NEW YORK UNIVERSITY

Natural Language Understanding

Kyunghyun Cho

Language Understanding?
Modelling?

LANGUAGE UNDERSTANDING

Topics: Natural Language Understanding

- What does it mean that a machine *understands natural languages*?
- Should we start reading *linguistics*?

"*Every time I fire a linguist, the performance of the recognizer goes up.*"

- Fred Jelinek (IBM), 1988

LANGUAGE UNDERSTANDING

Topics: Natural Language Understanding

- ***It's all about telling how likely a sentence is..***

- How likely is this sentence as an answer to the question?
  - Q. "*Who is the President of the United States?*"
  - *Likely* answer: "*Obama is the President of the U.S.*"
  - *Unlikely* answer: "*Tsipras is the President of America.*"

LANGUAGE UNDERSTANDING

Topics: Natural Language Understanding

- ***It's all about telling how likely a sentence is..***

- How likely is this sentence given this view?
    - *Likely*: "*Two dolphins are diving*"
    - *Unlikely*: "*Two men are flying*"

LANGUAGE UNDERSTANDING

Topics: Natural Language Understanding

It's all about telling how likely a sentence is..

Language Modelling

HOW LIKELY IS THIS SENTENCE?

Topics: Language Modelling

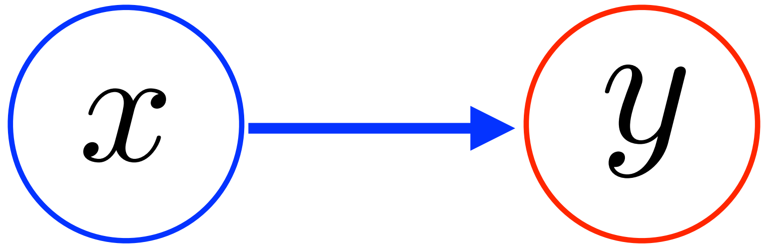
- A sentence $(x_1, x_2, \ldots, x_T)$
  - Ex) ("the", "cat", "is", "eating", "a", "sandwich", "on", "a", "couch")
- How likely is this sentence?
- In other words, what is the probability of $(x_1, x_2, \ldots, x_T)$?
  - i.e., $p(x_1, x_2, \ldots, x_T) =?$

HOW LIKELY IS THIS SENTENCE?

Topics: Probability 101 - Conditional Probability

- Joint probability $p(x, y)$
- Conditional probability $p(x|y)$
- Marginal probability $p(x)$ and $p(y)$

- They are related by $p(x, y) = p(x|y)p(y) = p(y|x)p(x)$



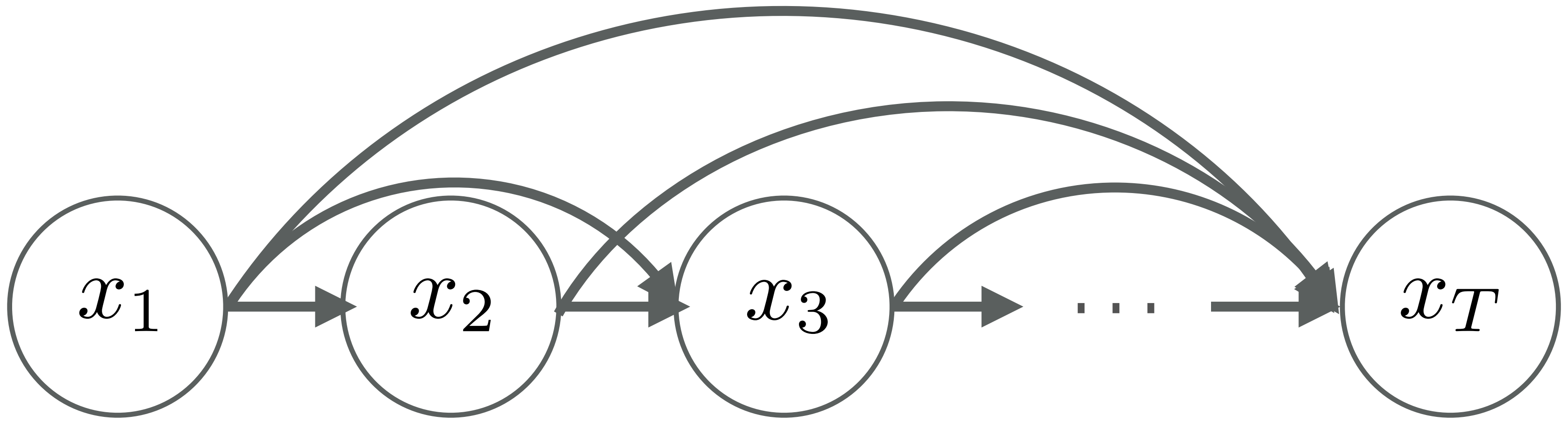
HOW LIKELY IS THIS SENTENCE?

Topics: Language Modelling as a Product of Conditionals

- Rewrite $p(x_1, x_2, \ldots, x_T)$ into

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

- Graphically,



STATISTICAL LM

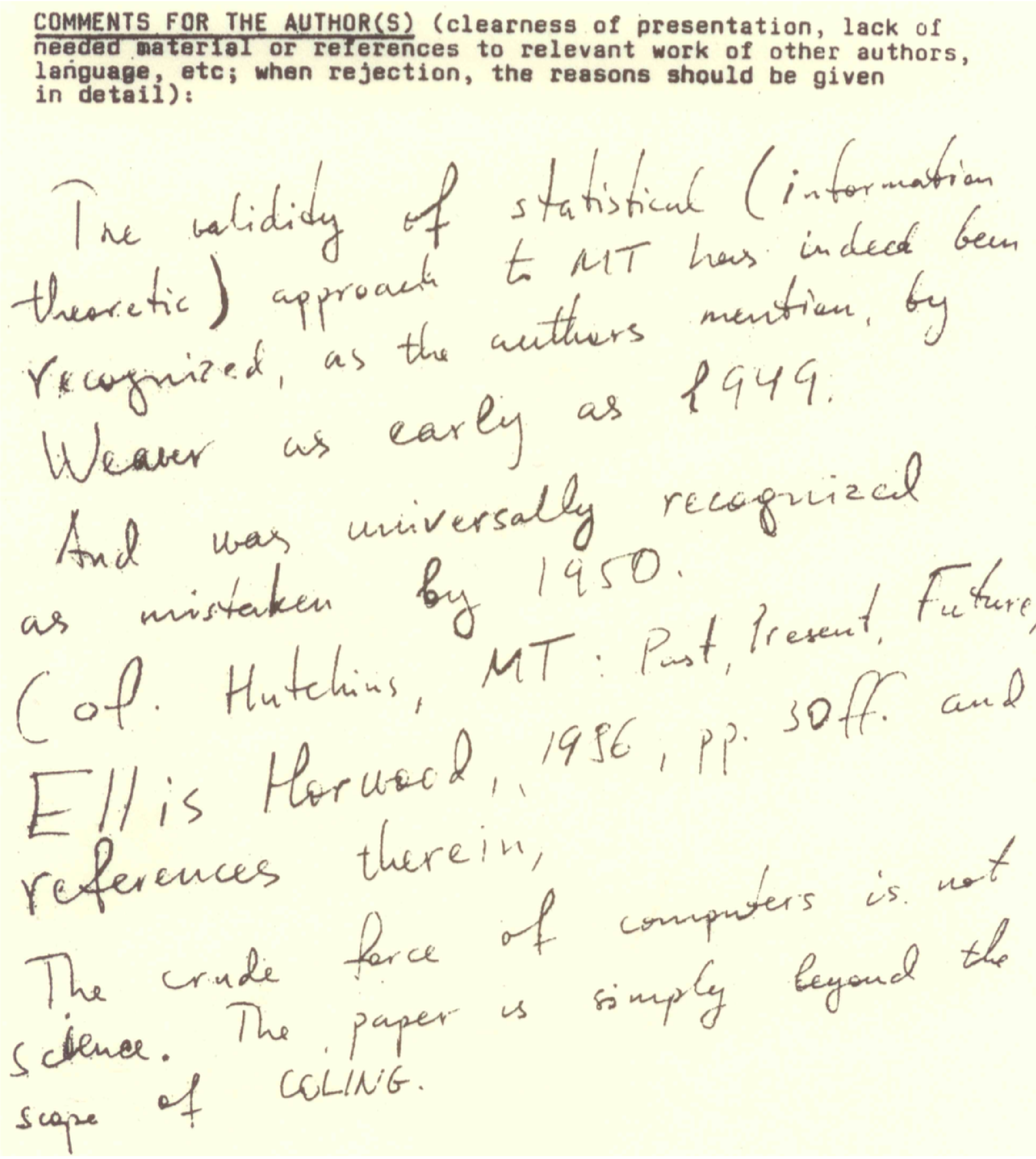
Topics: Statistical Language Modelling

- Maximize the (log-)probabilities of sentences in corpora

$$\max \mathbb{E}_D \left[\log p(x_1, x_2, \ldots, x_T)\right]$$

- Obvious to us, but not to everyone:
  - "*The validity of statistical (information theoretic) approach to MT has indeed been recognized … as early as 1949. And was universally recognized as mistaken [sic] by 1950. … The crude force of computers is not science.*"

(Review of Brown et al. (1990))



COMMENTS FOR THE AUTHOR(S) (clearness of presentation, lack of needed material or references to relevant work of other authors, language, etc; when rejection, the reasons should be given in detail):

The validity of statistical (information theoretic) approach to MT has indeed been recognized, as the authors mention, by Weaver as early as 1949. And was universally recognized as mistaken by 1950. (cf. Hutchins, MT: Past, Present, Future, Ellis Horwood, 1986, pp. 30ff. and references therein). The crude force of computers is not science. The paper is simply beyond the scope of COLING.

***n*-gram** Language Modelling

(Blunsom, 2015)

HOW LIKELY IS THIS SENTENCE?

Topics: Non-parametric Approach — n-gram modelling

- *n*-th order Markov assumption: why?

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

- Collect *n*-gram statistics from a *large* corpus:

$$p(x_t | x_{t-n}, \ldots, x_{t-1}) = \frac{\text{count}(x_{t-n}, \ldots, x_{t-1}, x_t)}{\text{count}(x_{t-n}, \ldots, x_{t-1})}$$

HOW LIKELY IS THIS SENTENCE?

Topics: Non-parametric Approach — n-gram modelling

- Ex) $p(\mathrm{i}, \mathrm{would}, \mathrm{like}, \mathrm{to}, \ldots, ., \langle/\mathrm{s}\rangle)$
- Unigram Modelling

  $p(\mathrm{i})p(\mathrm{would})p(\mathrm{like})\cdots p(\langle/\mathrm{s}\rangle)$
- Bigram Modelling

  $p(\mathrm{i})p(\mathrm{would}|\mathrm{i})p(\mathrm{like}|\mathrm{would})\cdots p(\langle/\mathrm{s}\rangle\,|.)$
- Trigram Modelling

  $p(\mathrm{i})p(\mathrm{would}|\mathrm{i})p(\mathrm{like}|\mathrm{i}, \mathrm{would})\cdots$

$\vdots$

word	unigram	bigram	trigram	4-gram
i	6.684	3.197	3.197	3.197
would	8.342	2.884	2.791	2.791
like	9.129	2.026	1.031	1.290
to	5.081	0.402	0.144	0.113
commend	15.487	12.335	8.794	8.633
the	3.885	1.402	1.084	0.880
rapporteur	10.840	7.319	2.763	2.350
on	6.765	4.140	4.150	1.862
his	10.678	7.316	2.367	1.978
work	9.993	4.816	3.498	2.394
.	4.896	3.020	1.785	1.510
</s>	4.828	0.005	0.000	0.000
average	8.051	4.072	2.634	2.251
perplexity	265.136	16.817	6.206	4.758

HOW LIKELY IS THIS SENTENCE?

Topics: n-gram modelling — *Two closely-related issues*

- Data Sparsity
  - # of all possible n-grams: $|V|^n$, where $|V|$: size of vocabulary

$$p(\text{a}, \text{tenured}, \text{professor}, \text{like}, \text{drinking}, \text{whiskey}, .) =$$
$$p(\text{a})p(\text{tenured}|\text{a})\,\underbrace{p(\text{professor}|\text{a}, \text{tenured})}_{=0}$$
$$p(\text{likes}|\text{tenured}, \text{professor}) \cdots p(.|\text{drinking}, \text{whiskey})$$
$$= 0$$

HOW LIKELY IS THIS SENTENCE?

Topics: n-gram modelling — *Two closely-related issues*

- Conventional Solutions to Data Sparsity:
  - Smoothing:

$$p(x_t|x_{t-n},\ldots,x_{t-1}) = \frac{\text{count}(x_{t-n},\ldots,x_{t-1},x_t)+\alpha}{\text{count}(x_{t-n},\ldots,x_{t-1})+\alpha|V|}$$

(add-α smoothing)

  - Backoff:

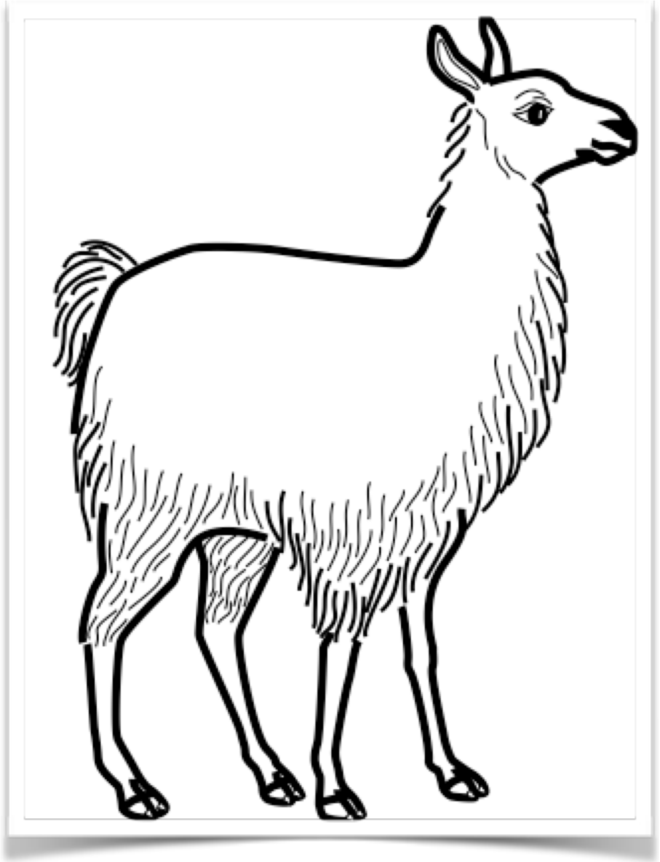
$$p(x_t|x_{t-n},\ldots,x_{t-1}) = \begin{cases} \alpha_n(x_t|x_{t-n},\ldots,x_{t-1}), \\ \qquad \text{if count}_n(x_{t-n},\ldots,x_t)>0 \\ d_n(x_{t-n},\ldots,x_{t-1})p(x_t|x_{t-n+1}\ldots,x_{t-1}), \\ \qquad \text{otherwise} \end{cases}$$

(α_n: adjusted prediction model, d_n: discount factor)

HOW LIKELY IS THIS SENTENCE?

Topics: n-gram modelling — *Two closely-related issues*

- Lack of Generalization
  - (chases, a, **dog**), (chases, a, **cat**), (chases, a, **rabbit**)
  - (chases, a, **llama**)=?



Neural Language Modelling

LANGUAGE MODELLING

Topics: Neural Language Modelling

- ~~Non parametric estimator~~ $\longrightarrow$ Parametric estimator

$$p(x_t|x_{t-n},\ldots,x_{t-1}) = \cancel{\frac{\text{count}(x_{t-n},\ldots,x_{t-1},x_t)}{\text{count}(x_{t-n},\ldots,x_{t-1})}}$$

$$= f_{x_t}(x_{t-n},\ldots,x_{t-1})$$

LANGUAGE MODELLING

Topics: Neural Language Modelling

$$p(x_t | x_{t-n}, \ldots, x_{t-1}) = f_{x_t}(x_{t-n}, \ldots, x_{t-1})$$

- Building a neural language model (Bengio et al., 2000)

(1) *1*-of-*K* encoding of each word $x_{t'}$

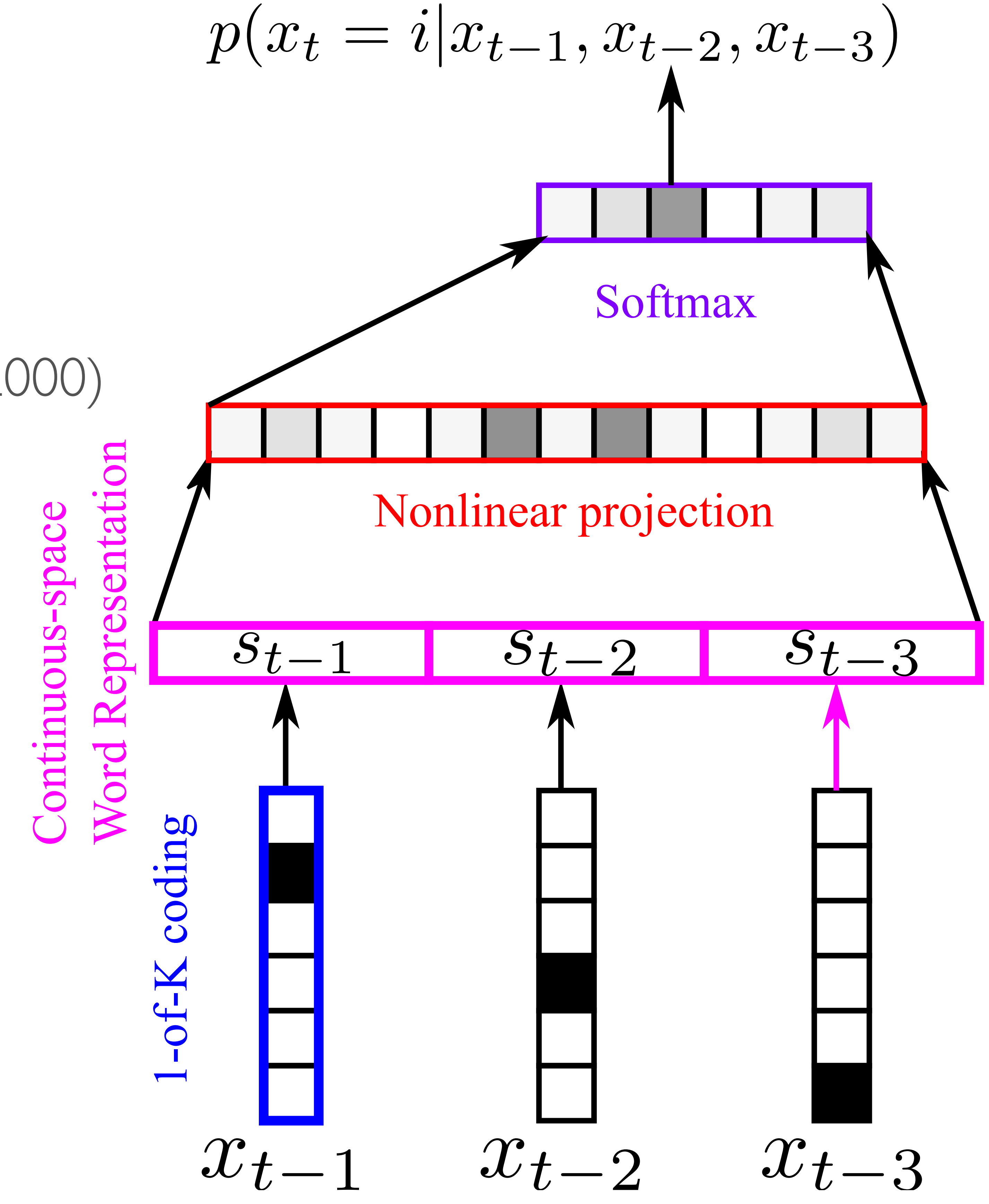
(2) Continuous space word representation

$$s_{t'} = W^\top x_{t'}, \text{ where } W \in \mathbb{R}^{|V| \times d}$$

(3) Nonlinear hidden layer

$$h = \tanh(U^\top [s_{t-1}; s_{t-2}; \cdots ; s_{t-n}] + b)$$

$$\text{, where } U \in \mathbb{R}^{nd \times d'} \text{ and } b \in \mathbb{R}^{d'}$$

