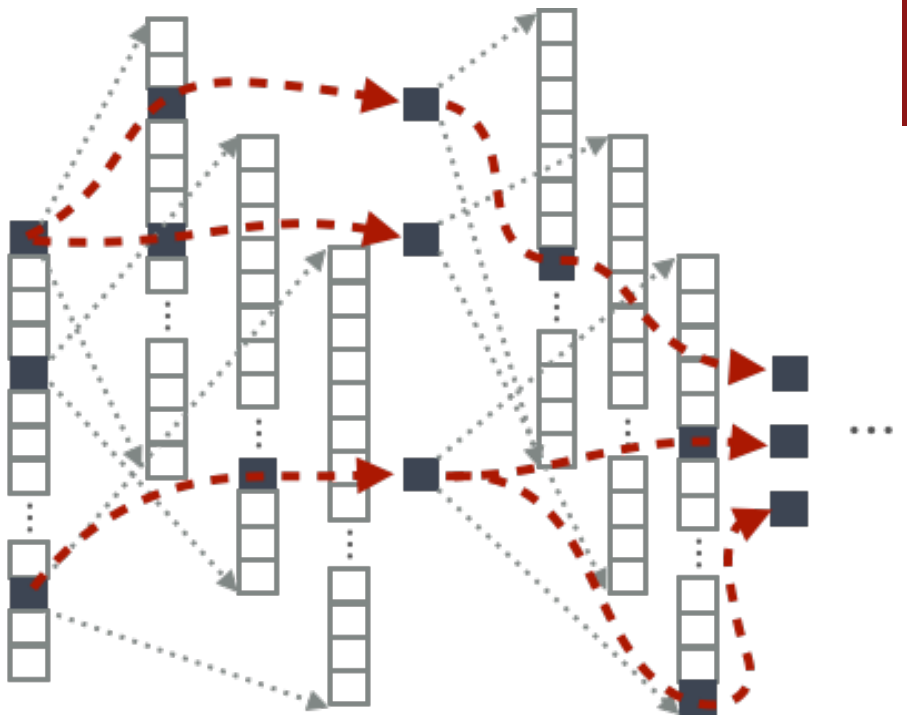
- Expand each hypothesis

$$\mathcal{H}_t^k = \{(\tilde{x}_1^k, \tilde{x}_2^k, \ldots, \tilde{x}_{t-1}^k, v_1), (\tilde{x}_1^k, \tilde{x}_2^k, \ldots, \tilde{x}_{t-1}^k, v_2), \ldots, (\tilde{x}_1^k, \tilde{x}_2^k, \ldots, \tilde{x}_{t-1}^k, v_{|V|})\}$$

- Pick top-K hypotheses from the union $\mathcal{H}_t = \cup_{k=1}^K \mathcal{B}_k$, where

$$\mathcal{B}_k = \arg\max_{\tilde{X} \in \mathcal{A}_k} \log p(\tilde{X}|Y), \ \mathcal{A}_k = \mathcal{A}_{k-1} - \mathcal{B}_{k-1}, \text{ and } \mathcal{A}_1 = \cup_{k'=1}^K \mathcal{H}_t^{k'}.$$

# Decoding (3) – Beam Search



- Asymptotically exact, as $K \rightarrow \infty$
- But, not necessarily monotonic improvement w.r.t. $K$
- *K* should be selected to maximize the translation quality on a validation set.

# Decoding

- En-Cz: 12m training sentence pairs

| Strategy | # Chains | Valid Set | | Test Set | |
|---|---|---|---|---|---|
| | | NLL | BLEU | NLL | BLEU |
| Ancestral Sampling | 50 | 22.98 | 15.64 | 26.25 | 16.76 |
| Greedy Decoding | - | 27.88 | 15.50 | 26.49 | 16.66 |
| Beamsearch | 5 | 20.18 | 17.03 | 22.81 | 18.56 |
| Beamsearch | 10 | 19.92 | 17.13 | 22.44 | 18.59 |



[Cho, arXiv 2016]

# Decoding

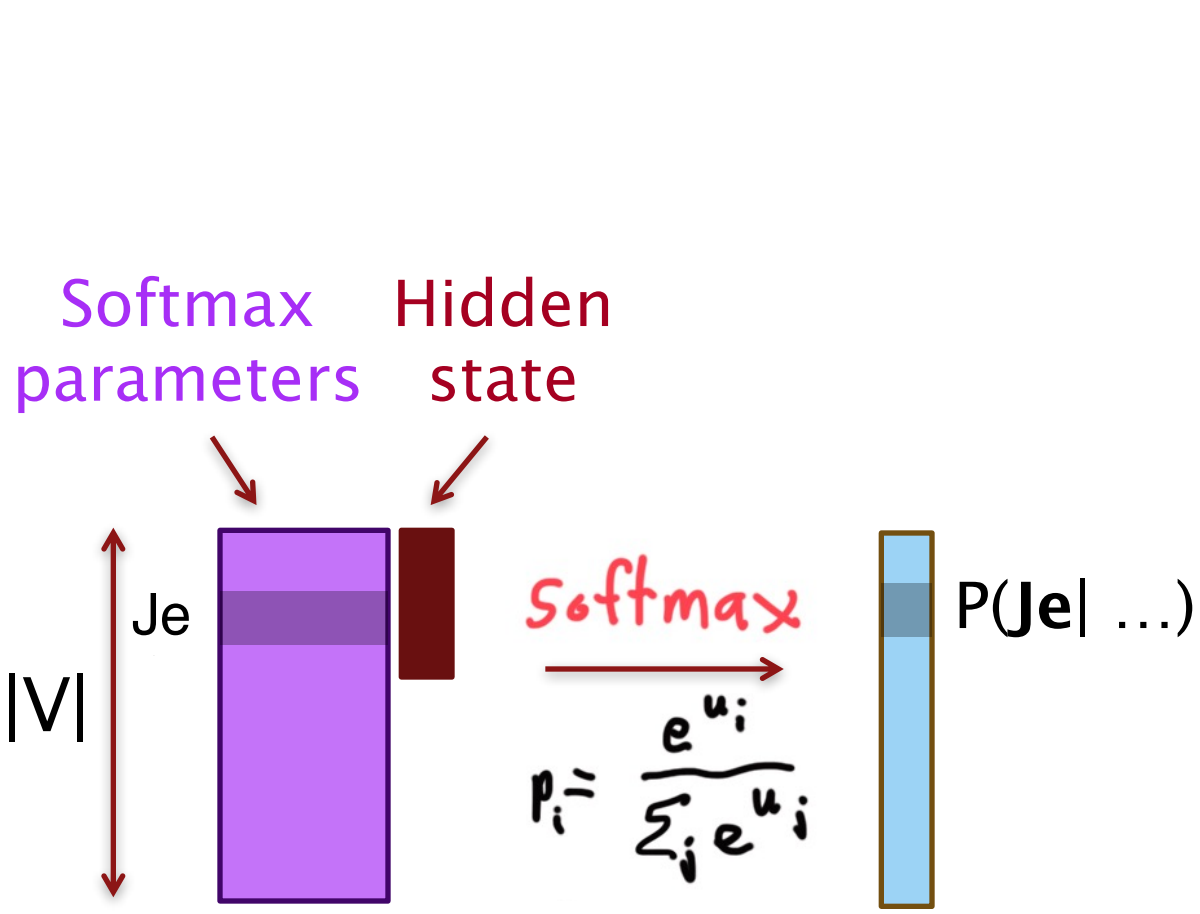
- Greedy Search
  - Computationally efficient
  - Not great quality
- Beam Search
  - Computationally expensive
  - Not easy to parallelize
  - Much better quality

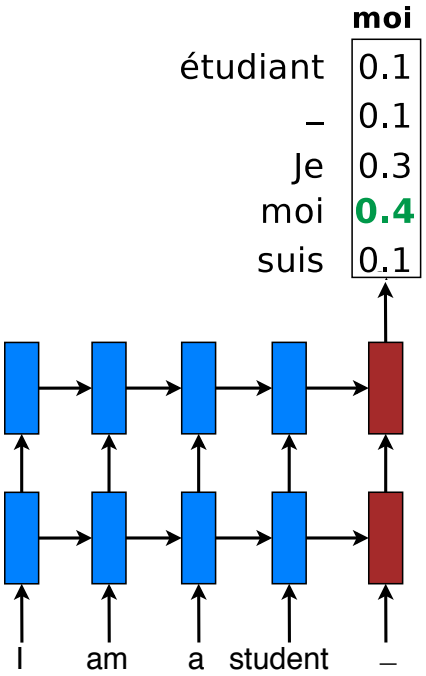
***Is there anything in-between?***



[Cho, arXiv 2016]