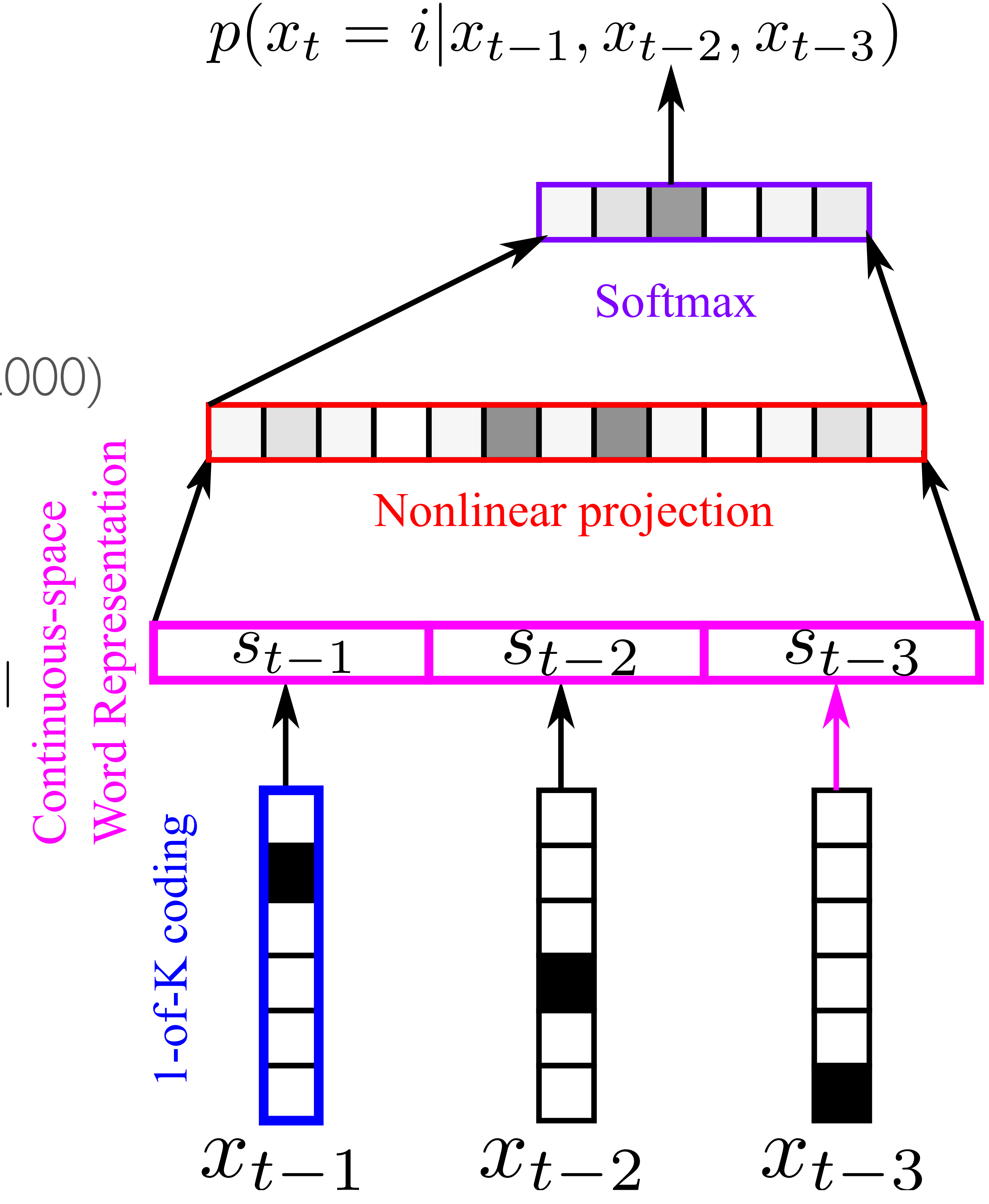


LANGUAGE MODELLING

Topics: Neural Language Modelling

$$p(x_t|x_{t-n},\ldots,x_{t-1}) = f_{x_t}(x_{t-n},\ldots,x_{t-1})$$

- Building a neural language model (Bengio et al., 2000)



(1)Unnormalized probabilities

$$y = Vh + c, \text{ where } V \in \mathbb{R}^{|V|\times d'} \text{ and } c \in \mathbb{R}^{|V|}$$

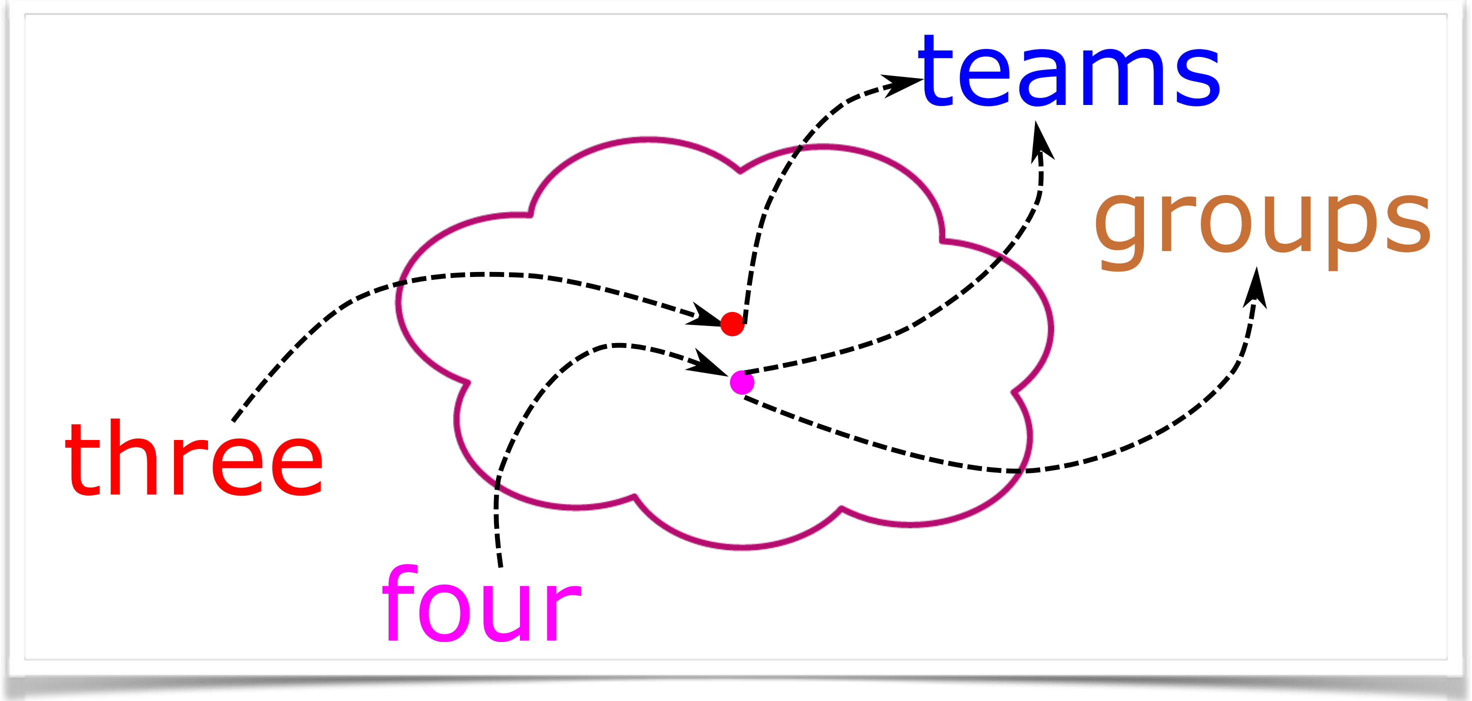
(2)Softmax normalization

$$p(x_t = i|x_{t-n},\ldots,x_{t-1}) = \frac{\exp(y_i)}{\sum_{j=1}^{|V|}\exp(y_j)}$$

LANGUAGE MODELLING

Topics: Neural LM generalizes to *unseen* n-gram's

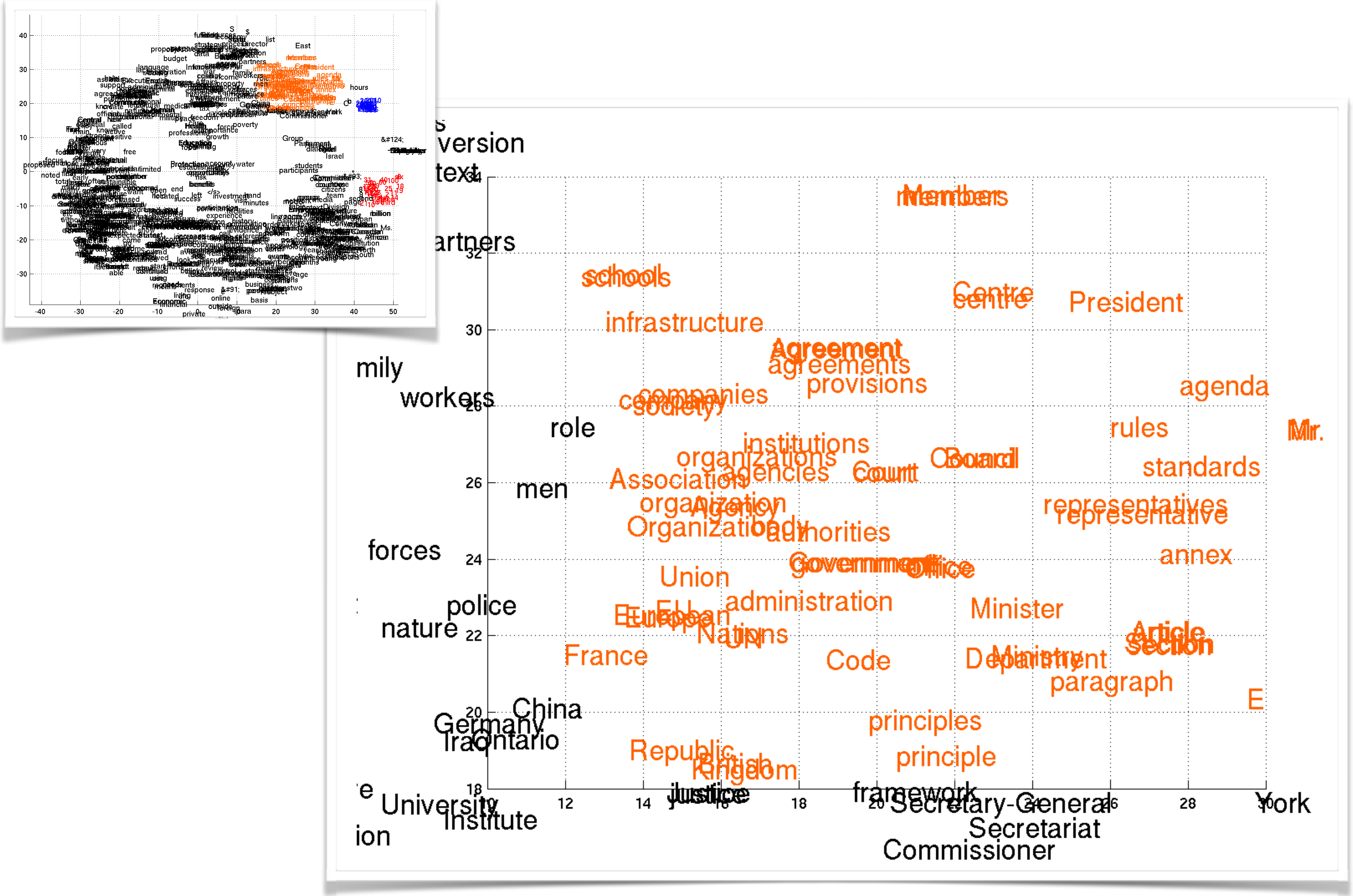
- Example sentences
  - there are three teams left for the qualification.
  - four teams have passed the first round.
  - four groups are playing in the field.

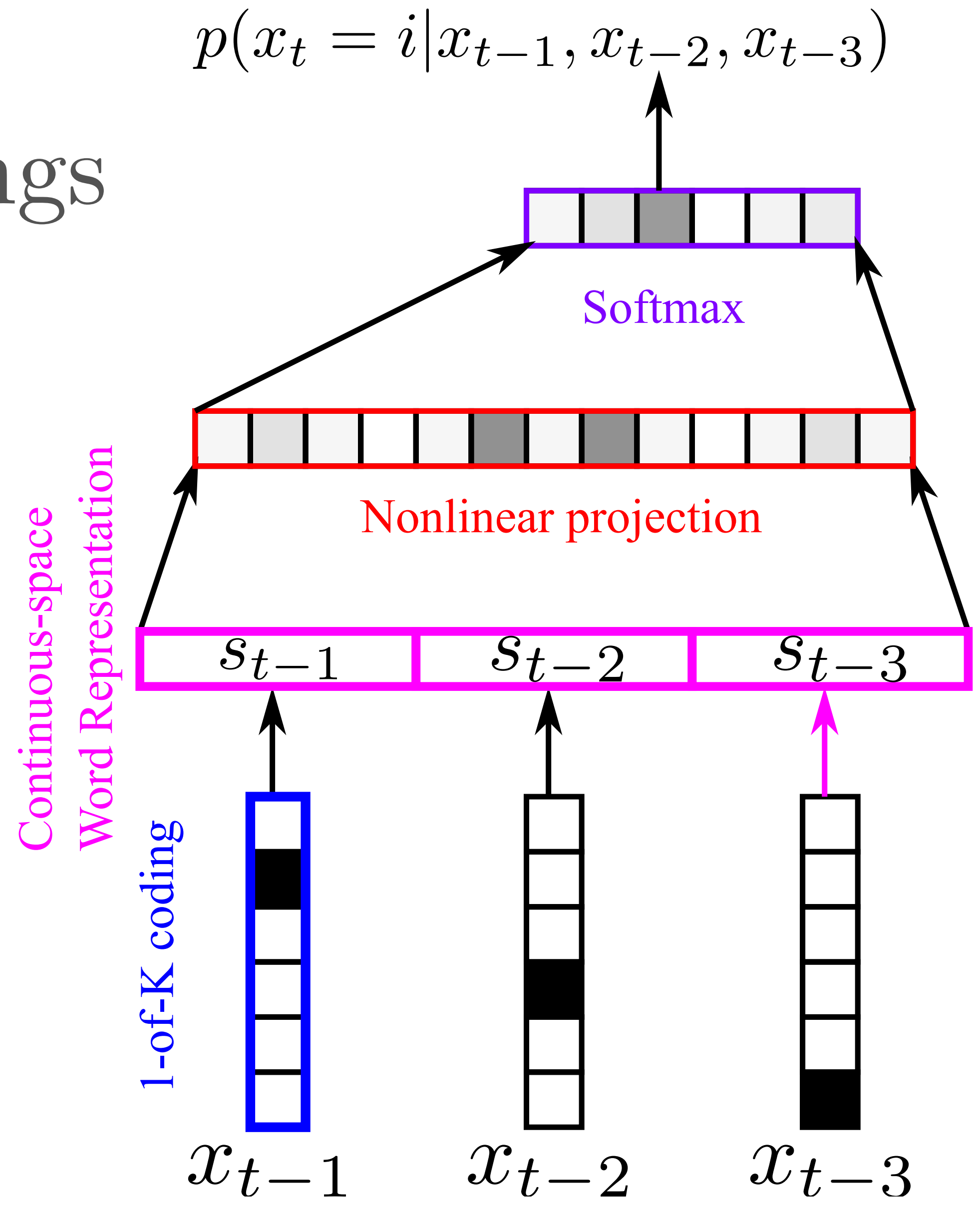


- How likely is groups followed by three?
- *Why?*

LANGUAGE MODELLING

Topics: Continuous-space representation — Embeddings





Q&A

Non-Markovian Language Modelling

LANGUAGE MODELLING

Topics: Markov Assumption

- Markov Assumption in *n*-gram modeling

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

- Issue: Dependency beyond the context window is ignored
  - Ex) *the same **stump** which had impaled the car of many a guest in the past thirty years and which he refused to have **removed***

LANGUAGE MODELLING

Topics: Non-Markovian Language Modelling

- Directly model the original conditional probabilities

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

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

- *How can we let* f *take variable-length input?*

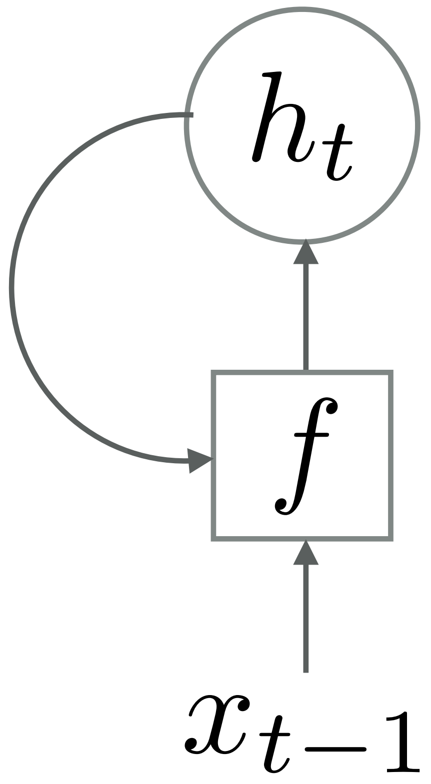
LANGUAGE MODELLING

Topics: Language Modelling via Recursion

- Directly model the original conditional probabilities

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

- Recursive Construction of f
  - Initial Condition: $h_0 = 0$
  - Recursion: $h_t = f(x_{t-1}, h_{t-1})$
- *We call* h_t *an internal hidden state or* ***memory***
  - h_t summarizes/memorizes the history from x_1 up to x_{t-1}



LANGUAGE MODELLING

Topics: Language Modelling via Recursion

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

  (1) Initialization: $h_0 = 0$

  (2) Recursion

  (1) $h_1 = f(h_0, \text{the})$

  (2) $h_2 = f(h_1, \text{cat})$

  (3) $h_3 = f(h_2, \text{is})$

  (3) Readout: $p(\text{eating}|\text{the}, \text{cat}, \text{is}) = g(h_3)$

- It works for any number of context words

RNN Language Modelling

LANGUAGE MODELLING

Topics: Recurrent neural network language model

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

  (1) Initialization: $h_0 = 0 \longrightarrow p(\text{the}) = g(h_0)$

  (2) Recursion with Readout

  (1) $h_1 = f(h_0, \text{the}) \longrightarrow p(\text{cat}|\text{the}) = g(h_1)$

  (2) $h_2 = f(h_1, \text{cat}) \longrightarrow p(\text{is}|\text{the}, \text{cat}) = g(h_2)$

  (3) $h_3 = f(h_2, \text{is}) \longrightarrow p(\text{eating}|\text{the}, \text{cat}, \text{is}) = g(h_3)$

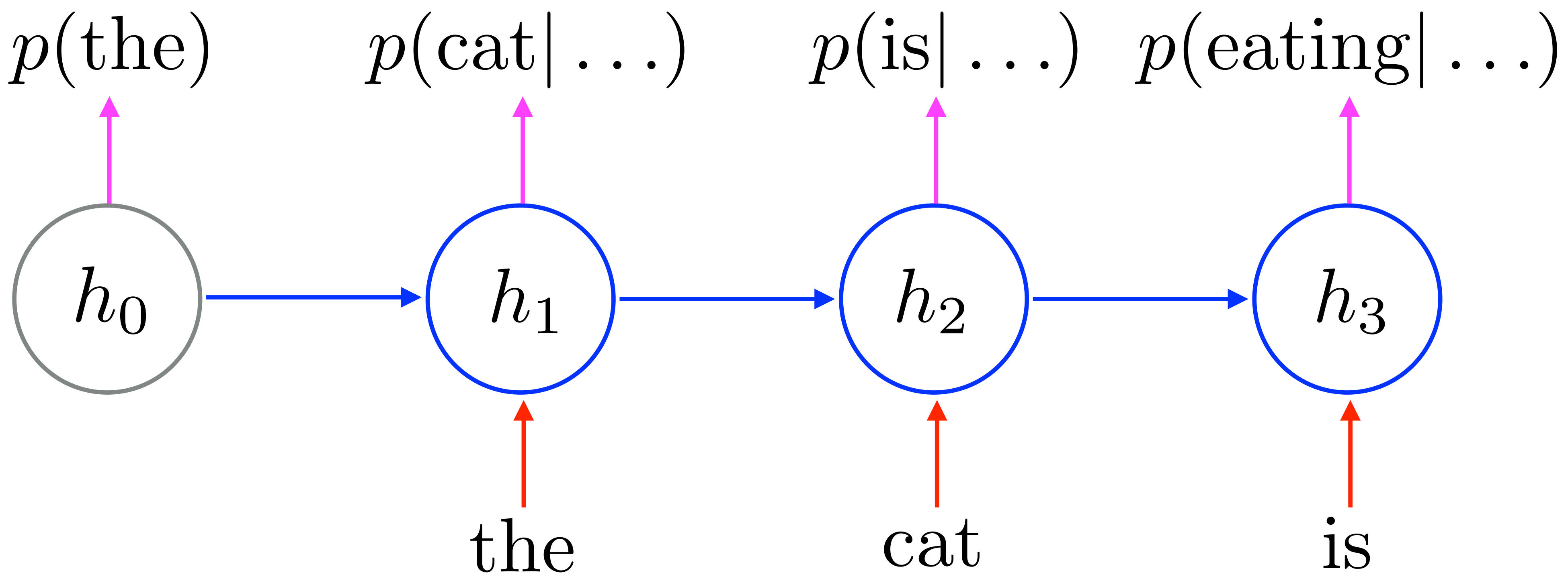
  (3) Combination: $p(\text{the}, \text{cat}, \text{is}, \text{eating}) = g(h_0)g(h_1)g(h_2)g(h_3)$

- Read, Update and Predict

LANGUAGE MODELLING

Topics: Recurrent neural network language model

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

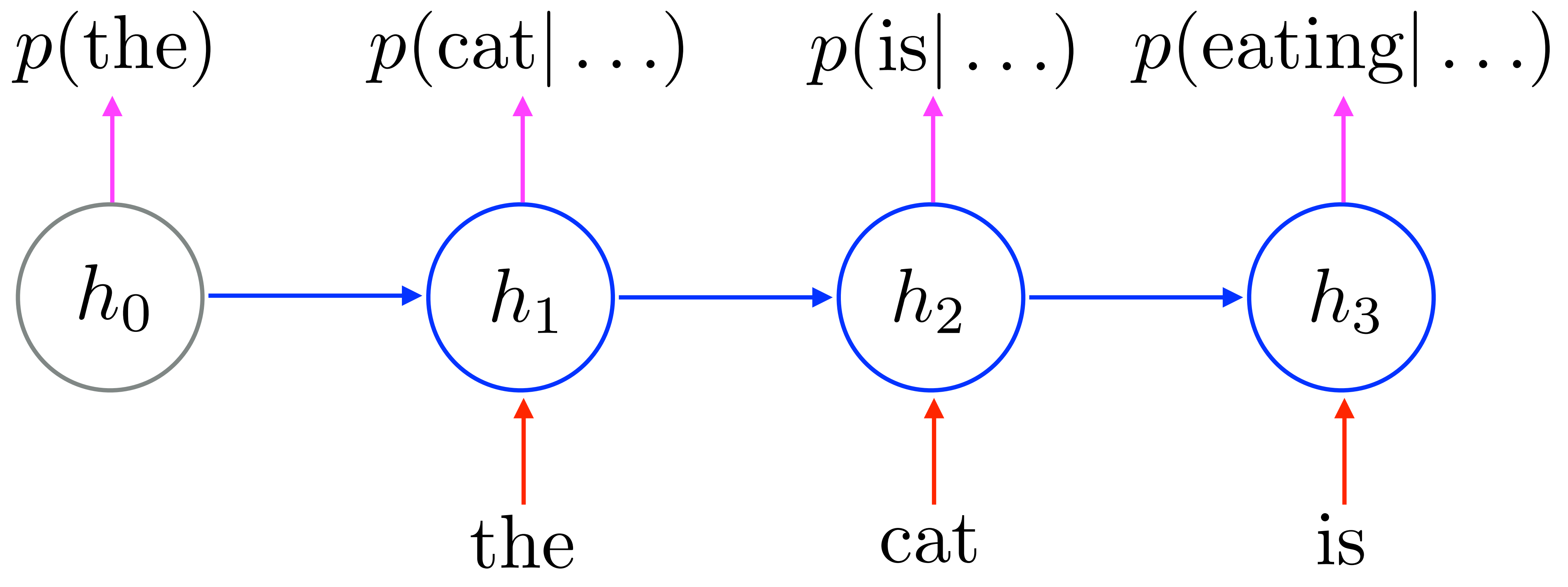


- Read, Update and Predict

LANGUAGE MODELLING

Topics: Building an RNN Language Model

- What do we need?
  - Transition Function $h_t = f(h_{t-1}, x_{t-1})$
  - Output/Readout Function $p(x_t = w | x_1, \ldots, x_{t-1}) = g_w(h_t)$



LANGUAGE MODELLING

Topics: Building an RNN Language Model - Transition Function

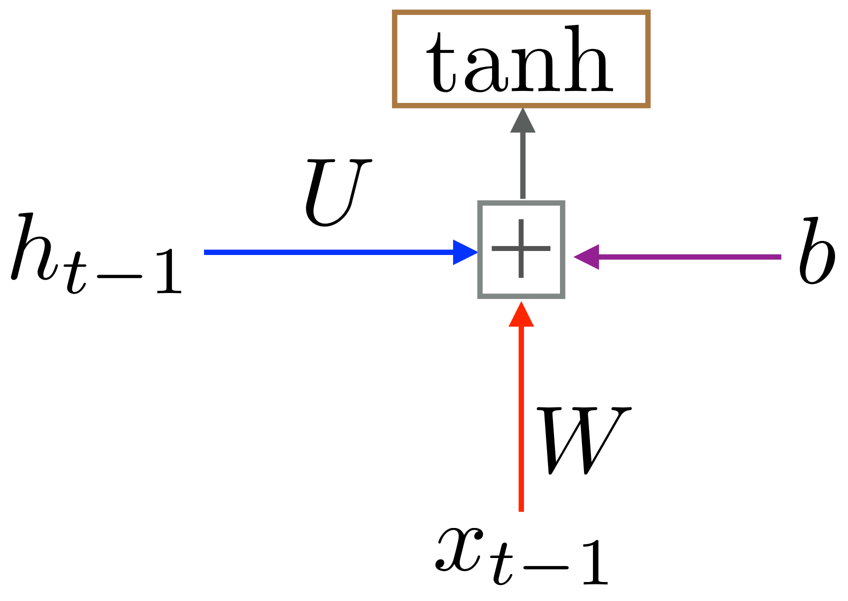
- **Inputs**
  - Input $x_{t-1} \in \{0,1\}^{|V|}$: one-hot vector, i.e., $x_{t-1} = w \in \{1, \ldots, |V|\}$
  - Hidden state $h_{t-1} \in \mathbb{R}^d$
- **Parameters**
  - Input weight matrix $W \in \mathbb{R}^{d \times |V|}$ *(often called word embeddings)*
  - Transition weight matrix $U \in \mathbb{R}^{d \times d}$
  - Bias vector $b \in \mathbb{R}^d$

LANGUAGE MODELLING

Topics: Building an RNN Language Model - Transition Function

- **Inputs**: $x_{t-1} \in \{0,1\}^{|V|}$, $h_{t-1} \in \mathbb{R}^d$
- **Parameters**: $W \in \mathbb{R}^{d \times |V|}$, $U \in \mathbb{R}^{d \times d}$, $b \in \mathbb{R}^d$
- Naive **Transition Function**

$$h_t = \tanh(W x_{t-1} + U h_{t-1} + b)$$



(1) Continuous-space Representation of word: $W x_{t-1}$

(2) Linear Transformation of the Previous Hidden State: $U h_{t-1}$

(3) Additive combination of x_{t-1} and h_{t-1} together with b

(4) Point-wise nonlinear transformation

LANGUAGE MODELLING

Topics: Building an RNN Language Model - Readout Function

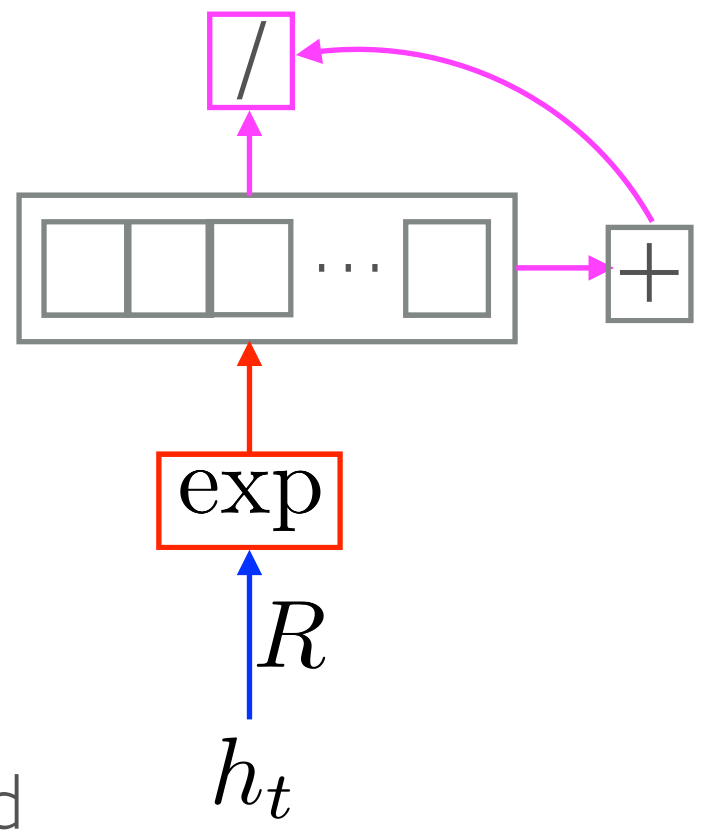
- **Inputs**
  - (Current) Hidden State $h_t \in \mathbb{R}^d$
- **Parameters**
  - Output matrix $R \in \mathbb{R}^{|V| \times d}$ *(often called* ***target*** *word embeddings)*
  - Bias vector $c \in \mathbb{R}^{|V|}$

LANGUAGE MODELLING

Topics: Building an RNN Language Model - Readout Function

- **Inputs** $h_t \in \mathbb{R}^d$
- **Parameters** $R \in \mathbb{R}^{|V| \times d}$, $c \in \mathbb{R}^{|V|}$
- *Softmax* **Readout Function**

$$p(x_t = w | x_{<t}) = g_w(h_t) = \frac{\exp(R_w^\top h_{t-1} + c_w)}{\sum_{i=1}^{|V|} \exp(R_i^\top h_{t-1} + c_i)}$$



(1) Linear projection of the hidden state for each possible target word

$$v_i = R_i^\top h_{t-1} \text{ for all } i = 1, \ldots, |V|$$

(3) Transform each projected vector v_i to be *positive* $\tilde{p}_i = \exp(v_i)$

(4) Normalize $\tilde{p}_i$'s to make them into *probabilities* of the i-th target words

LANGUAGE MODELLING

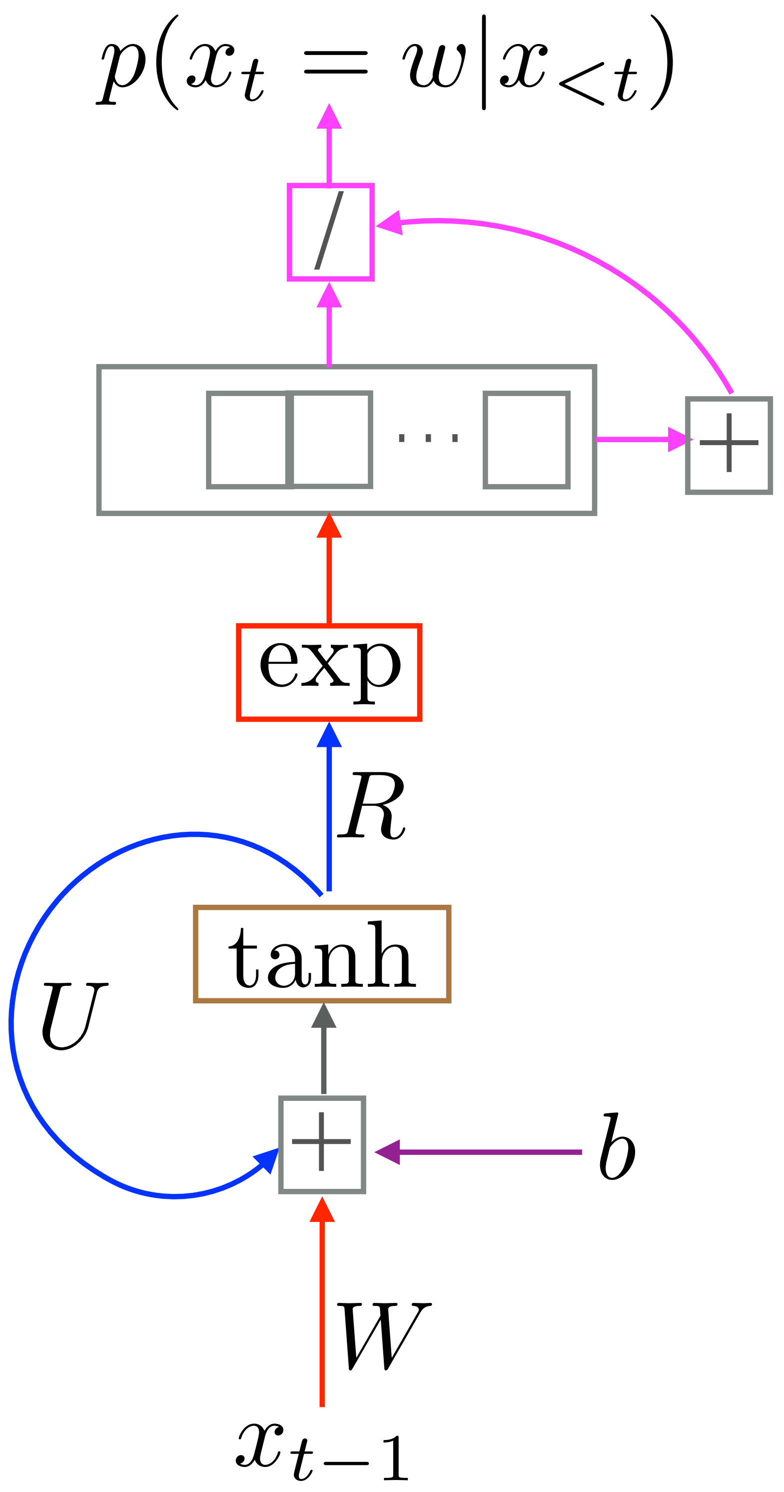
Topics: Building an RNN Language Model

- Recursion and Readout:
  - Recursion

    $$h_t = \tanh(Wx_{t-1} + Uh_{t-1} + b)$$

  - Readout/Output

    $$p(x_t = w | x_{<t}) = \frac{\exp(R_w^\top h_{t-1})}{\sum_{i=1}^{|V|} \exp(R_i^\top h_{t-1})}$$



Training RNN-LM

LANGUAGE MODELLING

Topics: Cost Function $J(\Theta)$

- Log-Probability of a sentence $(x_1, x_2, \ldots, x_T)$

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

- Train an RNN LM to maximize the log-prob's of *training* sentences
- Given a training set of N sentences: $\{(x_1^1, \ldots, x_{T_1}^1), \ldots, (x_1^N, \ldots, x_{T_N}^N)\}$

$$\text{maximize}_\Theta \frac{1}{N} \sum_{n=1}^{N} \log p(x_1^n, \ldots, x_{T_n}^n)$$

$$\iff \text{minimize}_\Theta J(\Theta) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \log p(x_t^n | x_1^n \ldots, x_{t-1}^n)$$

LANGUAGE MODELLING

Topics: Minibatch Stochastic Gradient Descent - Recap

(1)Randomly select a minibatch of N' sentences: $D = \left\{x^1, \ldots, x^{N'}\right\}$