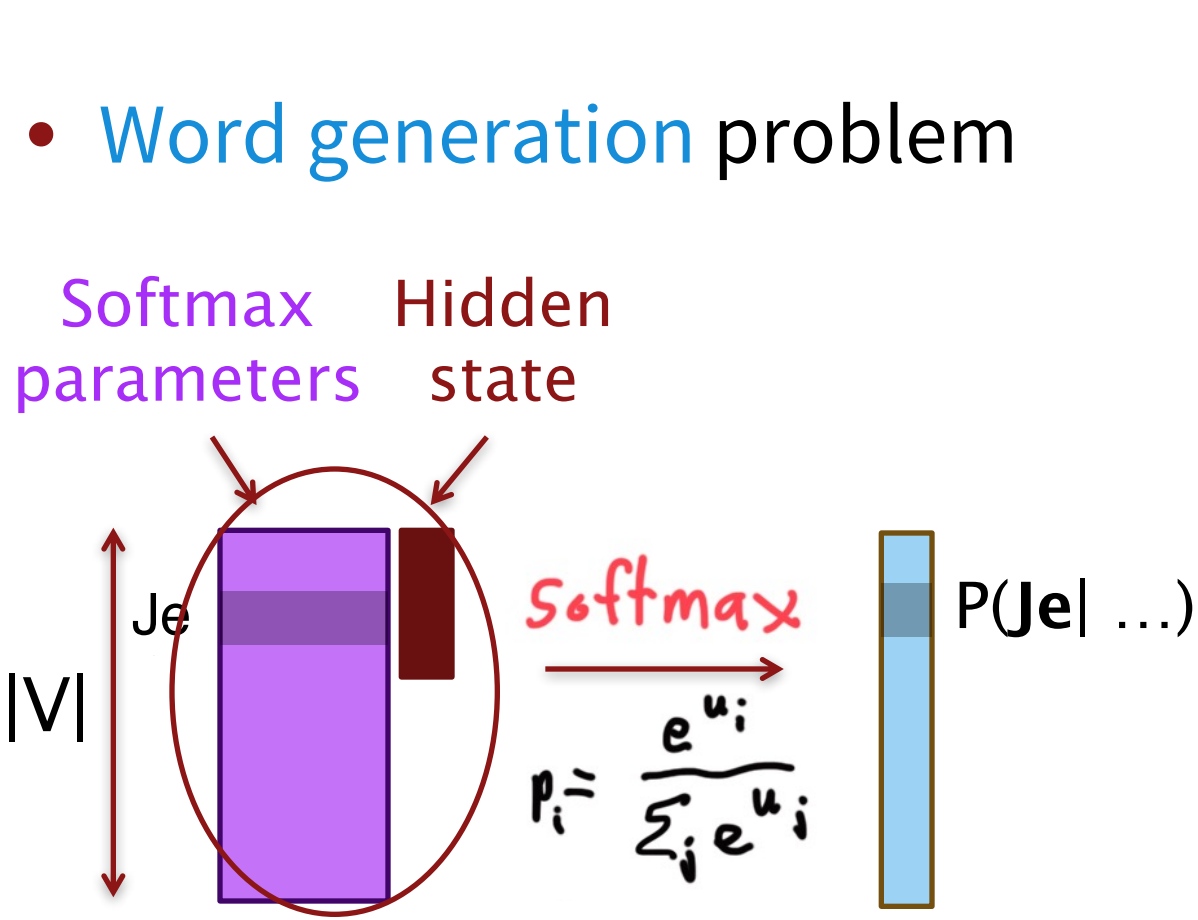
# The word generation problem

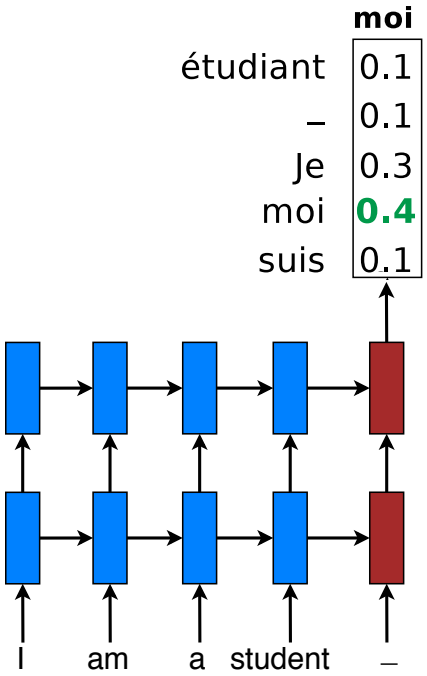


# The word generation problem

- Word generation problem

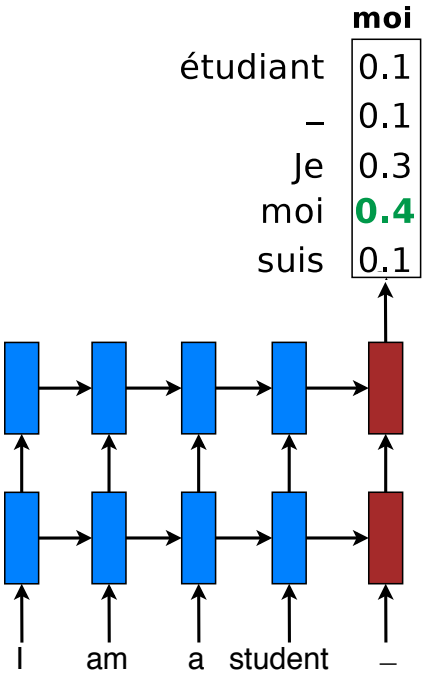



Softmax computation is expensive.

# The word generation problem

- Word generation problem
  - Vocabs are modest: 50K.




The ecotax portico in Pont-de-Buis
Le portique écotaxe de Pont-de-Buis

The <unk> portico in <unk>
Le <unk> <unk> de <unk>

# *First thought*: scale the softmax

- Lots of ideas from the neural LM literature!
- *Hierarchical models*: tree-structured vocabulary
  - [Morin & Bengio, AISTATS'05], [Mnih & Hinton, NIPS'09].
  - Complex, sensitive to tree structures.
- *Noise-contrastive estimation*: binary classification
  - [Mnih & Teh, ICML'12], [Vaswani et al., EMNLP'13].
  - Different noise samples per training example.*

Not GPU-friendly



*We'll mention a simple fix for this!

# Copy Mechanism

- Simple way to track *target* <unk>.

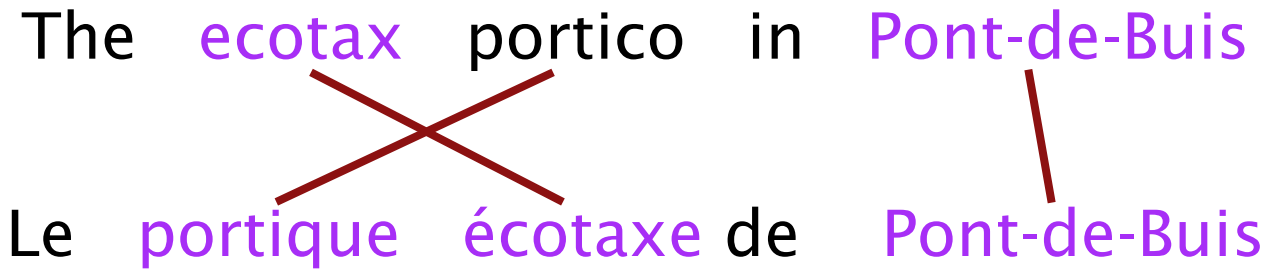
- Treat any NMT as a black box.
  - Annotate training data.
  - Post-process translations.

Complementary to softmax scaling!

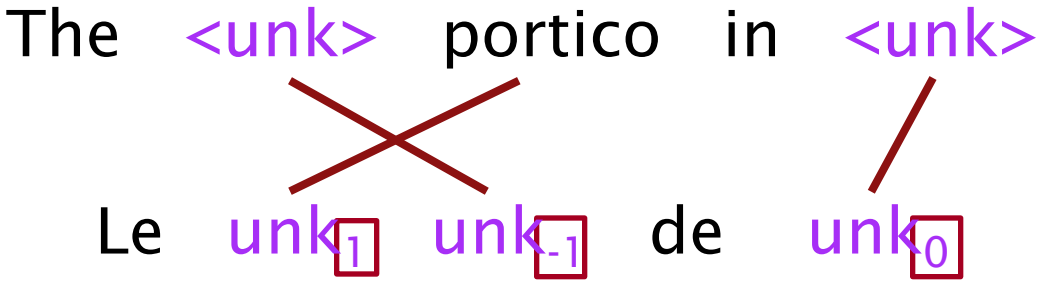
*Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, Wojciech Zaremba.* ***Addressing the Rare Word Problem in Neural Machine Translation****. ACL'15.*