(2)Compute the gradient of per-sample cost w.r.t. Θ : $\nabla J(\Theta, x^n)$

(3)Compute the minibatch gradient:

$$\nabla J(\Theta, D) = \frac{1}{N'} \sum_{n=1}^{N'} \nabla J(\Theta, x^n)$$

(4)Update the parameters Θ

$$\Theta \leftarrow \Theta + \eta \nabla J(\Theta, D)$$

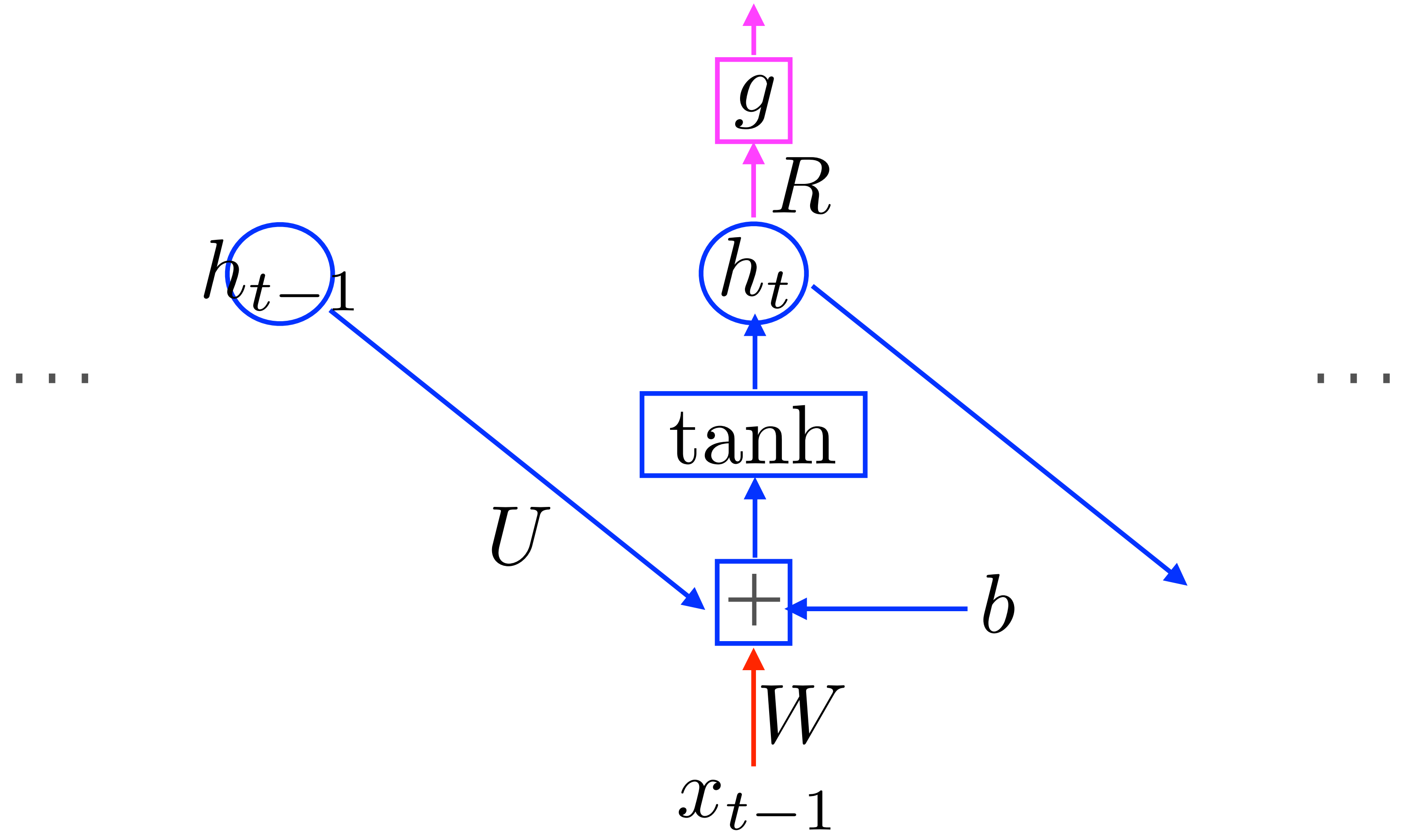
(5)Repeat until *convergence*

LANGUAGE MODELLING

Topics: Backpropagation through time

- Decomposition of a per-sample cost function $J(\Theta, x) = -\sum_{t=1}^{T} J_t(\Theta, x_t)$
- Unrolled Computational Graph

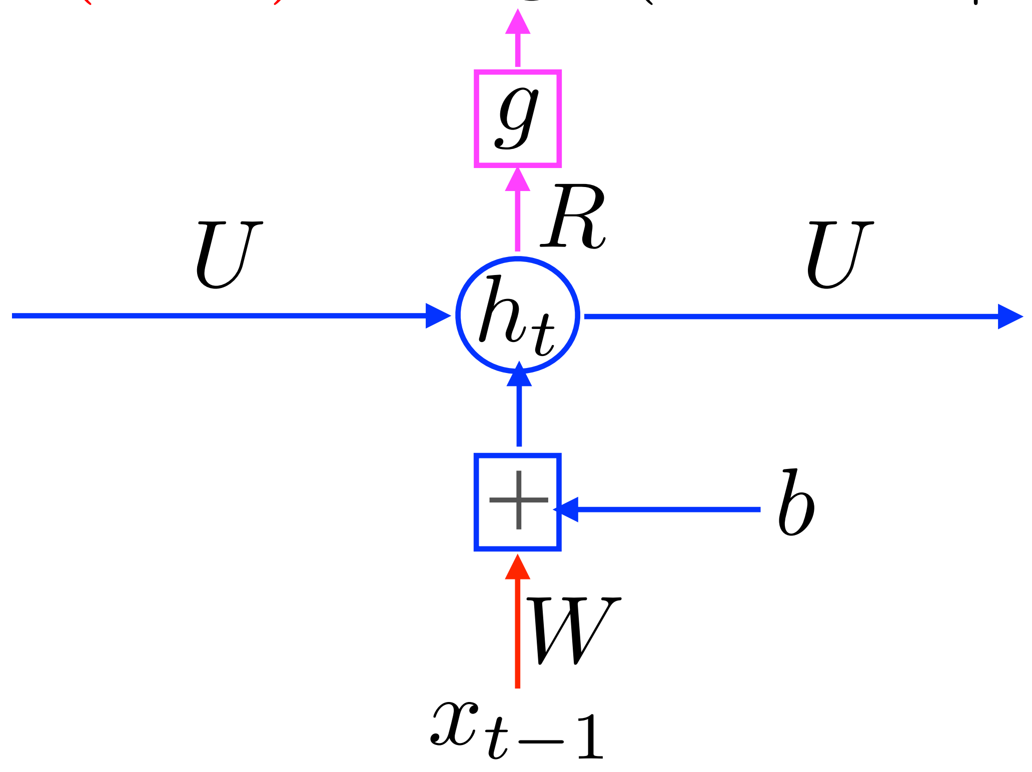
$$J_t(\Theta, \hat{x}) = \log p(x_t = \hat{x}_t | x_{<t})$$



LANGUAGE MODELLING

Topics: Backpropagation through time

$$J_t(\Theta, \hat{x}) = \log p(x_t = \hat{x}_t | x_{<t})$$



(1)Initialize $\nabla_R, \nabla_U, \nabla_W, \nabla_b$ and $t = T$

(1)The per-step cost derivative: $\frac{\partial J_t}{\partial g}$

(2)Gradient w.r.t. $R: \frac{\partial J_t}{\partial g} \frac{\partial g}{\partial R}$

(3)Gradient w.r.t. $h_t: \frac{\partial J_t}{\partial g} \frac{\partial g}{\partial h_t} + \frac{\partial J_{>t}}{\partial h_{t+1}} \frac{\partial h_{t+1}}{\partial h_t}$

(4)Gradient w.r.t. $U: \frac{\partial J_{\geq t}}{\partial h_t} \frac{\partial h_t}{\partial U}$

(5)Gradient w.r.t W and $b: \frac{\partial J_{\geq t}}{\partial h_t} \frac{\partial h_t}{\partial W}, \frac{\partial J_{\geq t}}{\partial h_t} \frac{\partial h_t}{\partial b}$

(2)Update the parameter gradient and repeat until $t = 1$

$$\nabla_R \leftarrow \nabla_R + \frac{\partial J_t}{\partial R}, \nabla_U \leftarrow \nabla_U + \frac{\partial J_{\geq t}}{\partial U}$$

$$\nabla_W \leftarrow \nabla_W + \frac{\partial J_{\geq t}}{\partial W}, \nabla_b \leftarrow \nabla_b + \frac{\partial J_{\geq t}}{\partial b}$$

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

Q&A

Code: https://github.com/nyu-dl/dl4mt-tutorial/tree/master/session0

Gated Recurrent Units

GATED RECURRENT UNITS

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

Topics: Temporal Dependency and Vanishing Gradient

- How much influence does h_t have on $\log p(x_{t+n}|x_{<t+n})$?

$$\frac{\partial J_{t+n}}{\partial h_t} = \frac{\partial J_{t+n}}{\partial g} \frac{\partial g}{\partial h_{t+N}} \frac{\partial h_{t+N}}{\partial h_{t+N-1}} \cdots \frac{\partial h_{t+1}}{\partial h_t}$$

- With the naive transition function?

$$\frac{\partial h_{t+1}}{\partial h_t} = U^\top \frac{\partial \tanh(a)}{\partial a}, \text{where } a = W x_t + U h_t + b$$

- Let's rewrite it

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

Bengio et al. (1994)

GATED RECURRENT UNITS

Topics: Temporal Dependency and Vanishing Gradient

- Upper bound on the **norm of the gradient** w.r.t. h_t ?

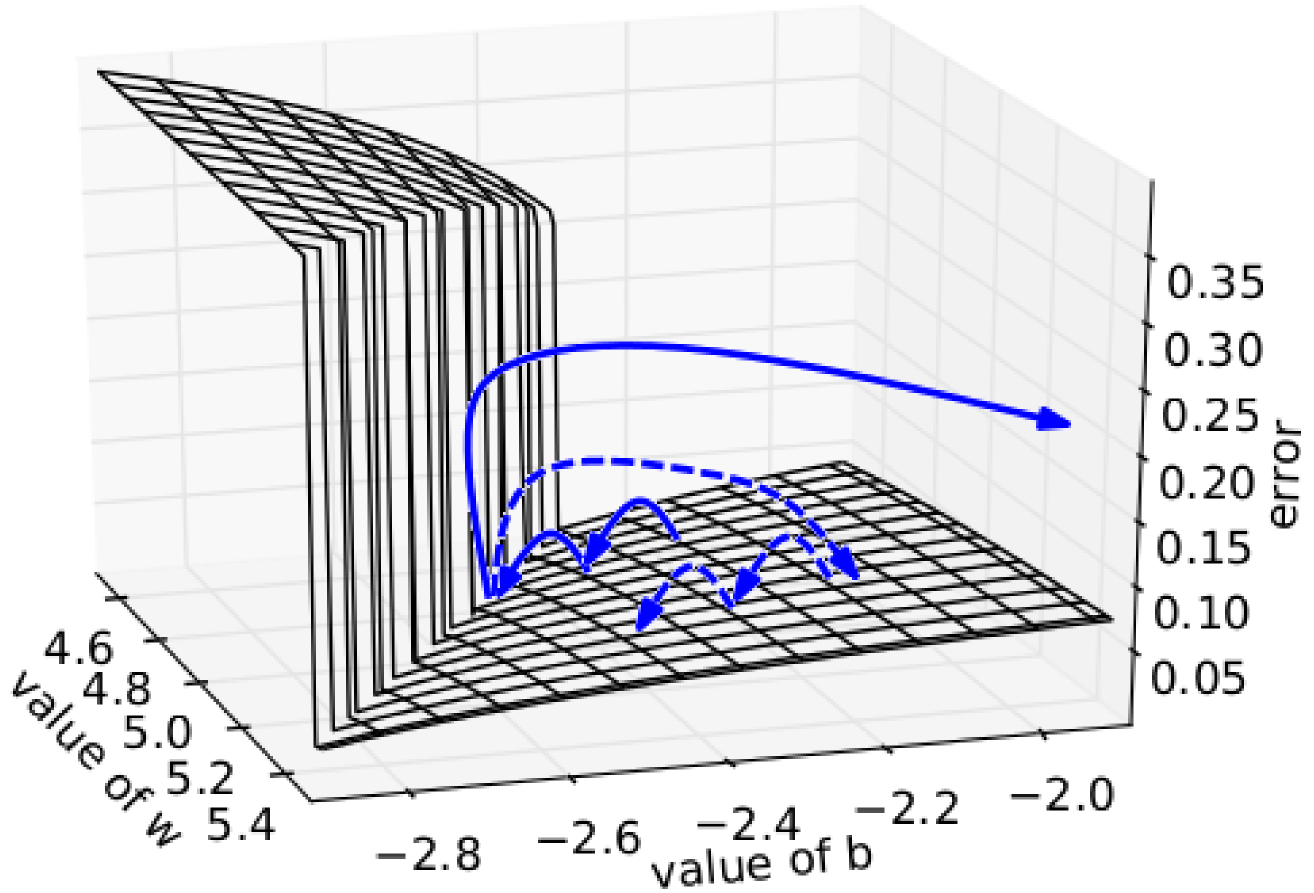
$$\left\| \prod_{n=1}^{N} U^\top \mathrm{diag}\left(\frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}} \right) \right\| \leq \prod_{n=1}^{N} \|U^\top\| \prod_{n=1}^{N} \left\| \frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}} \right\|$$

- **Observations**
  (1) *Vanishing gradient* when $\lambda_{\max}(U) < 1$: $\prod_{n=1}^{N} \|U^\top\| \to 0$
  (2) *Vanishing gradient* when the units are *saturated*: $\frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}} \to 0$
  (3) Potentially, *exploding gradient* when $\lambda_{\max}(U) > 1$
- **Problem**: *It's likely that there's no learning signal!*

GATED RECURRENT UNITS

Topics: Exploding gradient is *less* problematic

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



Pascanu et al. (2013)

(1)Gradient norm clipping

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

(2)Element-wise gradient clipping

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

GATED RECURRENT UNITS

Topics: But, vanishing gradient is very problematic

- Why does the gradient vanish?

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

- Can we simply "maximize" $\left\| \frac{\partial h_{t+N}}{\partial h_t} \right\|$?
  - "*we need to force the network to increase the norm of* $\frac{\partial h_{t+N}}{\partial h_t}$ *at the expense of larger errors*"
- Pascanu et al. (2013)
  - Regularize $\Omega = \sum_k \Omega_k = \sum_k \left(\frac{\left\| \frac{\partial \mathcal{E}}{\partial \mathbf{x}_{k+1}} \frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{x}_k} \right\|}{\left\| \frac{\partial \mathcal{E}}{\partial \mathbf{x}_{k+1}} \right\|} - 1 \right)^2$

GATED RECURRENT UNITS

Topics: But, vanishing gradient is very problematic

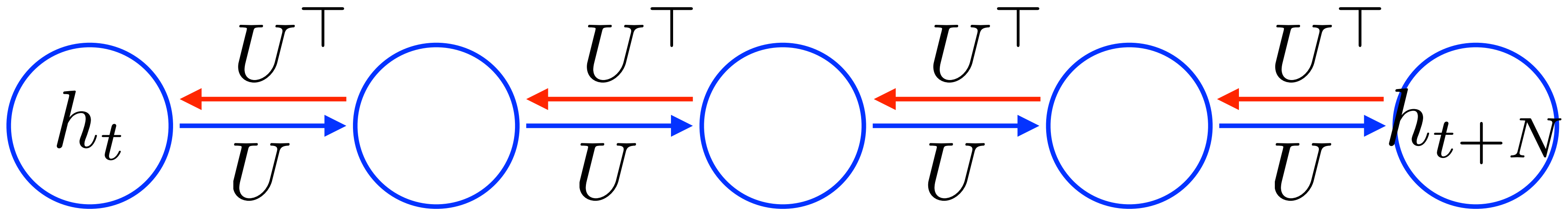
- Why does the gradient vanish?

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

- Perhaps, it is a problem with the naive transition function…

$$h_t = \tanh(W x_{t-1} + U h_{t-1} + b)$$

- Error is backpropagated through *every intermediate node*



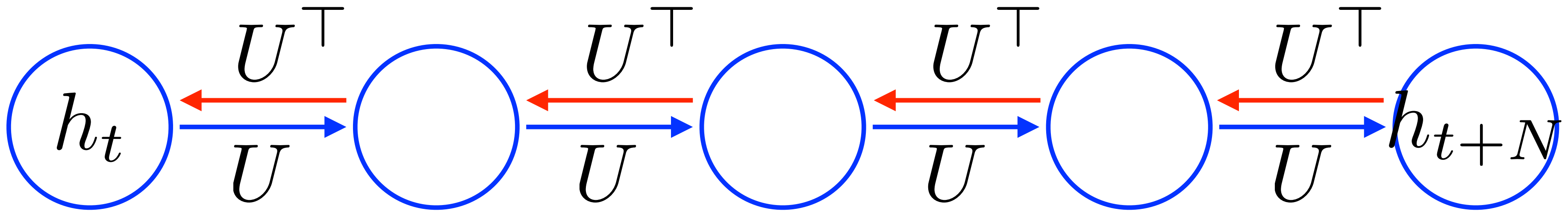
GATED RECURRENT UNITS

Topics: But, vanishing gradient is very problematic

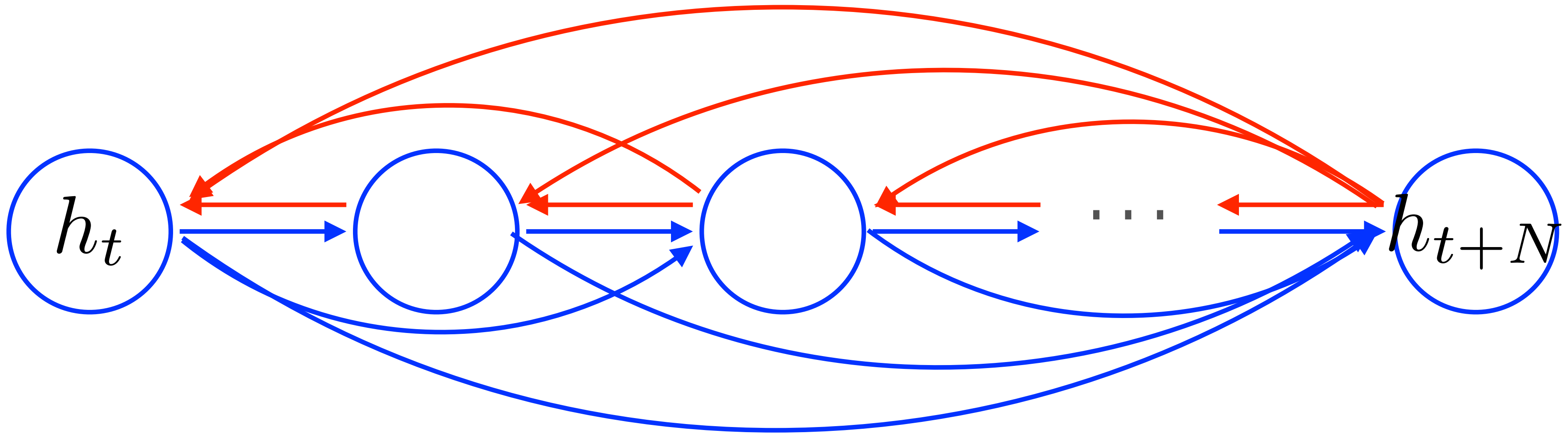
- Perhaps, it is a problem with the naive transition function…

$$h_t = \tanh(W x_{t-1} + U h_{t-1} + b)$$

- Error is backpropagated through *every intermediate node*



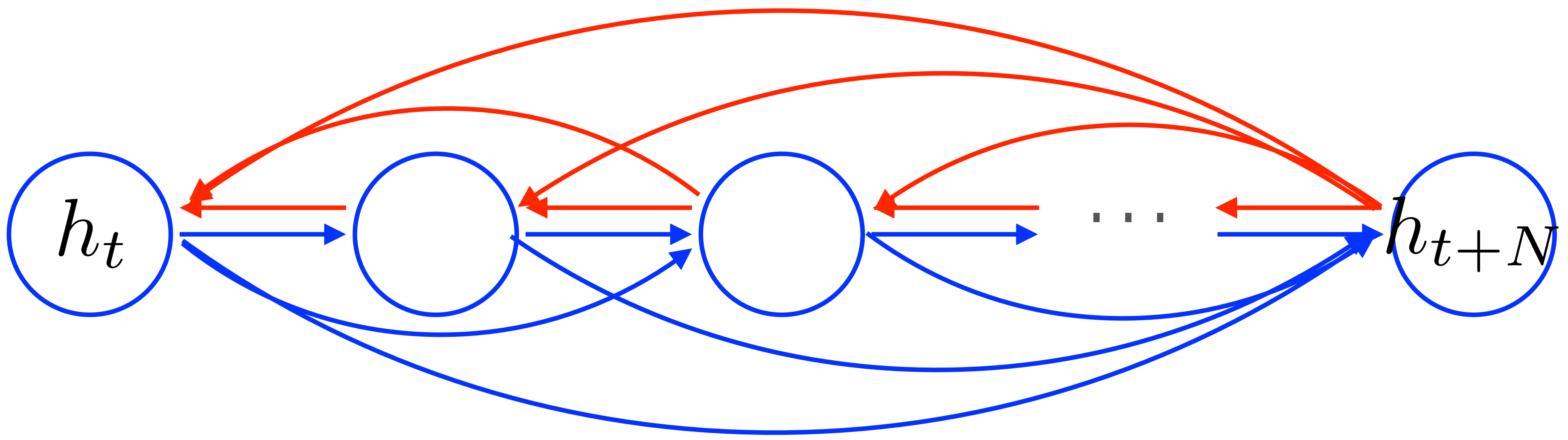
- Temporal *shortcut* connections



GATED RECURRENT UNITS

Topics: Gated Recurrent Units (GRU)