# Training annotation

- Learn alignments

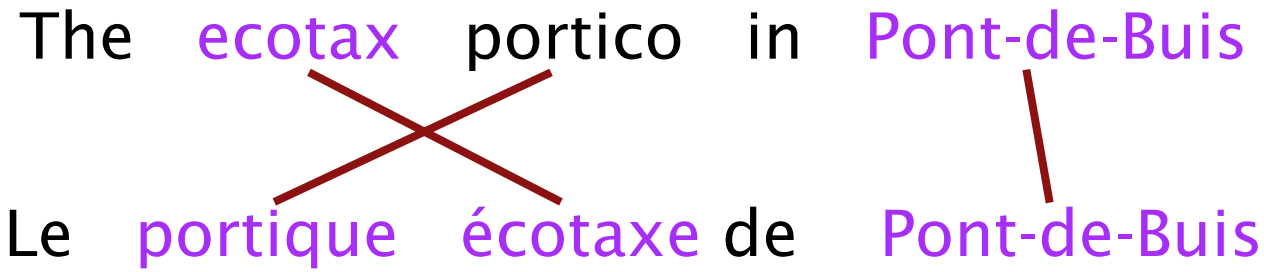



- Add relative positions

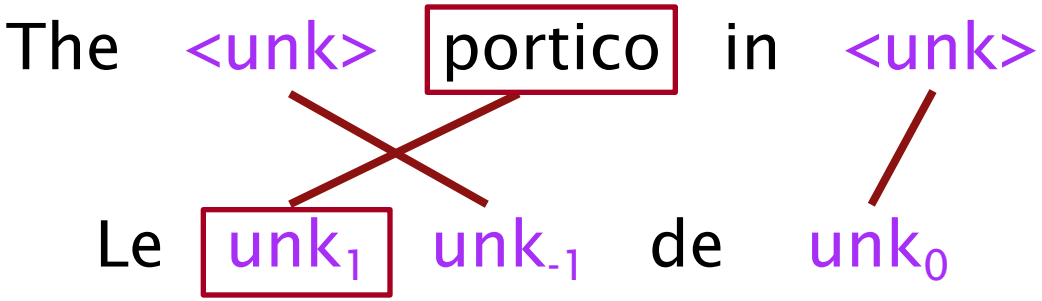


# Training annotation

- Learn alignments

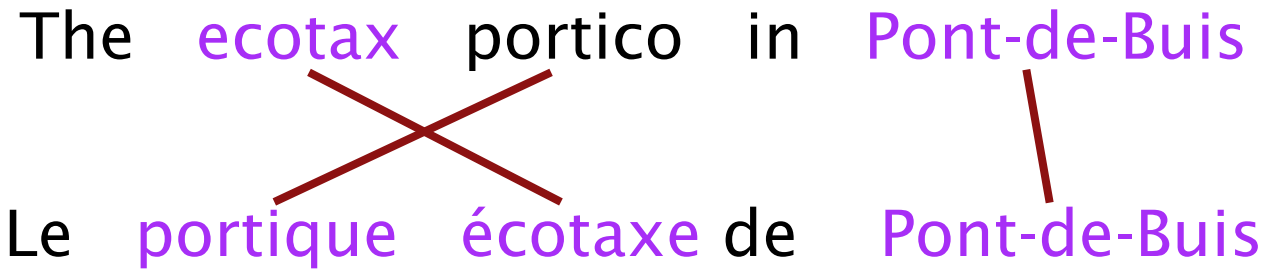



- Add relative positions

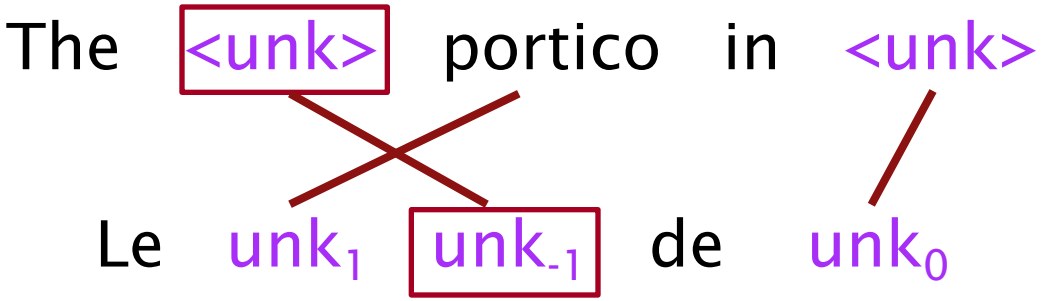


# Training annotation

- Learn alignments




- Add relative positions

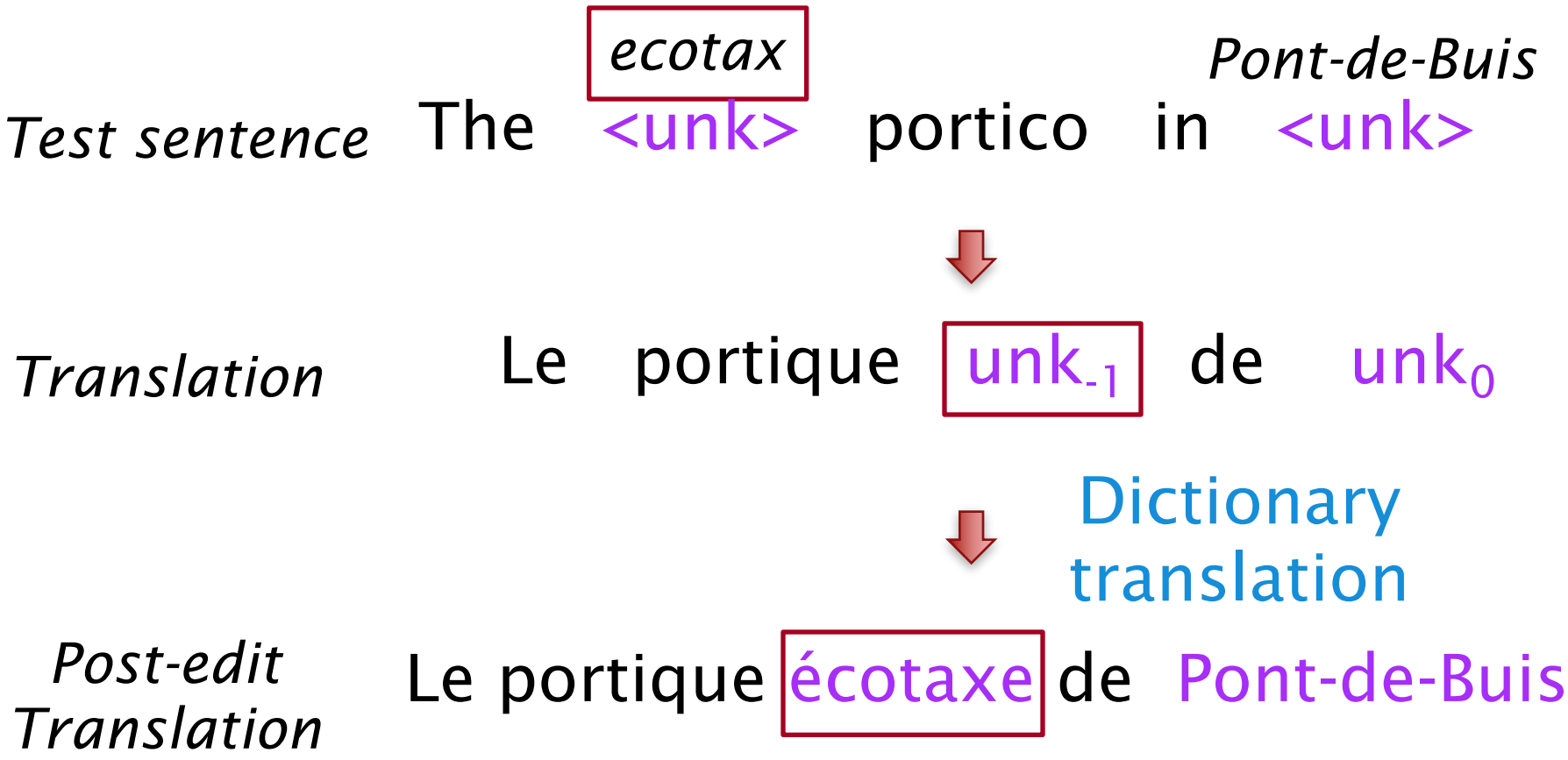


# Post-processing

*Test sentence* The *ecotax* <unk> portico in *Pont-de-Buis* <unk>

*Translation* Le portique $unk_{-1}$ de $unk_{0}$

# Post-processing



# Post-processing

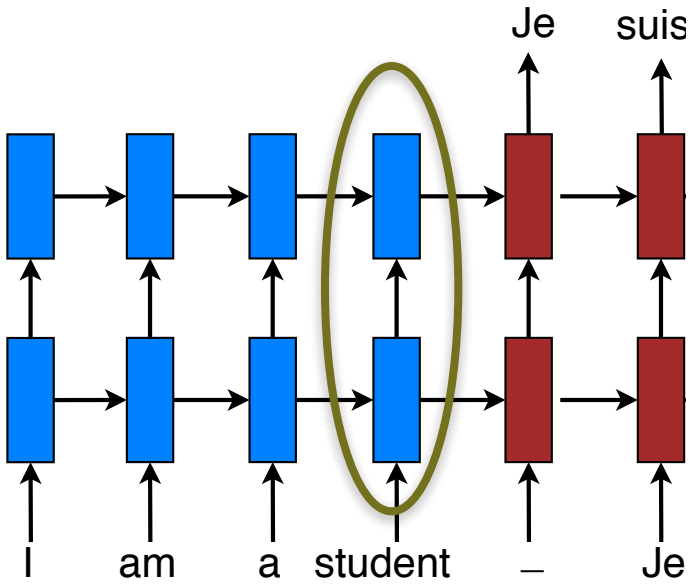
*Test sentence*: The *ecotax* <unk> portico in *Pont-de-Buis* <unk>

*Translation*: Le portique $unk_{-1}$ de $unk_0$

Identity copy

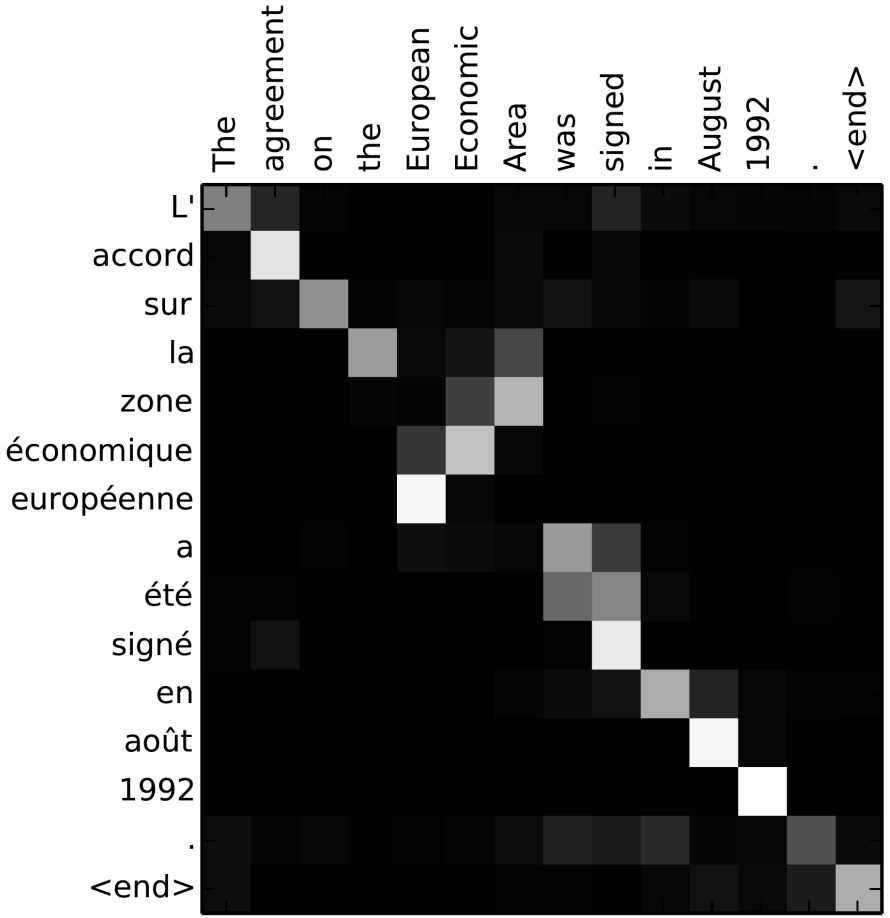
*Post-edit Translation*: Le portique écotaxe de Pont-de-Buis