- Temporal shortcut connections



- *Adaptive* Leaky integration

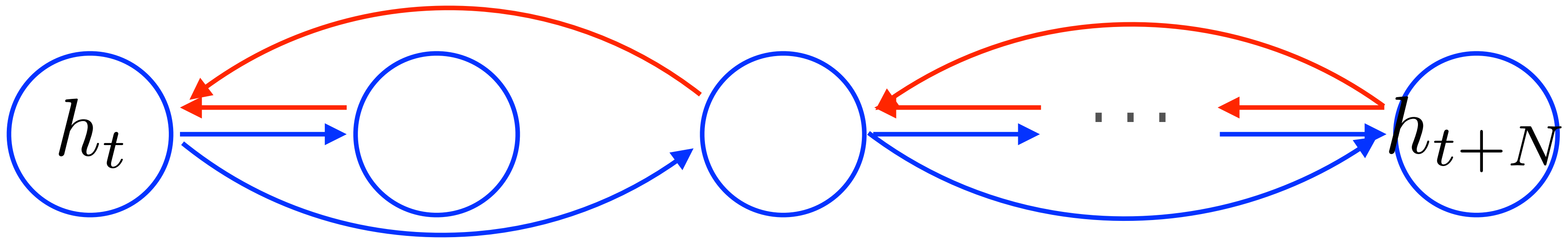
$$h_t = (1 - u_t) \odot h_{t-1} + u_t \odot \tilde{h}_t$$

- *Update* gate $u_t = \sigma(W_u x_{t-1} + U_u h_{t-1} + b_u)$
- Candidate state $\tilde{h}_t = \tanh(W x_{t-1} + U h_{t-1} + b)$

GATED RECURRENT UNITS

Topics: Gated Recurrent Units (GRU)

- Pruning connections: avoids the diffusion of signal



- *Adaptive* Reset

$$\tilde{h}_t = \tanh(Wx_{t-1} + U(r_t \odot h_{t-1}) + b)$$

- *Reset gate*

$$r_t = \sigma(W_r x_{t-1} + U_r h_{t-1} + b_r)$$

GATED RECURRENT UNITS

Topics: Gated Recurrent Units (GRU)

- *Update* and *Reset* gates

$$u_t = \sigma(W_u x_{t-1} + U_u h_{t-1} + b_u)$$

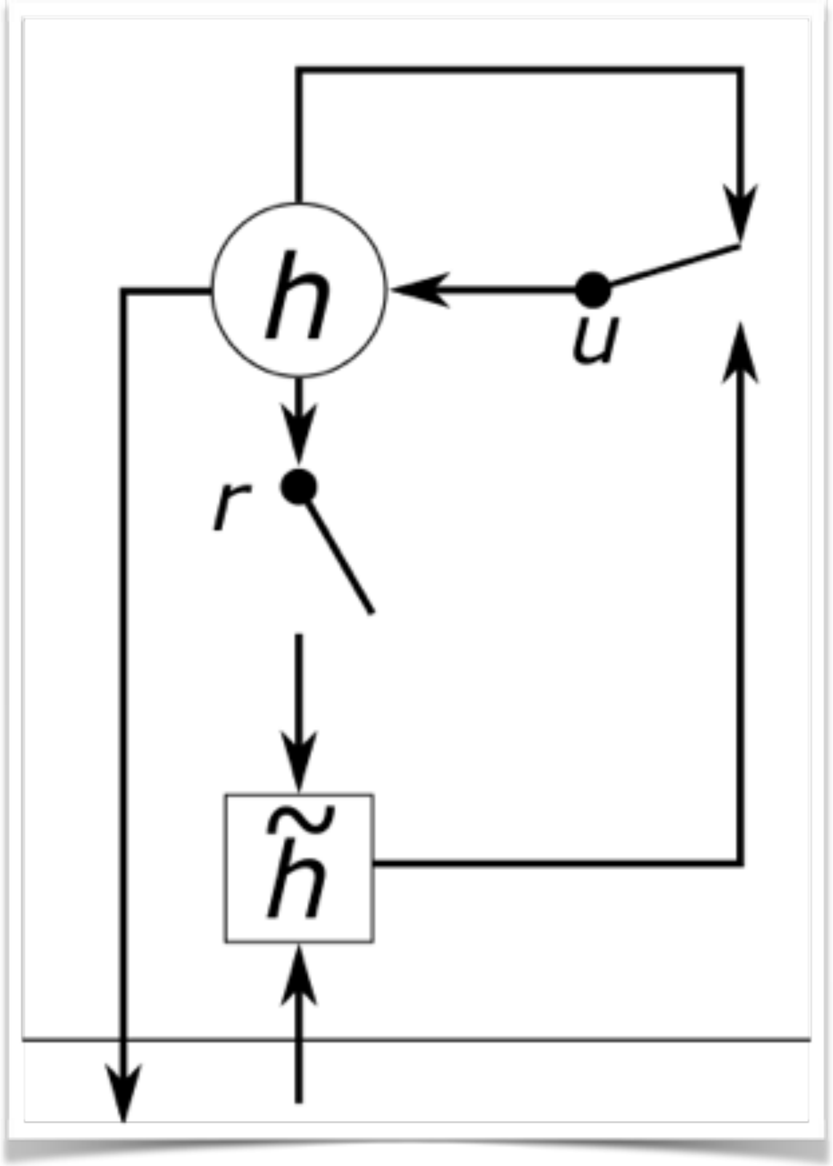
$$r_t = \sigma(W_r x_{t-1} + U_r h_{t-1} + b_r)$$

- *Candidate* hidden state

$$\tilde{h}_t = \tanh(W x_{t-1} + U(r_t \odot h_{t-1}) + b)$$

- *Adaptive* Leaky Integration

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



Cho et al. (2014)

GATED RECURRENT UNITS

Topics: Long Short-Term Memory (LSTM)

- *Input*, *Forget* and *Output* gates

$$i_t = \sigma(W_i x_{t-1} + U_i h_{t-1} + b_i)$$

$$f_t = \sigma(W_f x_{t-1} + U_f h_{t-1} + b_f)$$

$$o_t = \sigma(W_o x_{t-1} + U_o h_{t-1} + b_o)$$

- *Candidate memory* cell state

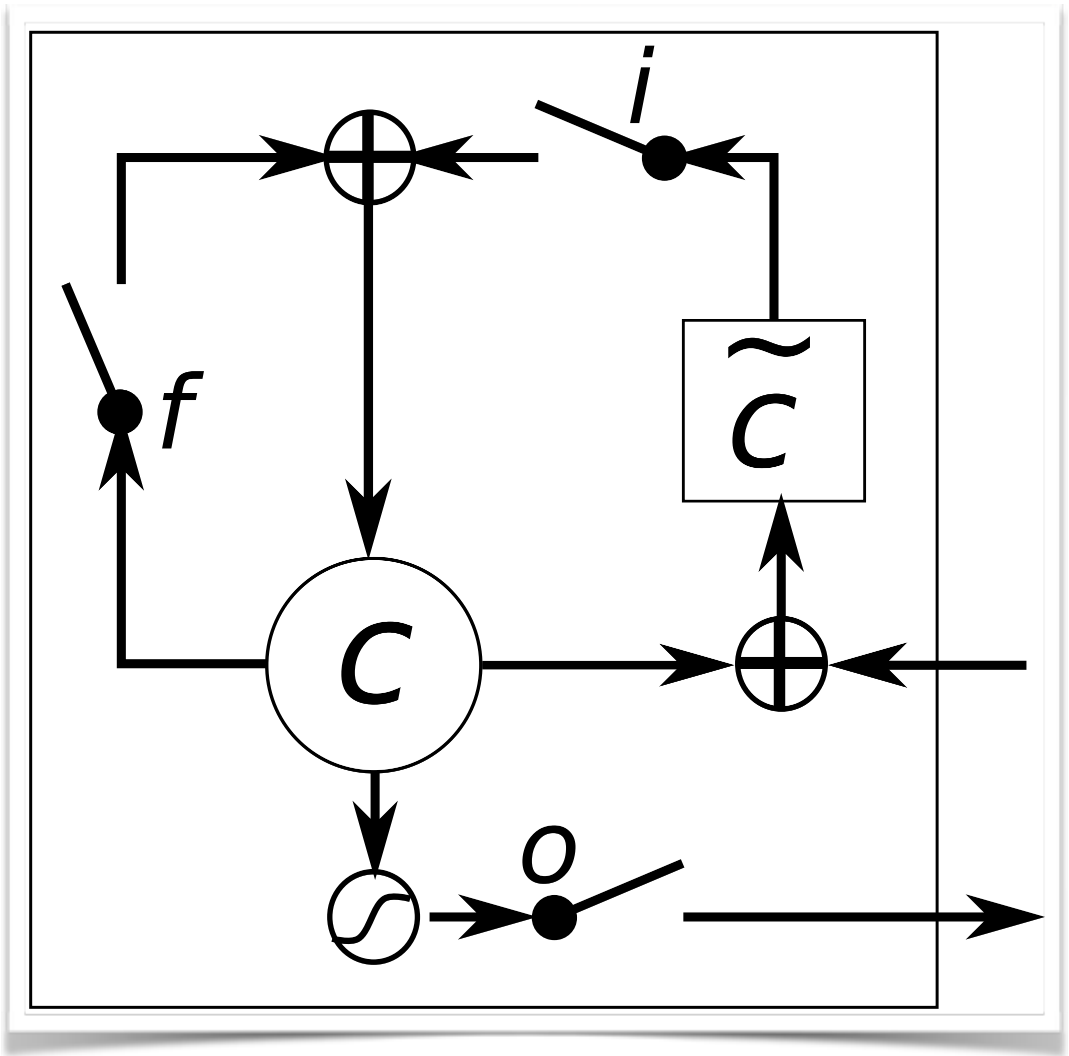
$$\tilde{c}_t = \tanh(W x_{t-1} + U h_{t-1} + b)$$

- *Adaptive* Leaky Integration

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

- Output

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



Hochreiter&Schmidhuber (1999), Gers et al. (2001)

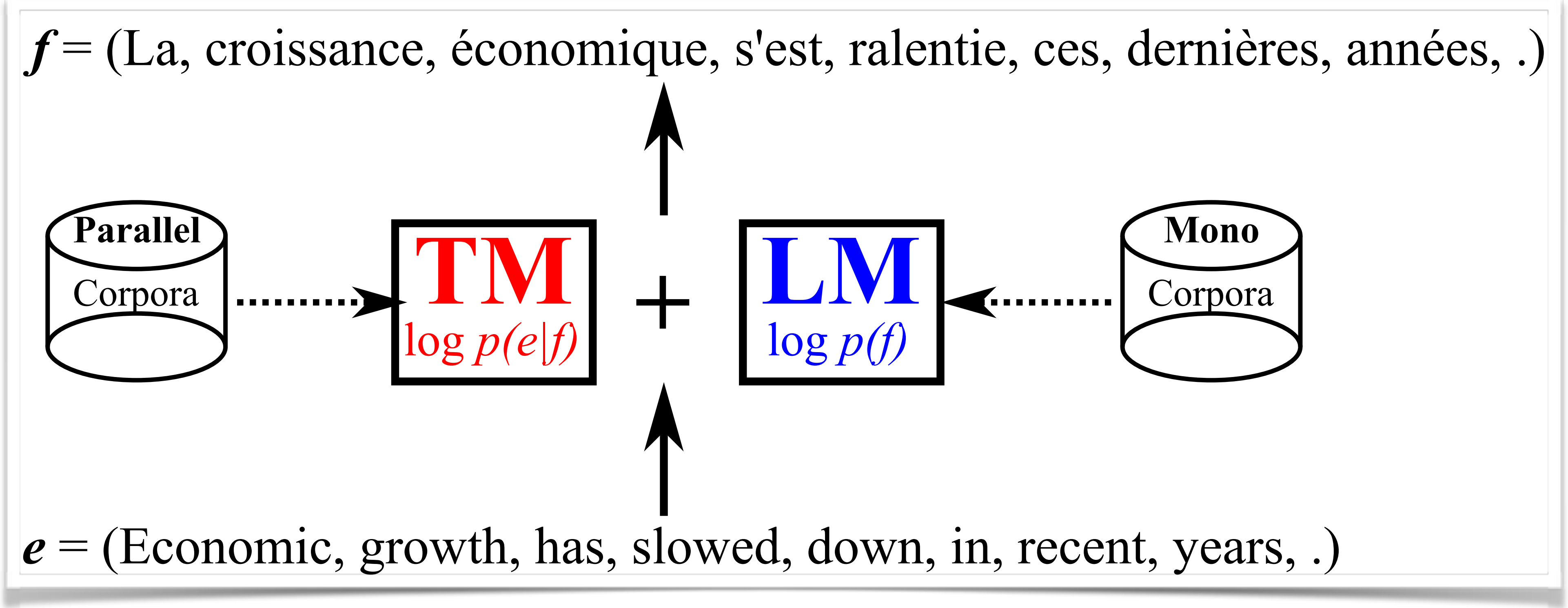
Q&A

Machine Translation

NEURAL MACHINE TRANSLATION

Topics: Statistical Machine Translation

- $\log p(f|e) = \log p(e|f) + \log p(f)$
  - Translation model: $\log p(e|f)$
    - Fit it with parallel corpora
  - Language model: $\log p(f)$
    - Fit it with monolingual corpora



- The whole task $\log p(f|e)$ is ***conditional* language modelling**.

NEURAL MACHINE TRANSLATION

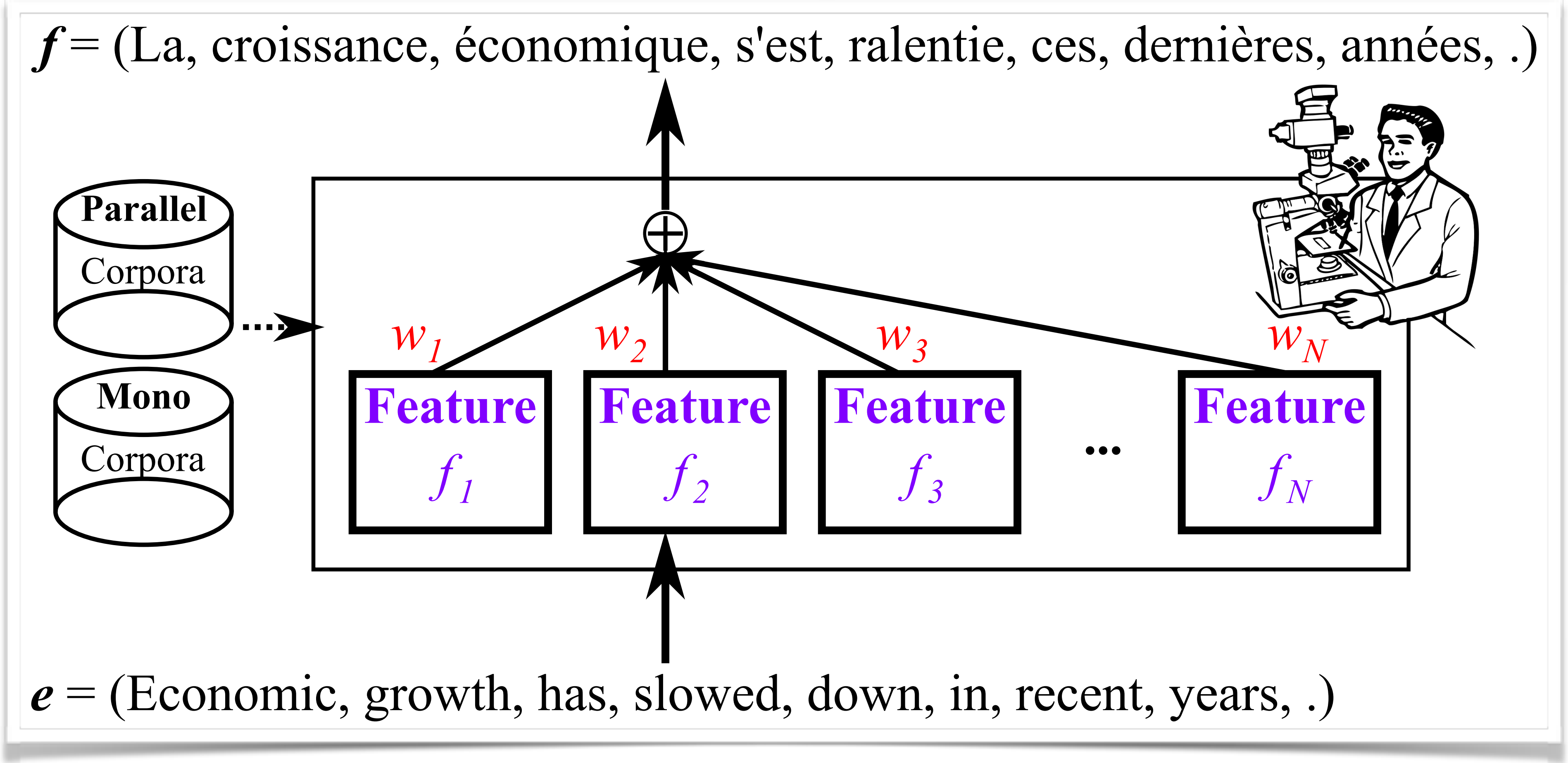
Topics: Statistical Machine Translation - In Reality

- $\log p(f|e) \approx \sum_{n=1}^{N} f_n(e,f) + C$
  - Log-linear model
  - Feature function $f_n(e,f)$
- Steps:

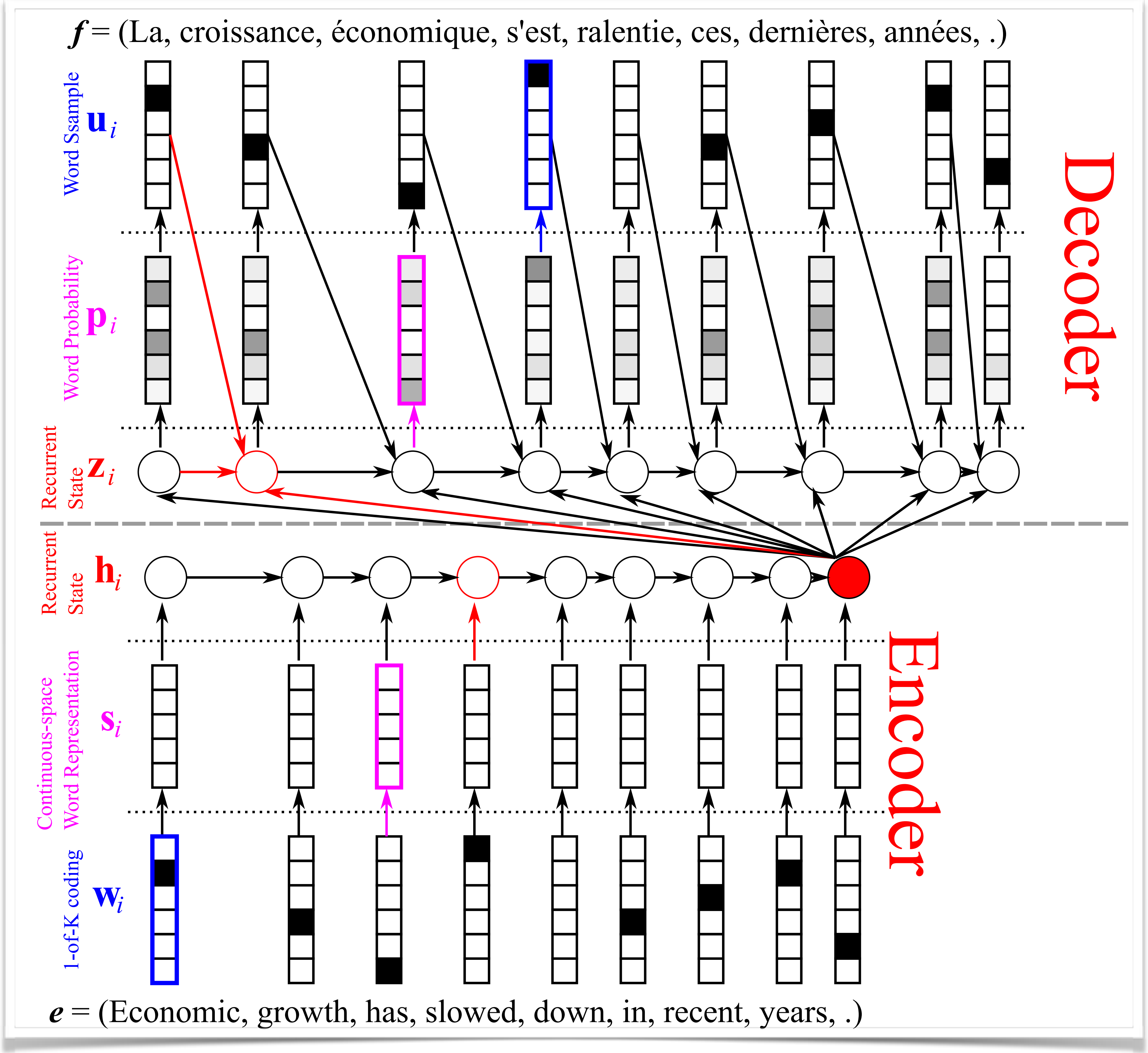
  (1) Experts engineer *useful* features

  (2) Use a simple log-linear model

  (3) Use a strong, external language model



NEURAL MACHINE TRANSLATION



(Chrisman, 1991;
Forcada&Ñeco, 1997;
Castaño&Casacuberta, 1997;
Kalchbrenner&Blunsom, 2013;
Sutskever et al., 2014;
Cho et al., 2014)

NEURAL MACHINE TRANSLATION

Topics: Sequence-to-Sequence Learning — Encoder

- Encoder

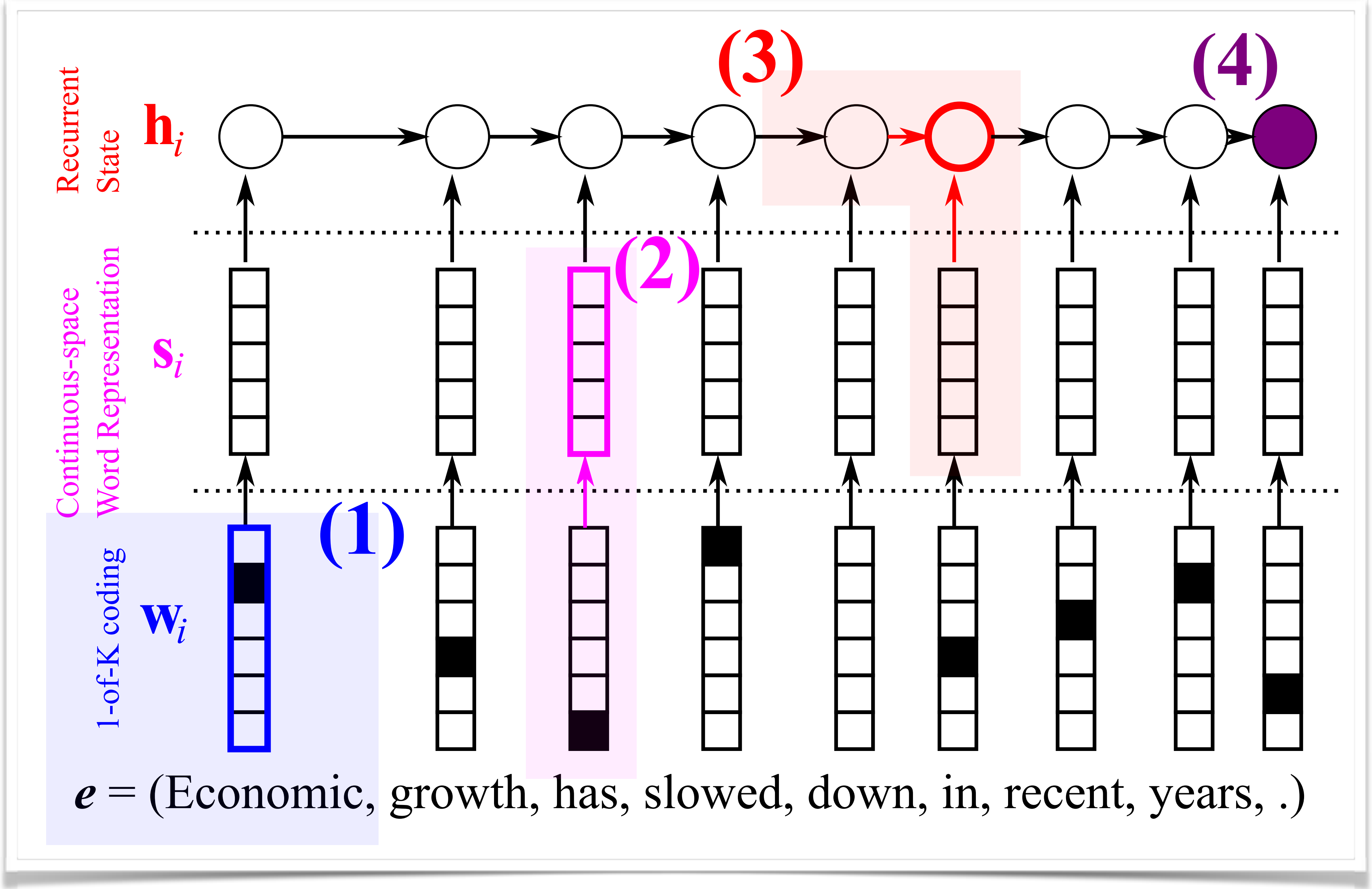
  (1) 1-of-K coding of *source* words

  (2) Continuous-space representation

  $$s_{t'} = W^\top x_{t'}, \text{ where } W \in \mathbb{R}^{|V| \times d}$$

  (3) Recursively read words

  $$h_t = f(h_{t-1}, s_t), \text{ for } t = 1, \ldots, T$$



NEURAL MACHINE TRANSLATION

Topics: Sequence-to-Sequence Learning — Decoder

- Decoder

  (1)Recursively update the memory

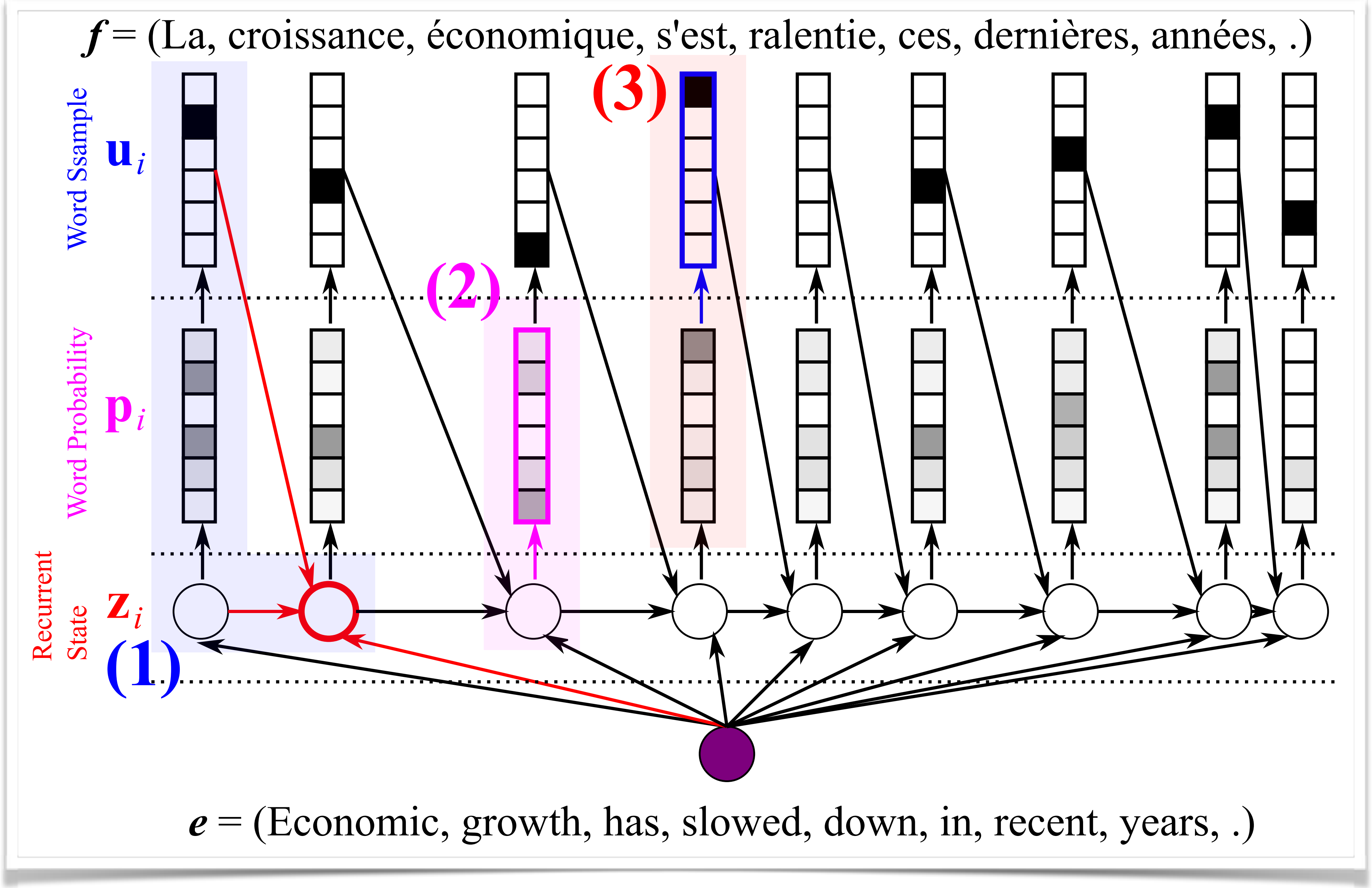
  $$z_{t'} = f(z_{t'-1}, u_{t'-1}, h_T)$$

  (2)Compute the next word prob.

  $$p(u_{t'}|u_{<t'}) \propto \exp(R_{u_{t'}}^\top z_{t'} + b_{u_{t'}})$$

  (3)Sample a next word

  - *Beam search is a good idea*

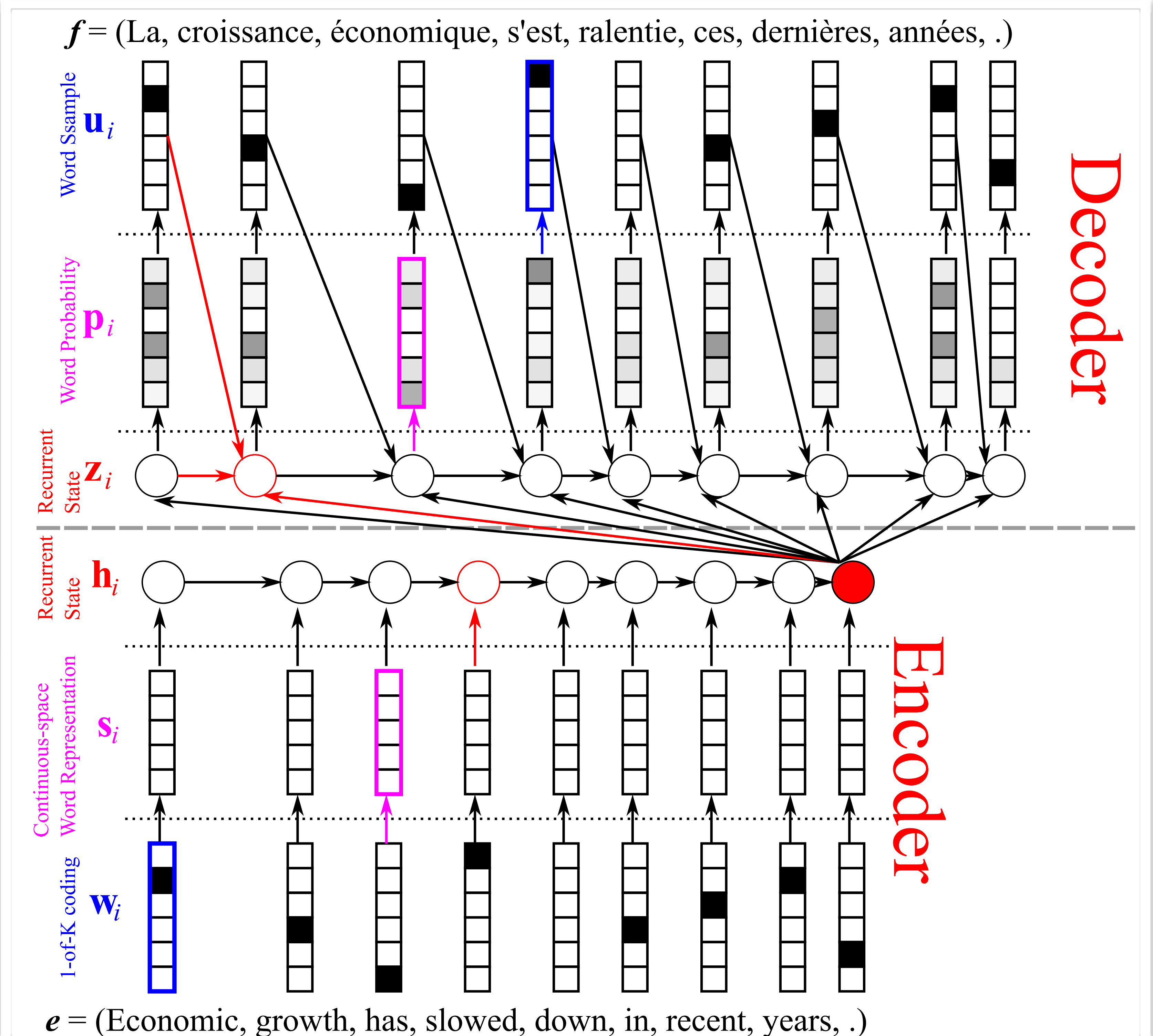


NEURAL MACHINE TRANSLATION

Topics: Sequence-to-Sequence Learning — Issue

- This is quite an unrealistic model.
- *Why?*

"You can't cram the meaning of a whole %&!$# sentence into a single $&!#* vector!" Ray Mooney