# Vanilla seq2seq & long sentences



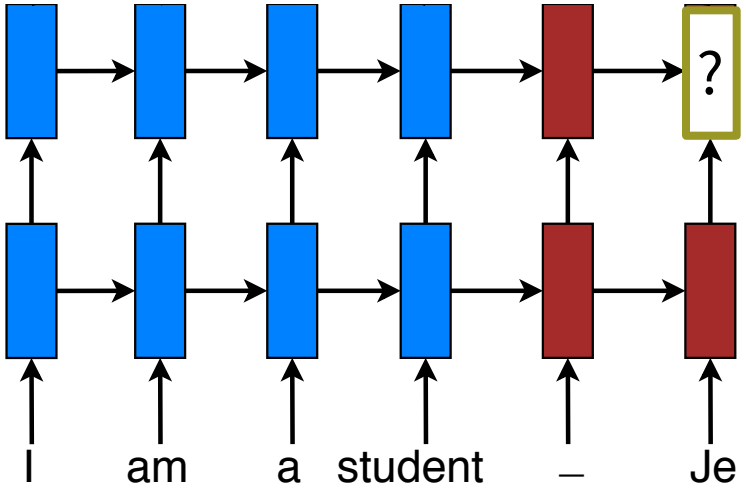

Problem: fixed-dimensional representations

# Learning both translation & alignment




*Dzmitry Bahdanau, KyungHuyn Cho, and Yoshua Bengio.* ***Neural Machine Translation by Jointly Learning to Translate and Align****. ICLR'15.*

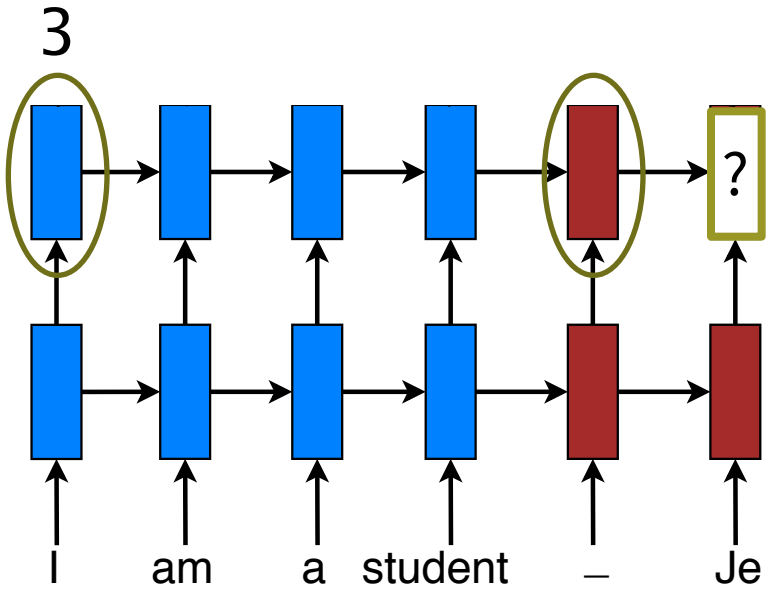
# Attention Mechanism




Simplified version of (Bahdanau et al., 2015)

# Attention Mechanism – *Scoring*

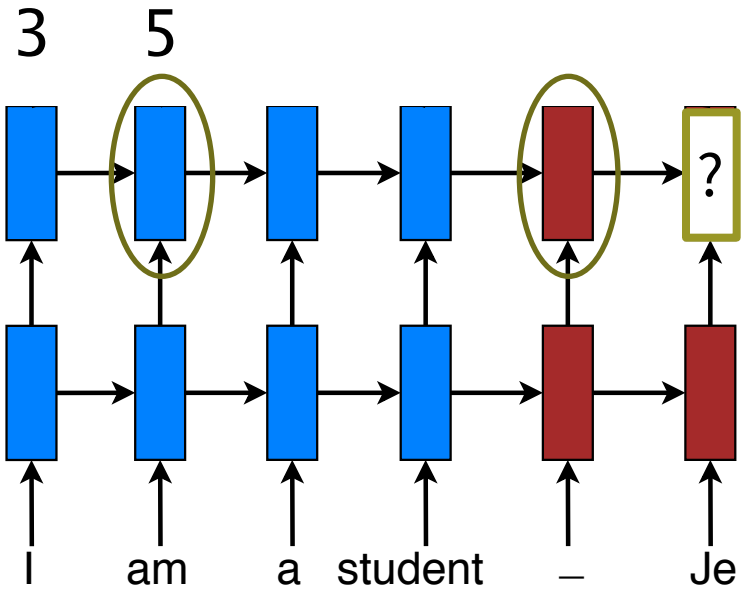
$$\text{score}(\boldsymbol{h}_{t-1}, \bar{\boldsymbol{h}}_s)$$




- Compare target and source hidden states.

# Attention Mechanism – *Scoring*

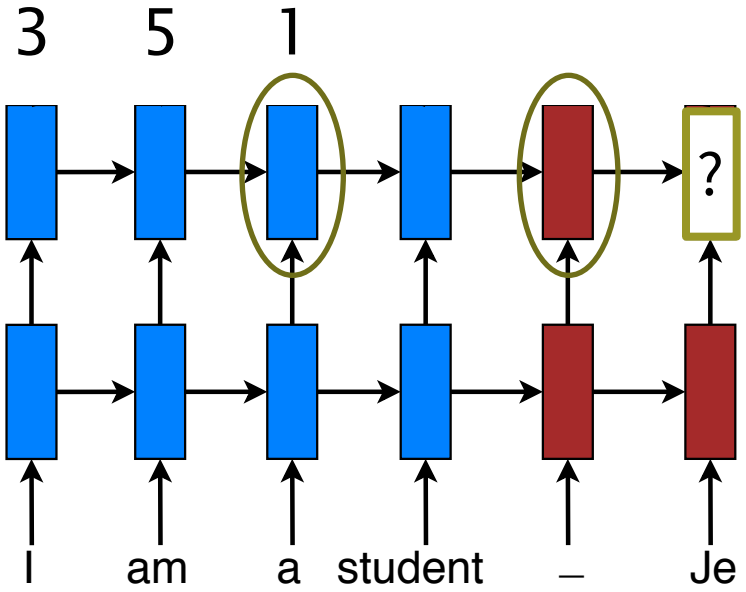
$$\text{score}(\boldsymbol{h}_{t-1}, \bar{\boldsymbol{h}}_s)$$




- Compare target and source hidden states.

# Attention Mechanism – *Scoring*

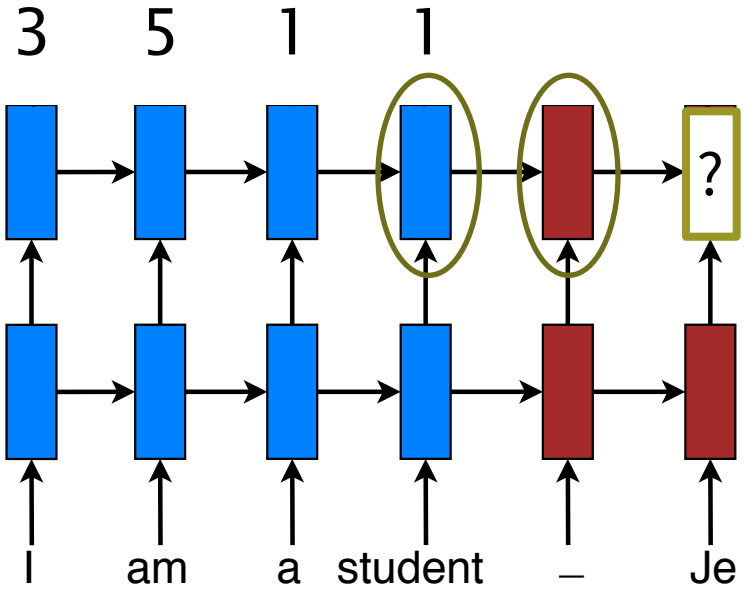
$$\text{score}(\boldsymbol{h}_{t-1}, \bar{\boldsymbol{h}}_s)$$




- Compare target and source hidden states.

# Attention Mechanism – *Scoring*

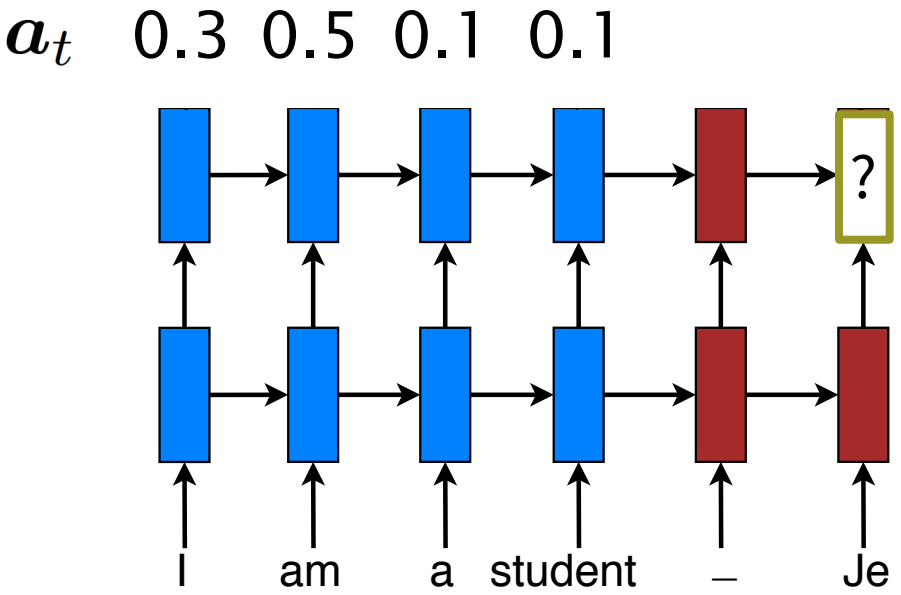
$$\text{score}(\boldsymbol{h}_{t-1}, \bar{\boldsymbol{h}}_s)$$




- Compare target and source hidden states.