NEURAL MACHINE TRANSLATION

Topics: Attention-based Model

- Encoder: Bidirectional RNN
  - A set of *annotation* vectors

    $\{h_1, h_2, \ldots, h_T\}$
- Attention-based Decoder

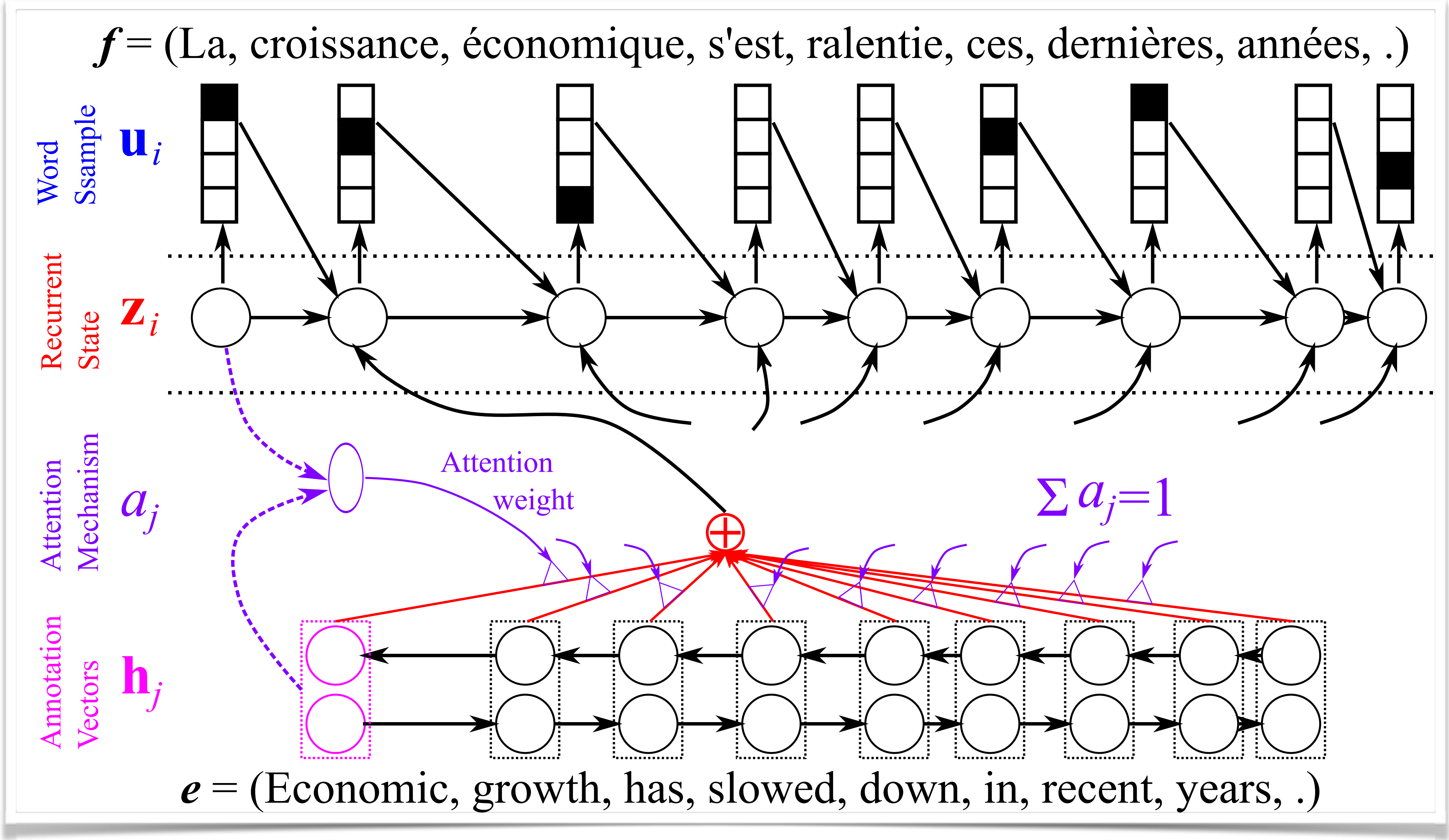
  (1)Compute attention weights

  $\alpha_{t',t} \propto \exp(e(z_{t'-1}, u_{t'-1}, h_t))$

  (2)Weighted-sum of the annotation vectors

  $c_{t'} = \sum_{t=1}^{T} \alpha_{t',t} h_t$

  (3)Use $c_{t'}$ instead of h_T



NEURAL MACHINE TRANSLATION

Topics: Attention-based Model

- Encoder: Bidirectional RNN
  - A set of *annotation* vectors

    $\{h_1, h_2, \ldots, h_T\}$
- Attention-based Decoder

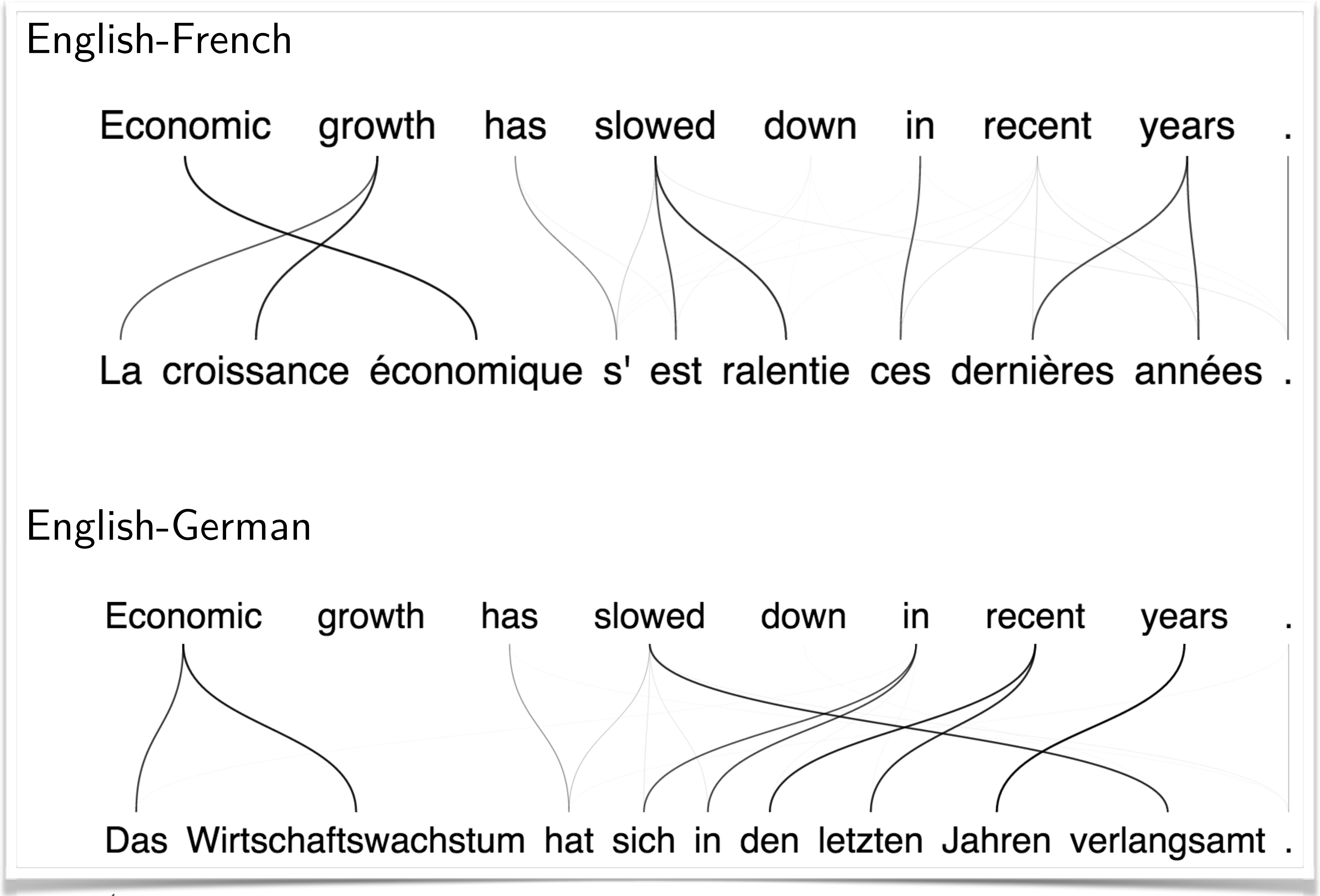
  (1)Compute attention weights

  $\alpha_{t',t} \propto \exp(e(z_{t'-1}, u_{t'-1}, h_t))$

  (2)Weighted-sum of the annotation vectors

  $c_{t'} = \sum_{t=1}^{T} \alpha_{t',t} h_t$

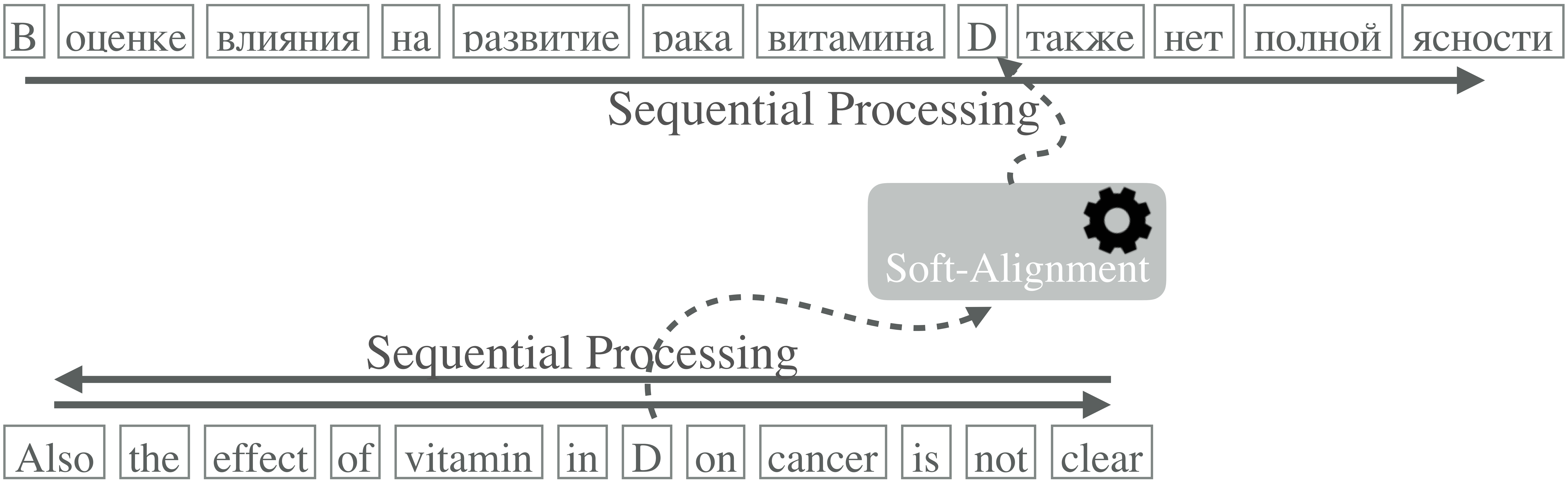
  (3)Use $c_{t'}$ instead of h_T



Deep Natural Language Processing

Deep Natural Language Processing
(1) Character-level Modelling

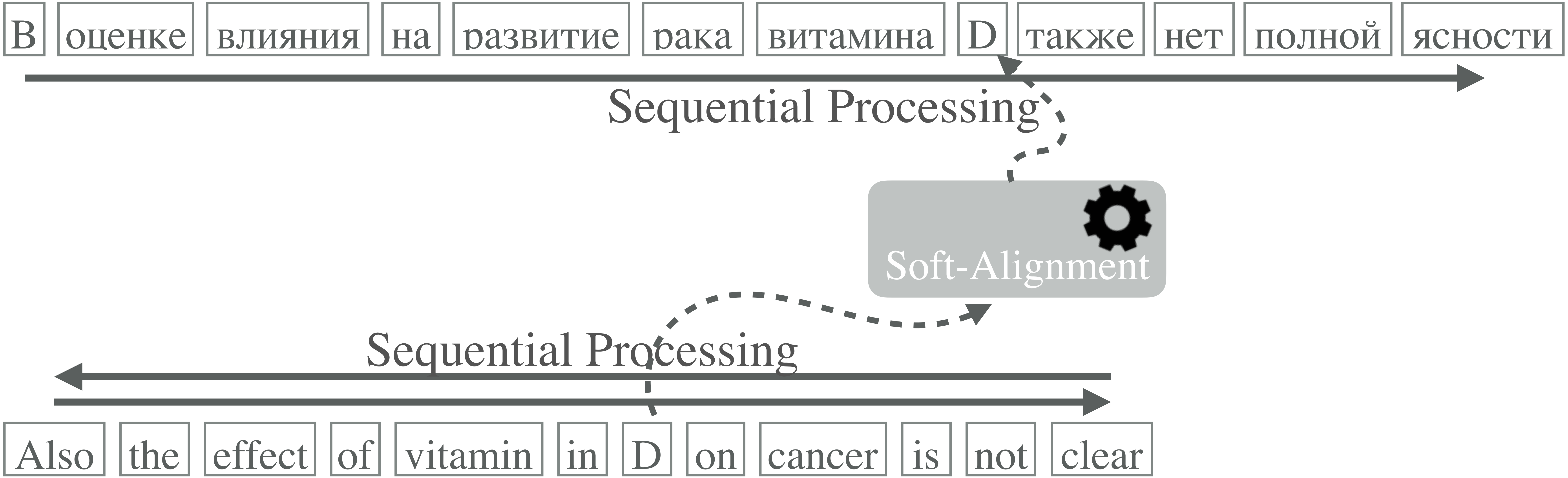
Note: Translation is horrible, not because of me, but because data is :(



(Luong et al., 2016; Sennrich et al., 2016; Chung et al., 2016; Jean et al., 2016)

Note: Translation is horrible, not because of me, but because data is :(

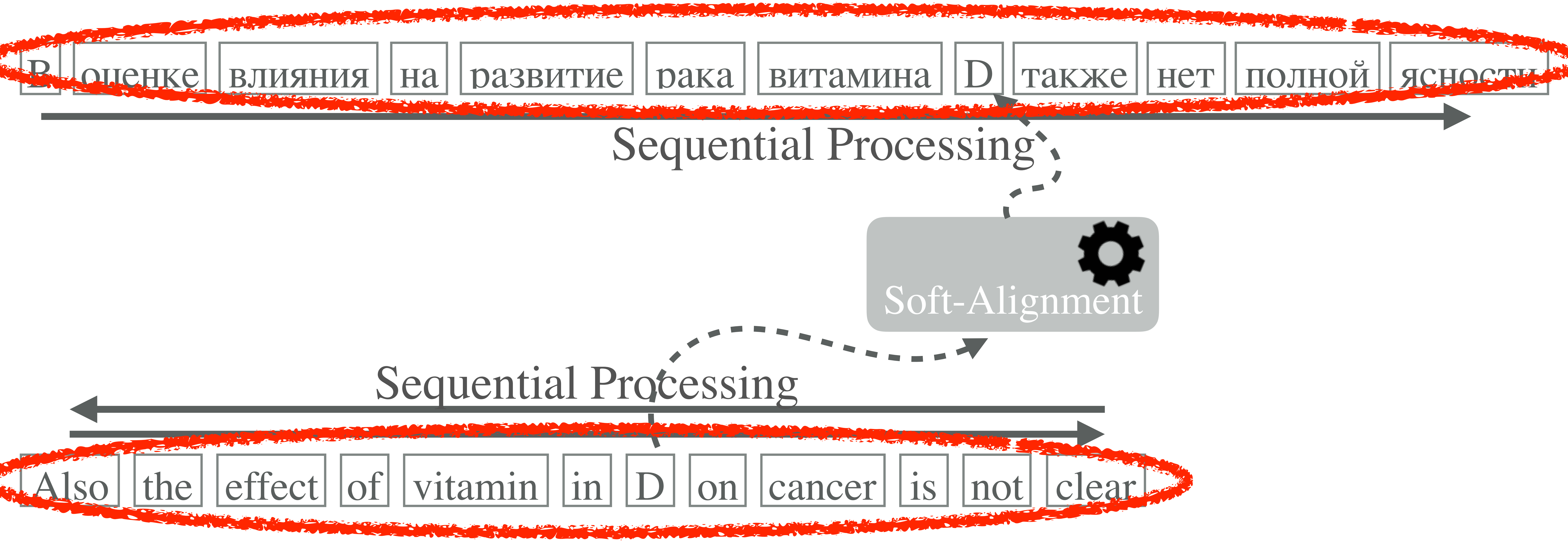
But, there are still too much explicit structures here…



(Luong et al., 2016; Sennrich et al., 2016; Chung et al., 2016; Jean et al., 2016)

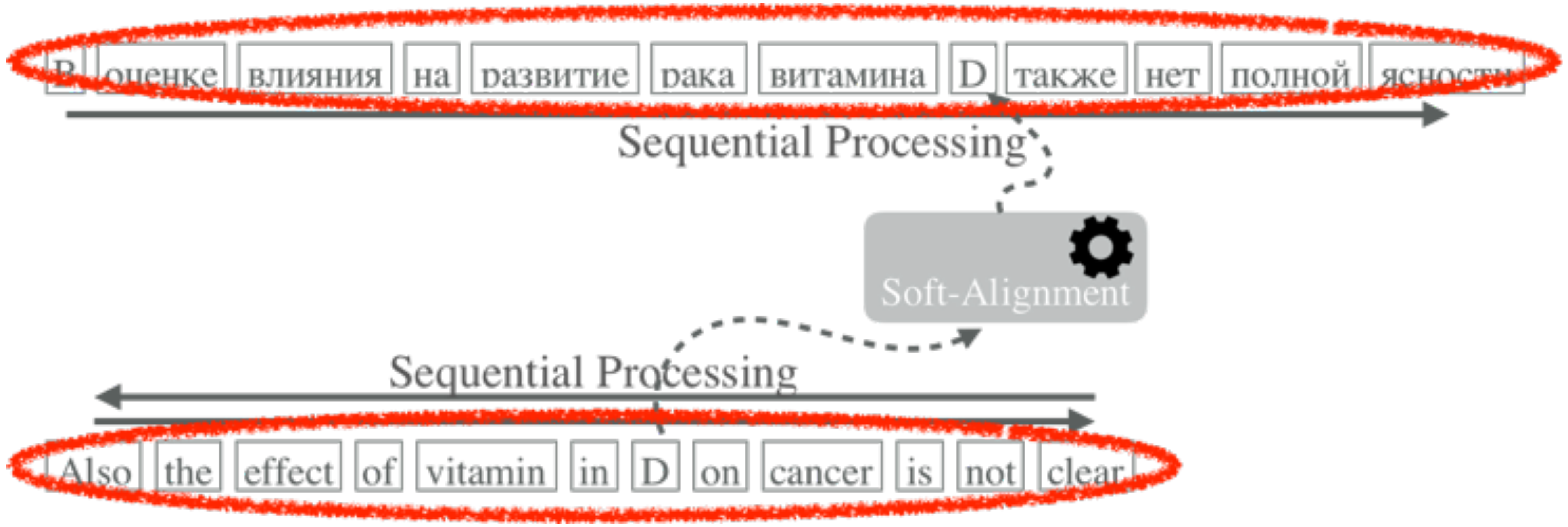
Note: Translation is horrible, not because of me, but because data is :(

Why the hell are we using a sequence of words?



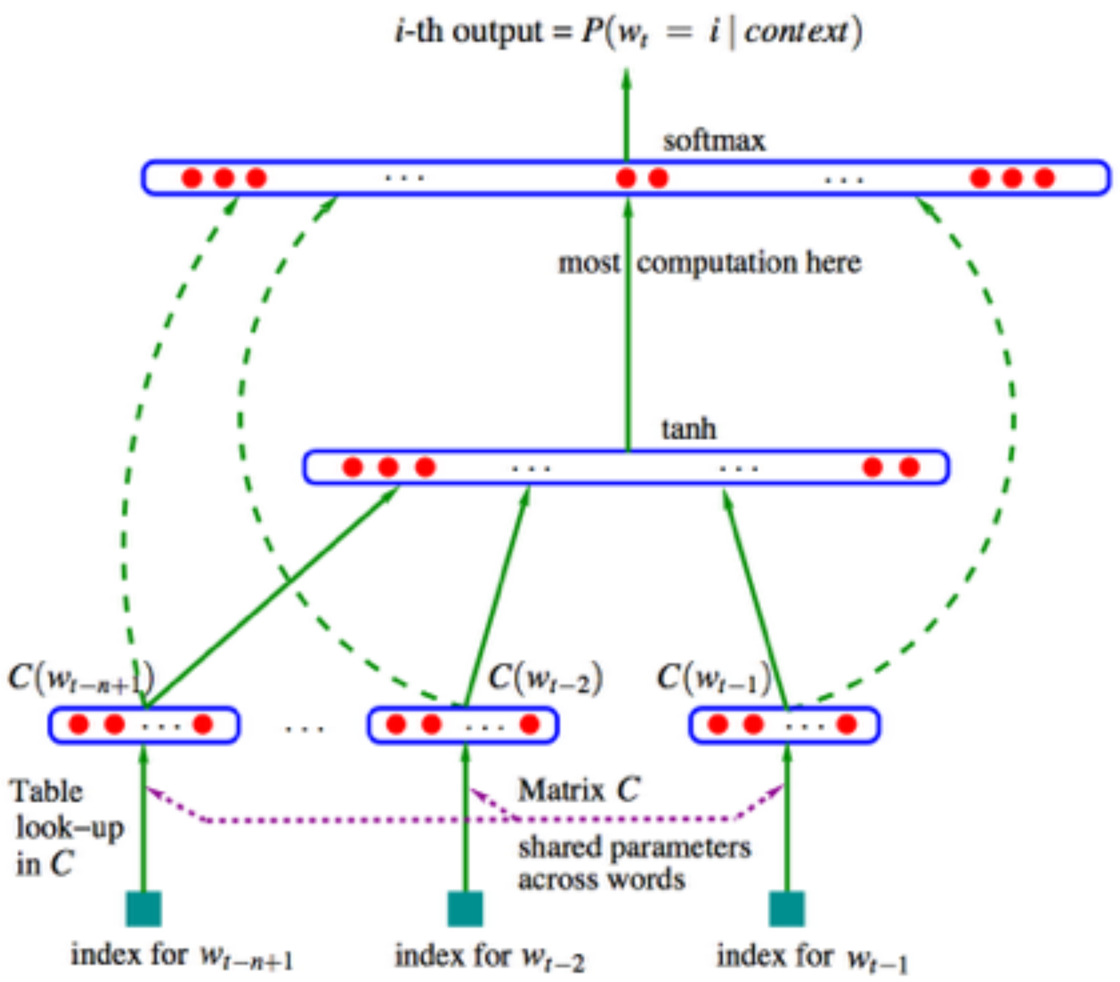
There are legitimate reasons… sorta…

1. We strongly believe that a word (lexeme) is a basic unit of meaning.
2. We have an inherent fear of data sparsity.
   - The size of state space grows exponentially w.r.t. the length.
   - A sentence is longer when counted in letters than in words.
3. We are worried that we cannot train a recurrent neural net.



But, are they really legit reasons?

1. We strongly believe that a word (lexeme) is a basic unit of meaning.
2. **We have an inherent fear of data sparsity.**
   - The size of state space grows exponentially w.r.t. the length.
   - A sentence is longer when counted in letters than in words.
3. We are worried that we cannot train a recurrent neural net.