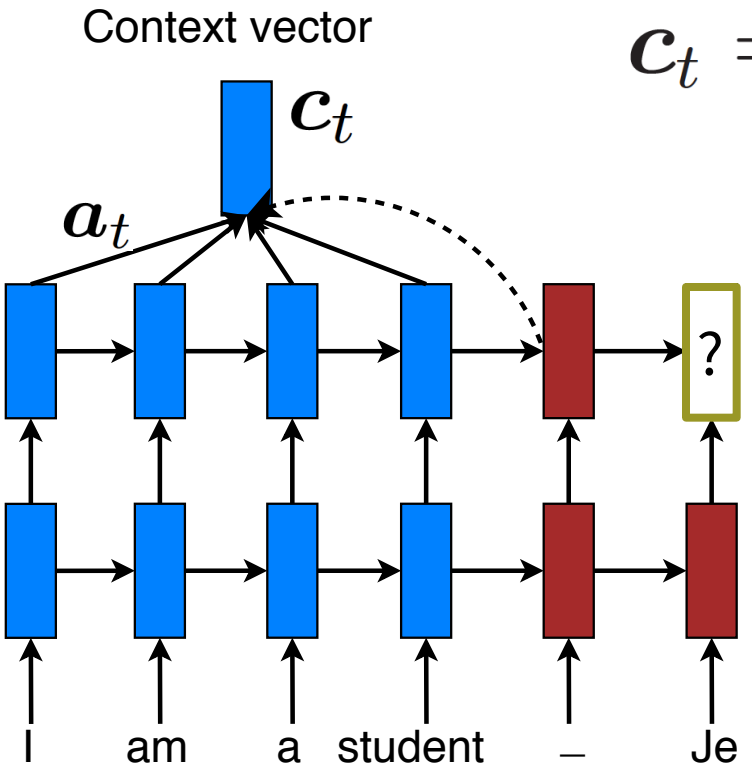
# Attention Mechanism – *Normalization*

$$a_t(s) = \frac{e^{\text{score}(s)}}{\sum_{s'} e^{\text{score}(s')}}$$




- Convert into alignment weights.

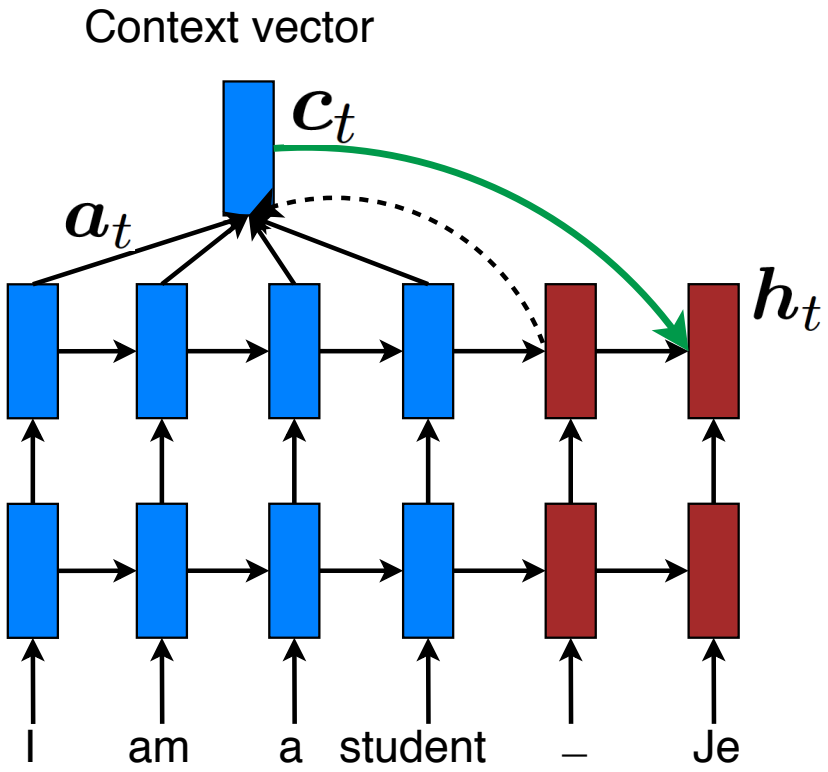
# Attention Mechanism – *Context*




$$c_t = \sum_s a_t(s)\bar{h}_s$$

- Build context vector: weighted average.

# Attention Mechanism – *Hidden State*




- Compute the next hidden state.

# Sample English-German translations

| | |
|---|---|
| source | Orlando Bloom and ***Miranda Kerr*** still love each other |
| human | Orlando Bloom und **Miranda Kerr** lieben sich noch immer |
| **+attn** | Orlando Bloom und **Miranda Kerr** lieben einander noch immer . |
| base | Orlando Bloom und **Lucas Miranda** lieben einander noch immer . |

- Translates names correctly.

# Sample English-German translations

| | |
|---|---|
| source | We ' re pleased the FAA recognizes that an enjoyable passenger experience is ***not incompatible*** with safety and security , said Roger Dow , CEO of the U.S. Travel Association . |
| human | Wir freuen uns , dass die FAA erkennt , dass ein angenehmes Passagiererlebnis nicht **im Wider- spruch zur Sicherheit steht** , sagte Roger Dow , CEO der U.S. Travel Association . |
| **+attn** | Wir freuen uns , dass die FAA anerkennt , dass ein angenehmes ist nicht mit Sicherheit und Sicherheit **unvereinbar** ist , sagte Roger Dow , CEO der US - die . |
| base | Wir freuen uns u ̈ber die <unk> , dass ein <unk> <unk> mit Sicherheit nicht **vereinbar** ist mit Sicherheit und Sicherheit , sagte Roger Cameron , CEO der US - <unk> . |

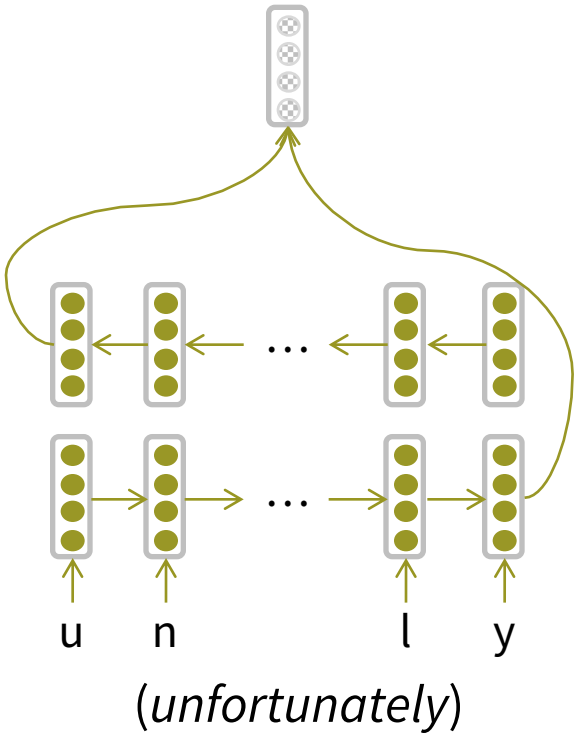
- Translates a doubly-negated phrase correctly.

# Sample English-German translations

| | |
|---|---|
| source | We ′ re pleased the FAA recognizes that an enjoyable passenger experience is ***not incompatible*** with safety and security , said Roger Dow , CEO of the U.S. Travel Association . |
| human | Wir freuen uns , dass die FAA erkennt , dass ein angenehmes Passagiererlebnis nicht **im Wider- spruch zur Sicherheit steht** , sagte Roger Dow , CEO der U.S. Travel Association . |
| **+attn** | Wir freuen uns , dass die FAA anerkennt , dass ein angenehmes ist nicht mit Sicherheit und Sicherheit **unvereinbar** ist , sagte Roger Dow , CEO der US - die . |
| base | Wir freuen uns u ̈ber die <unk> , dass ein <unk> <unk> mit Sicherheit nicht **vereinbar** ist mit Sicherheit und Sicherheit , sagte Roger Cameron , CEO der US - <unk> . |

- Translates a doubly-negated phrase correctly.

# Character-based LSTM

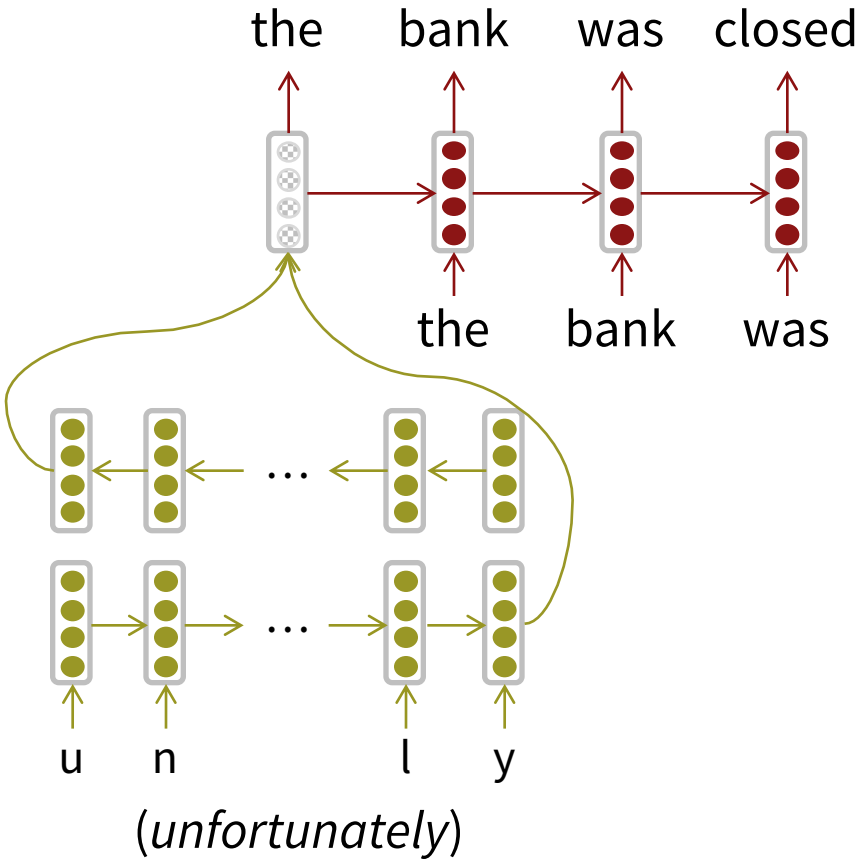



(*unfortunately*)

Bi-LSTM builds word representations

*Ling, Luís, Marujo, Astudillo, Amir, Dyer, Black, Trancoso.* ***Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation****. EMNLP'15.*

# Character-based LSTM

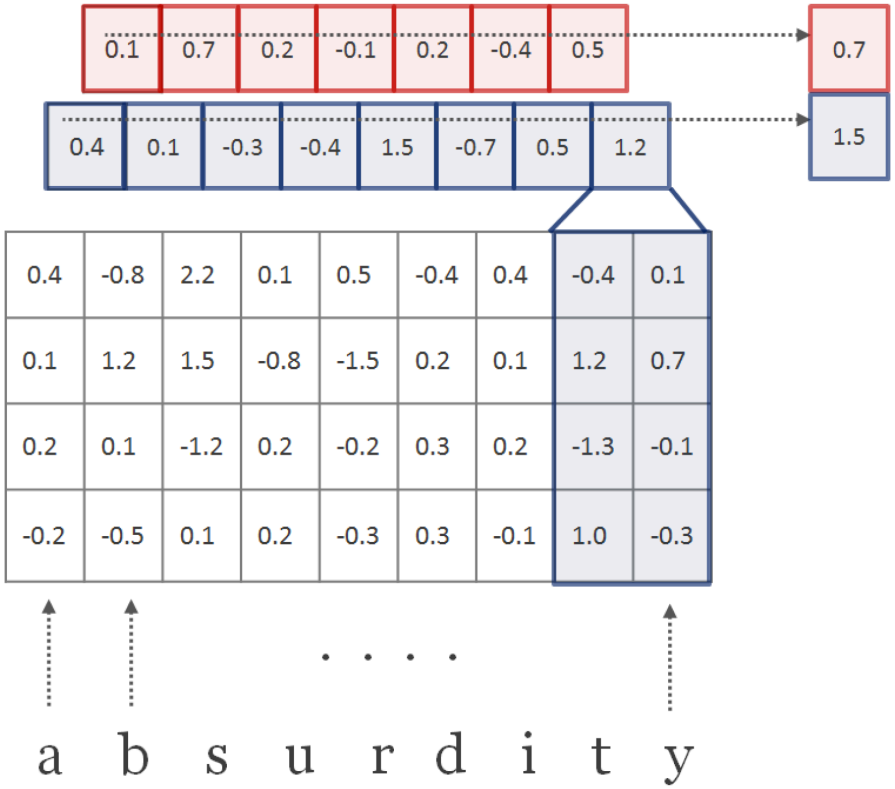



Recurrent Language Model

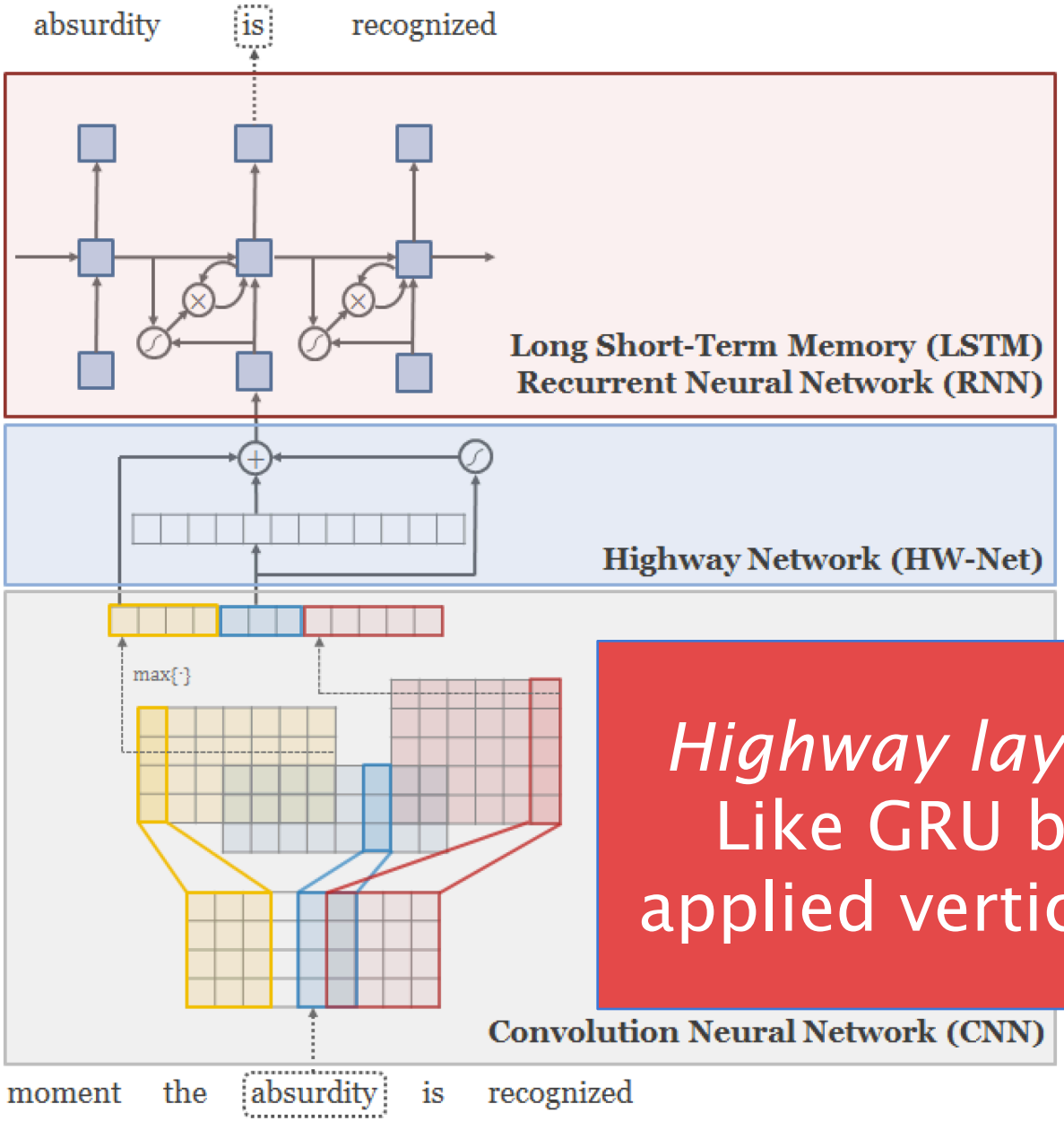
Bi-LSTM builds word representations

*Ling, Luís, Marujo, Astudillo, Amir, Dyer, Black, Trancoso.* ***Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation****. EMNLP'15.*

# Character ConvNet

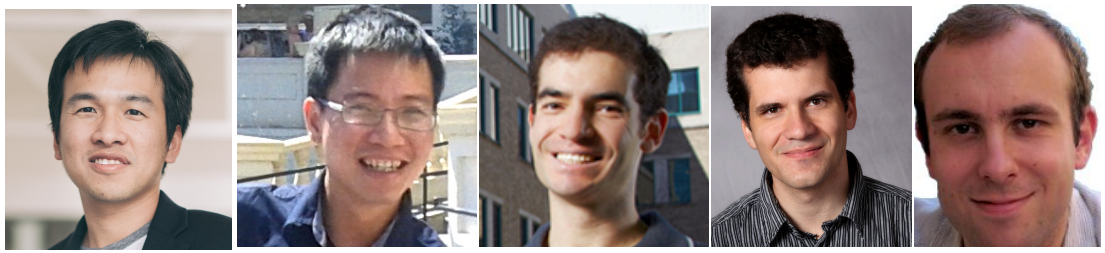



*Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush.* ***Character-Aware Neural Language Models****. AAAI 2016.*

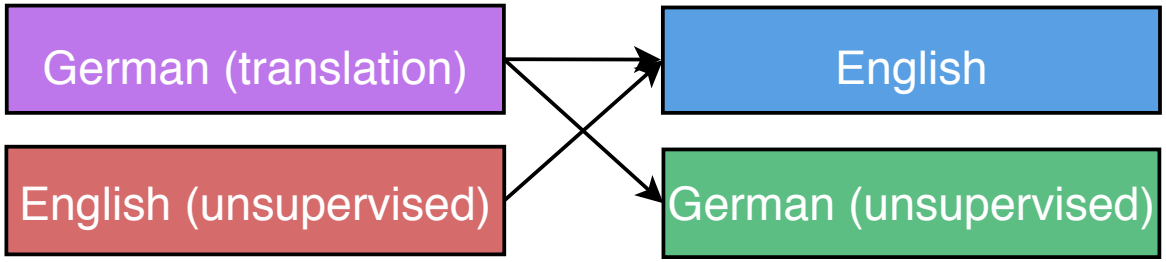



*Highway layer* ≅ Like GRU but applied vertically.

# Autoencoders



- Shared encoders & decoders: 3 tasks




- Small amount of mono data as regularization.
  - +0.9 BLEU improvements

How to utilize more monolingual data?

*Thang Luong, Quoc Le, Ilya Sutskever, Oriol Vinyals, Lukasz Kaiser.* ***Multi-task sequence to sequence learning****. ICLR 2016.*

# Enriching parallel data

- *Dummy* source sentences

| | | |
|---|---|---|
| She loves cute cats | Elle aime les chats mignons | (parallel) |
| <null> | Elle aime les chiens mignons | (mono) |

Small gain +0.4-1.0 BLEU.
Difficult to add more mono data.

*Rico Sennrich, Barry Haddow, and Alexandra Birch.* ***Improving Neural Machine Translation Models with Monolingual Data****. ACL 2016.*

# Enriching parallel data

- *Synthetic* source sentences

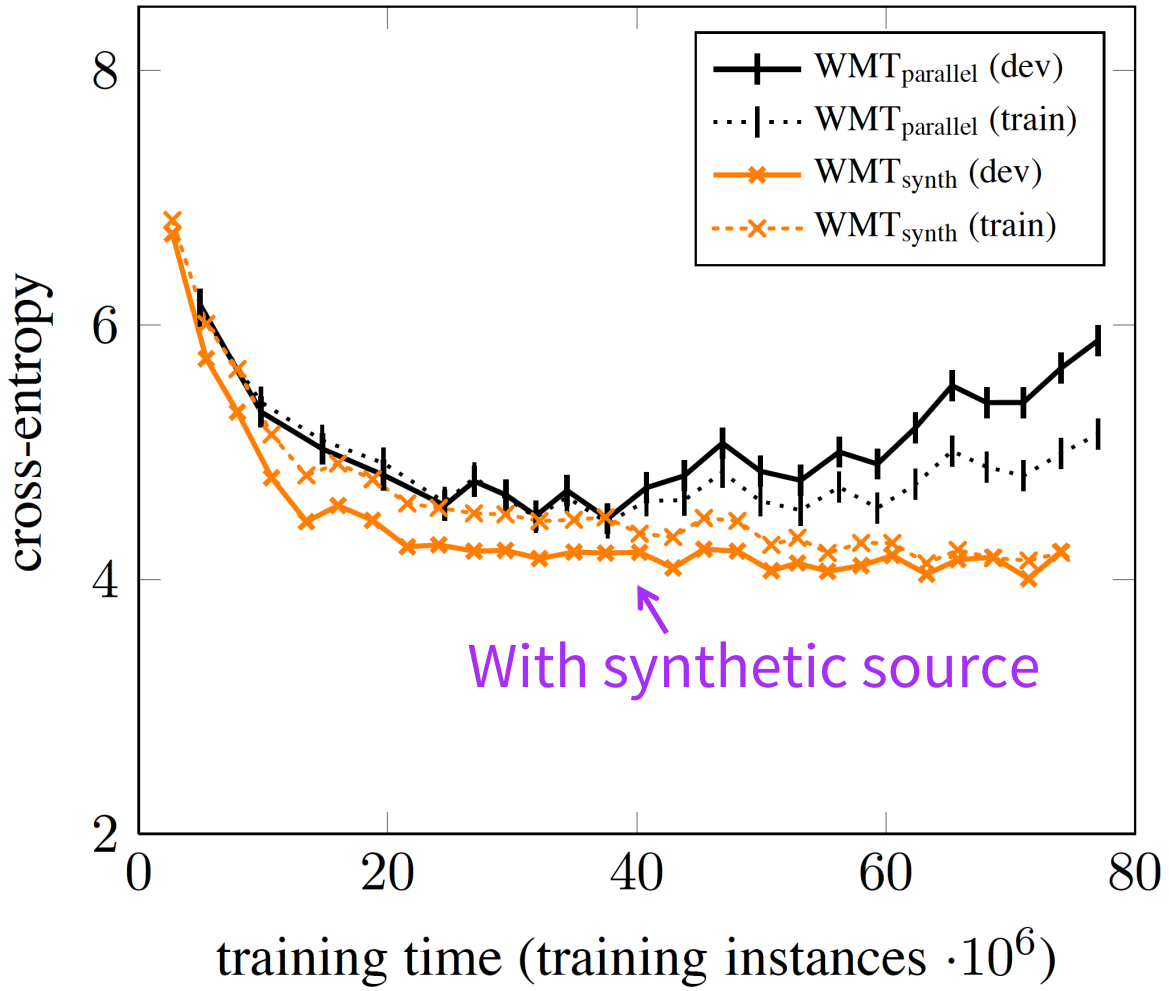
| | | |
|---|---|---|
| She loves cute cats | Elle aime les chats mignons | (parallel) |
| She likes cute cats | Elle aime les chiens mignons | (mono) |

↖ Back translated

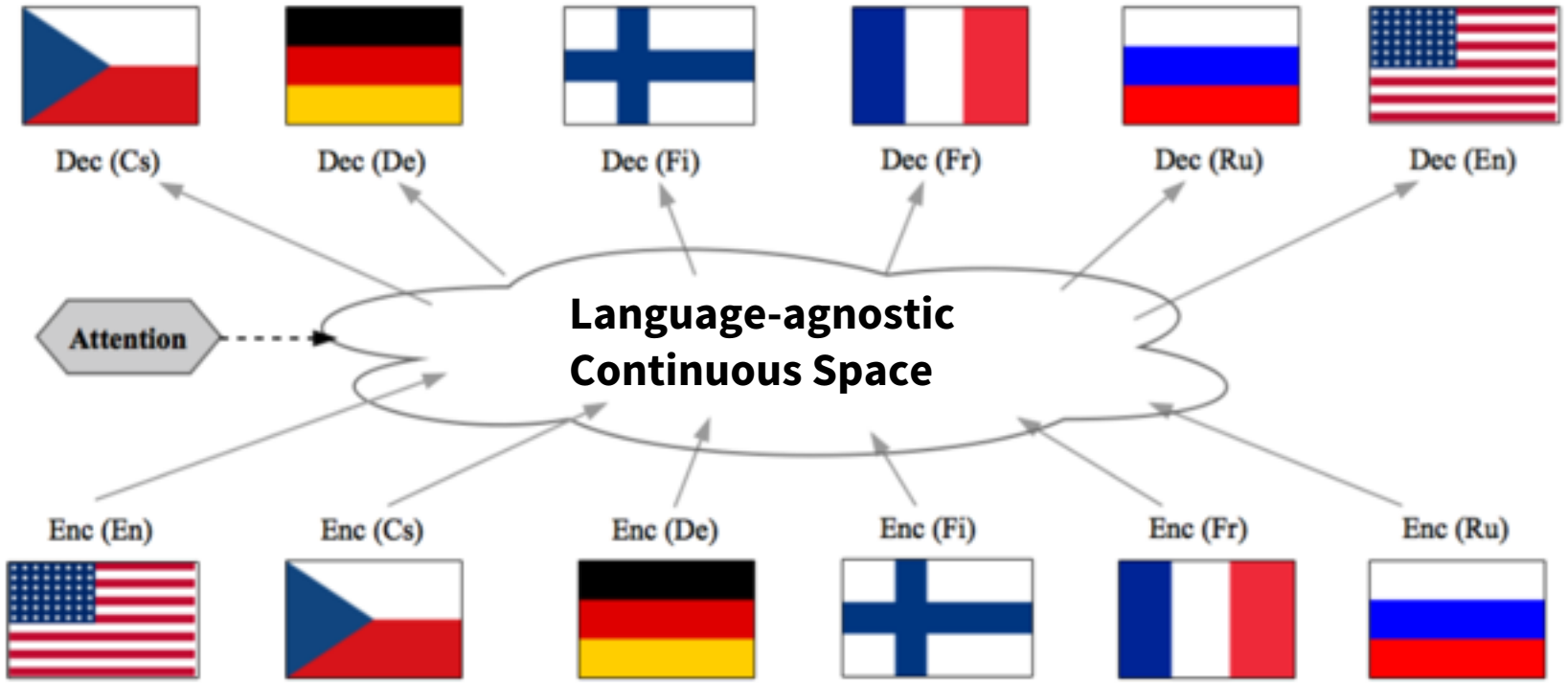
Large gain +2.1-3.4 BLEU.

*Rico Sennrich, Barry Haddow, and Alexandra Birch.* ***Improving Neural Machine Translation Models with Monolingual Data****. ACL 2016.*

# Prevent Over-fitting



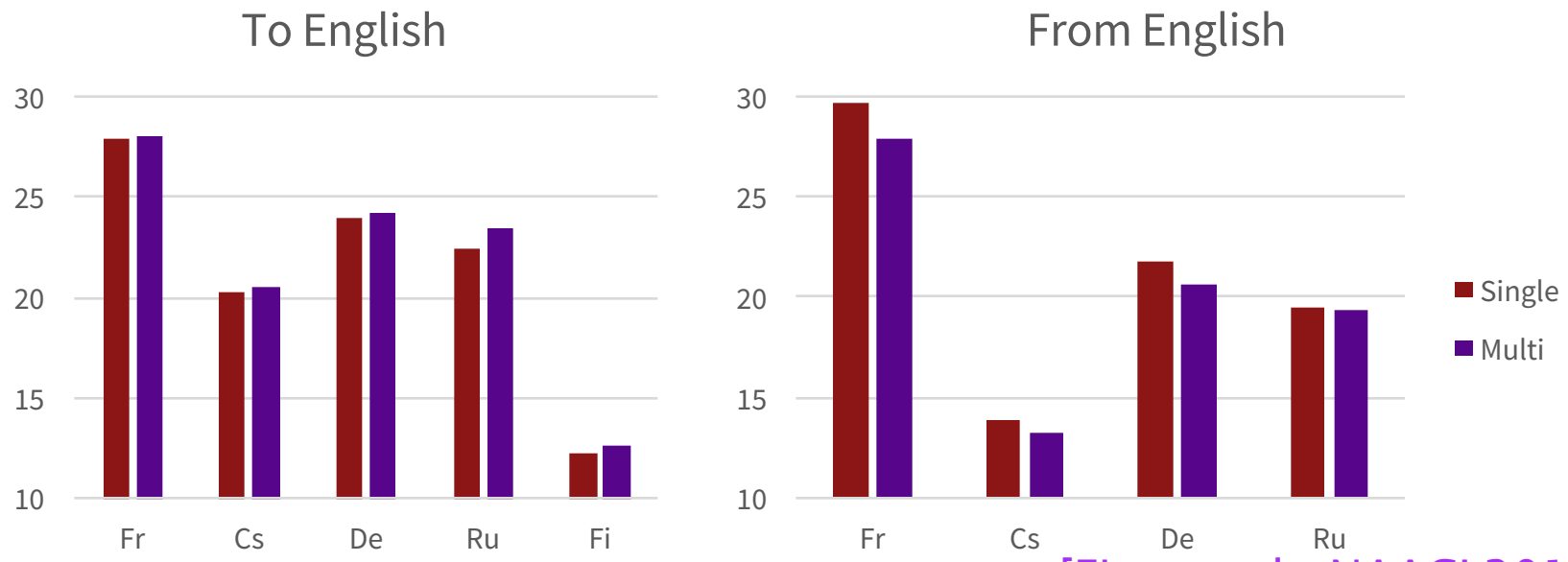
# Multilingual Translation




 [Dong et al., ACL2015; Luong et al., ICLR2016; Firat et al., NAACL2016]

# Multilingual Translation: First Result

- 10 language pair-directions
  - En → {Fr, Cs, De, Ru, Fi} + {Fr, Cs, De, Ru, Fi} → En
- 60+ million bilingual sentence pairs
- *Comparable to 10 single-pair models*




[Firat et al., NAACL2016]

# Multilingual Translation: Looking Ahead

- Low-resource translation
  - Positive language transfer from high-resource to low-resource language pair-directions

| | # Symbols | | # Sentence | | |
|---|---|---|---|---|---|
| | # En | Other | Train | Dev | Test |
| En-Uz | 1.361m | 1.186m | 73.66k | 948 | 882 |
| En-Es | 908.1m | 924.9m | 34.71m | 3003 | 3000 |
| En-Fr | 1.837b | 1.911b | 65.77m | 3003 | 3000 |



[Firat et al., under review]

# Multilingual Translation: Looking Ahead

- Low-resource translation: Example

Uz-En: 6.45

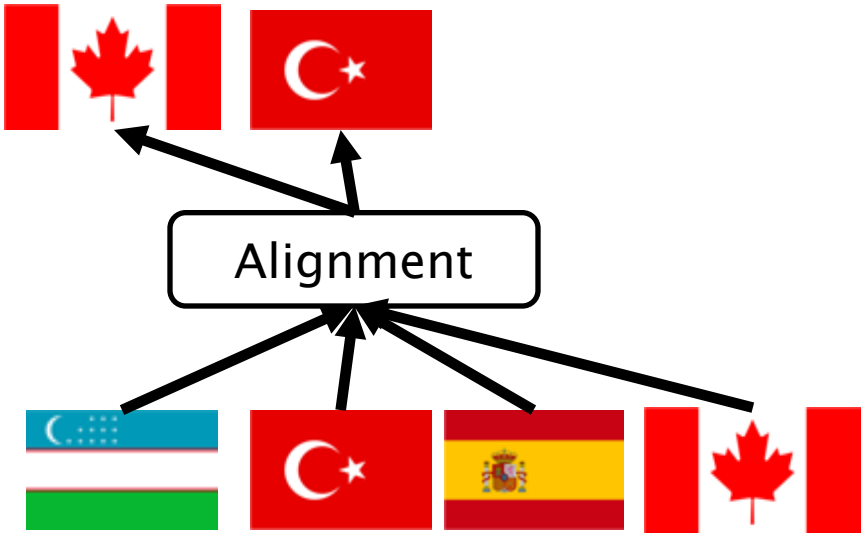
Uz-En + Tr-En: 9.34

Uz-En + Tr-En + Es-En: 10.34

Uz-En + Tr-En + Es-En + En-Tr: 9.41

Ensemble: 12.99

- 3x Uz-En + Tr-En + Es-En
- 3x Uz-En + Tr-En + Es-En + En-Tr






[Firat et al., 2016c]

# References (1)


- [Bahdanau et al., ICLR'15] Neural Translation by Jointly Learning to Align and Translate. http://arxiv.org/pdf/1409.0473.pdf
- [Chung, Cho, Bengio, ACL'16]. A Character-Level Decoder without Explicit Segmentation for Neural Machine Translation. http://arxiv.org/pdf/1603.06147.pdf
- [Cohn, Hoang, Vymolova, Yao, Dyer, Haffari, NAACL'16] Incorporating Structural Alignment Biases into an Attentional Neural Translation Model. https://arxiv.org/pdf/1601.01085.pdf
- [Dong, Wu, He, Yu, Wang, ACL'15]. Multi-task learning for multiple language translation. http://www.aclweb.org/anthology/P15-1166
- [Firat, Cho, Bengio, NAACL'16]. Multi-Way, Multilingual Neural Machine Translation with a Shared Attention Mechanism. https://arxiv.org/pdf/1601.01073.pdf
- [Gu, Lu, Li, Li, ACL'16] Incorporating Copying Mechanism in Sequence-to-Sequence Learning. https://arxiv.org/pdf/1603.06393.pdf
- [Gulcehre, Ahn, Nallapati, Zhou, Bengio, ACL'16] Pointing the Unknown Words. http://arxiv.org/pdf/1603.08148.pdf
- [Hochreiter & Schmidhuber, 1997] Long Short-term Memory. http://deeplearning.cs.cmu.edu/pdfs/Hochreiter97_lstm.pdf
- [Kim, Jernite, Sontag, Rush, AAAI'16]. Character-Aware Neural Language Models. https://arxiv.org/pdf/1508.06615.pdf

# References (2)


- [Ji, Haffari, Eisenstein, NAACL'16] A Latent Variable Recurrent Neural Network for Discourse-Driven Language Models. https://arxiv.org/pdf/1603.01913.pdf
- [Ji, Vishwanathan, Satish, Anderson, Dubey, ICLR'16] BlackOut: Speeding up Recurrent Neural Network Language Models with very Large Vocabularies. http://arxiv.org/pdf/1511.06909.pdf
- [Jia, Liang, ACL'16]. Data Recombination for Neural Semantic Parsing. https://arxiv.org/pdf/1606.03622.pdf
- [Ling, Luís, Marujo, Astudillo, Amir, Dyer, Black, Trancoso, EMNLP'15]. Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. http://arxiv.org/pdf/1508.02096.pdf
- [Luong et al., ACL'15a] Addressing the Rare Word Problem in Neural Machine Translation. http://www.aclweb.org/anthology/P15-1002
- [Luong et al., ACL'15b] Effective Approaches to Attention-based Neural Machine Translation. https://aclweb.org/anthology/D/D15/D15-1166.pdf
- [Luong & Manning, IWSLT'15] Stanford Neural Machine Translation Systems for Spoken Language Domain. http://nlp.stanford.edu/pubs/luong-manning-iwslt15.pdf
- [Mnih & Hinton, NIPS'09] A Scalable Hierarchical Distributed Language Model. https://www.cs.toronto.edu/~amnih/papers/hlbl_final.pdf
- [Mnih & Teh, ICML'12] A fast and simple algorithm for training neural probabilistic language models. https://www.cs.toronto.edu/~amnih/papers/ncelm.pdf
- [Mnih et al., NIPS'14] Recurrent Models of Visual Attention. http://papers.nips.cc/paper/5542-recurrent-models-of-visual-attention.pdf
- [Morin & Bengio, AISTATS'05] Hierarchical Probabilistic Neural Network Language Model. http://www.iro.umontreal.ca/~lisa/pointeurs/hierarchical-nnlm-aistats05.pdf

# References (3)


- [Sennrich, Haddow, Birch, ACL'16a]. Improving Neural Machine Translation Models with Monolingual Data. http://arxiv.org/pdf/1511.06709.pdf
- [Sennrich, Haddow, Birch, ACL'16b]. Neural Machine Translation of Rare Words with Subword Units. http://arxiv.org/pdf/1508.07909.pdf
- [Sutskever et al., NIPS'14] Sequence to Sequence Learning with Neural Networks. http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf
- [Tu, Lu, Liu, Liu, Li, ACL'16] Modeling Coverage for Neural Machine Translation. http://arxiv.org/pdf/1601.04811.pdf
- [Vaswani, Zhao, Fossum, Chiang, EMNLP'13] Decoding with Large-Scale Neural Language Models Improves Translation. http://www.isi.edu/~avaswani/NCE-NPLM.pdf
- [Wang, Cho, ACL'16]. Larger-Context Language Modelling with Recurrent Neural Network. http://aclweb.org/anthology/P/P16/P16-1125.pdf
- [Xu, Ba, Kiros, Cho, Courville, Salakhutdinov, Zemel, Bengio, ICML'15] Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. http://jmlr.org/proceedings/papers/v37/xuc15.pdf
- [Zoph, Knight, NAACL'16]. Multi-source neural translation. http://www.isi.edu/natural-language/mt/multi-source-neural.pdf
- [Zoph, Vaswani, May, Knight, NAACL'16] Simple, Fast Noise Contrastive Estimation for Large RNN Vocabularies. http://www.isi.edu/natural-language/mt/simple-fast-noise.pdf