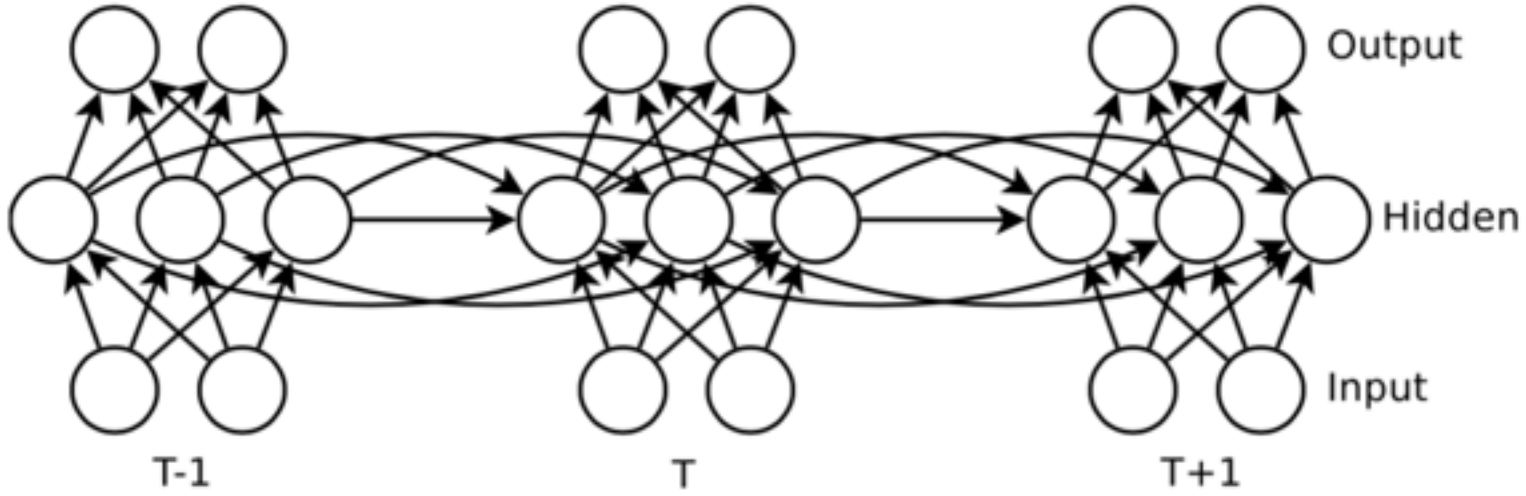
"In the proposed model, it will so generalize because "similar" words are expected to have a similar feature vector, and because the probability function is a *smooth* function of these feature values, a small change in the features will induce a small change in the probability" - Bengio et al. (2003)

(Bengio et al., 2003; Xu & Rudnicky, 2000)

But, are they really legit reasons?

1. We strongly believe that a word (lexeme) is a basic unit of meaning.
2. We have an inherent fear of data sparsity.
    - The size of state space grows exponentially w.r.t. the length.
    - **A sentence is longer when counted in letters than in words.**
3. We are worried that we cannot train a recurrent neural net.

"So, given a powerful learning system like an MRNN, the convenience of using characters may outweigh the extra work of having to learn the words. All our experiments show that an MRNN finds it very easy to learn words." - Sutskever et al. (2011)



(Sutskever et al., 2011; Mikolov, 2012; Graves, 2013)

But, are they really legit reasons?

1. We strongly believe that a word (lexeme) is a basic unit of meaning.
2. We have an inherent fear of data sparsity.
   - The size of state space grows exponentially w.r.t. the length.
   - A sentence is longer when counted in letters than in words.

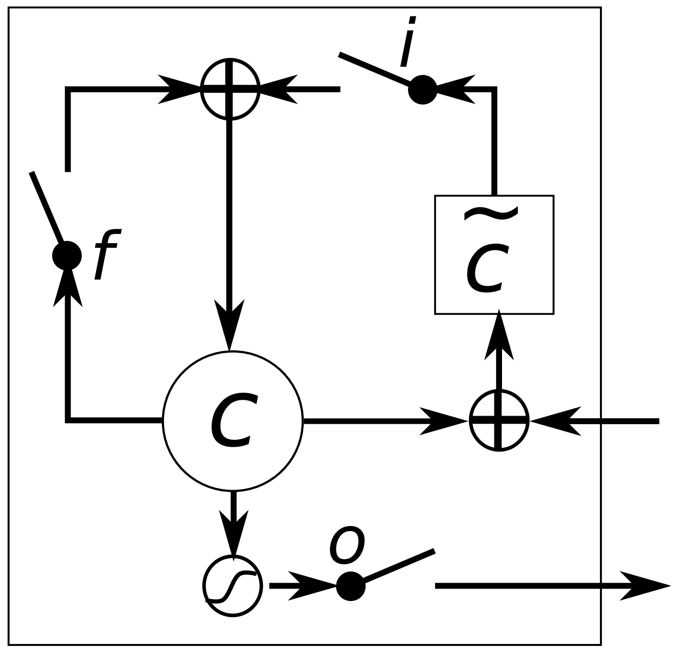
3. We are worried that we cannot train a recurrent neural net.

"Training a recurrent network to learn *long range* input/output dependencies is a hard problem."
- Bengio et al. (1994)

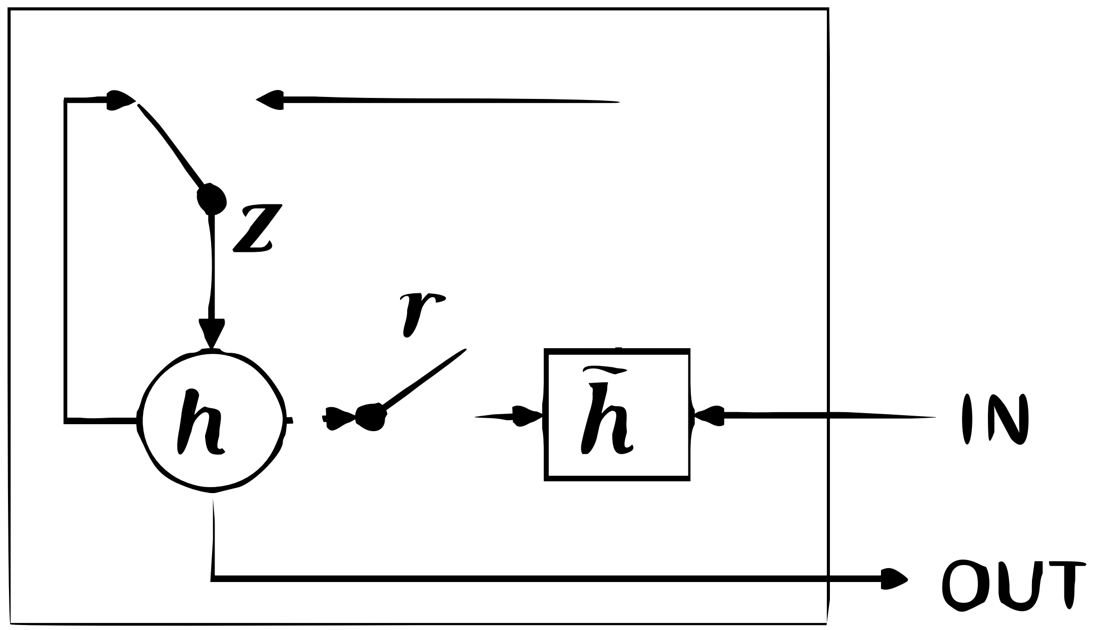
(Bengio et al., 1994; Hochreither et al., 2001)

Don't fear!!

LSTM



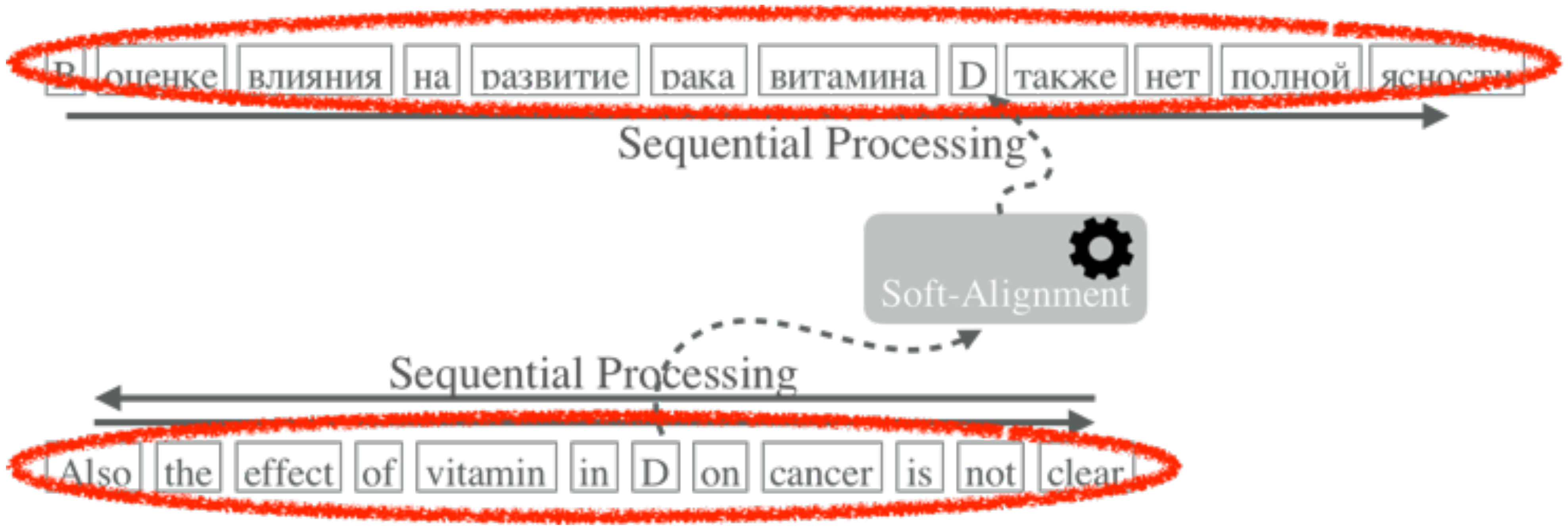
GRU



(Hochreither & Schmidhuber, 1999; Gers et al., 2001; Cho et al., 2014)

There are legitimate reasons… sorta…

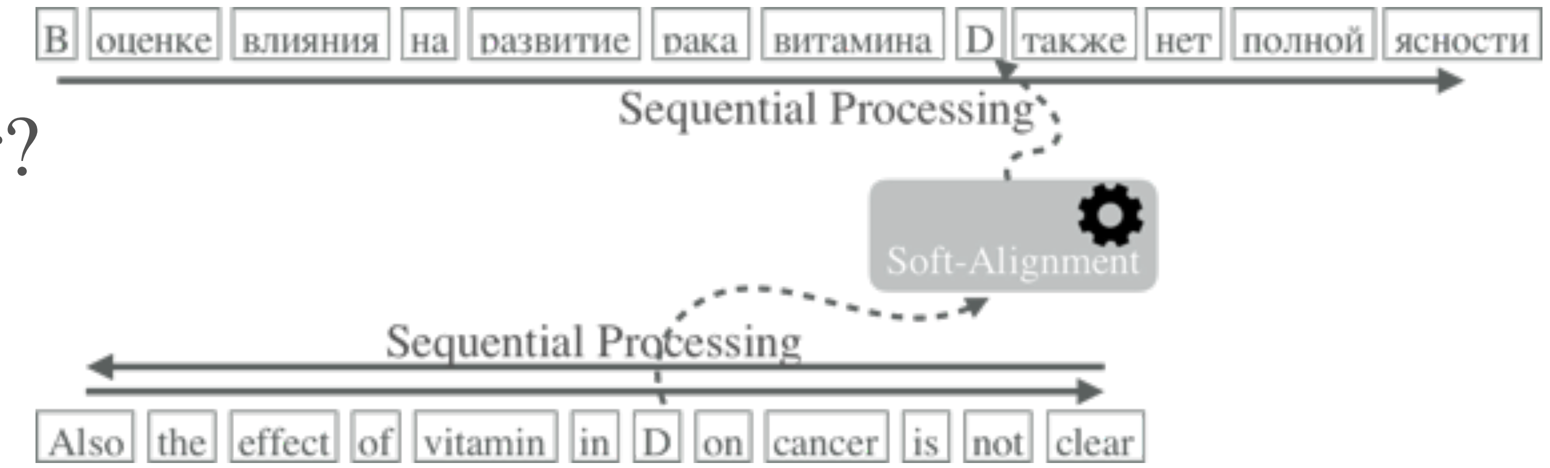
1. We strongly believe that a word (lexeme) is a basic unit of meaning.
2. ~~We have an inherent fear of data sparsity.~~
   - The size of state space grows exponentially w.r.t. the length.
   - A sentence is longer when counted in letters than in words.
3. ~~We are worried that we cannot train a recurrent neural net.~~



Note: Translation is horrible, not because of me, but because data is :(

Problems with treating each and every token separately

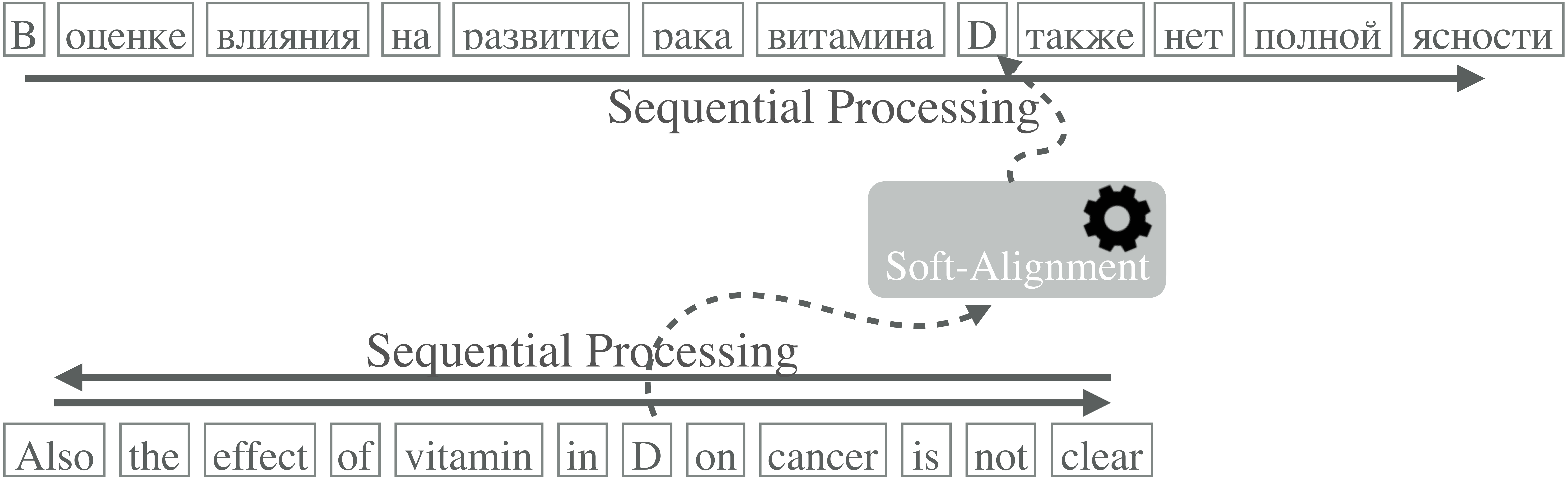
1. Inefficient handling of various morphological variants
   - Sub-optimal segmentation/tokenization
   - “run”, “runs”, “ran”, “running”: one lexeme “run”, but four independent vectors.
2. Lack of generalization to novel/rare morphological variants
   - For instance, ولمركبته in Arabic => “and to his vehicle”
3. One vector for compound words?
   - “**kolmi**/vaihe/kilo/watti/tunti/mittari” => one vector?
   - “**kolme**” => one vector?



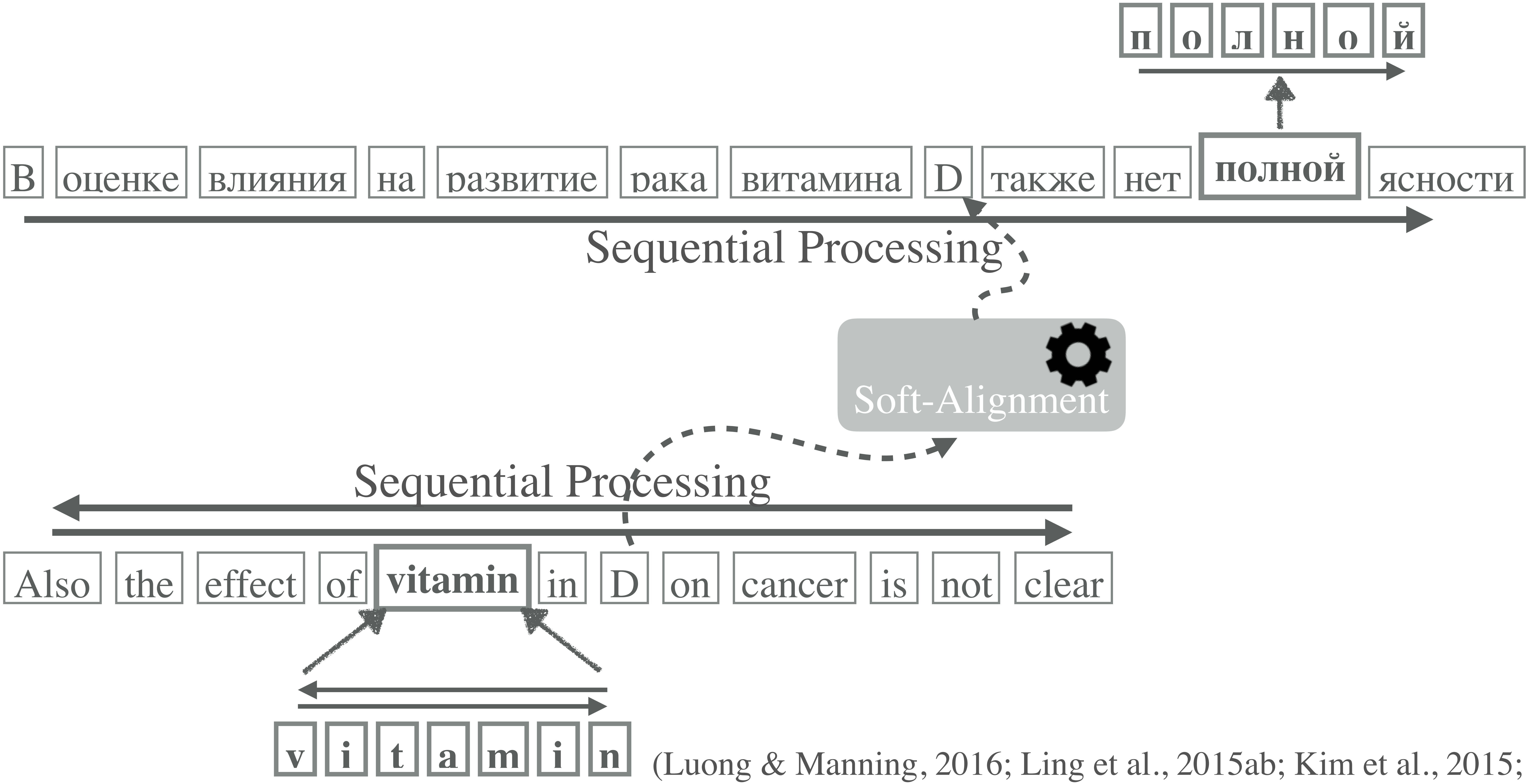
(Chung et al., 2016; Almahairi et al., 2016; Both & Blunsom, 2014)

Note: Translation is horrible, not because of me, but because data is :(

Obviously I'm not the first one to ask this question…



Note: Translation is horrible, not because of me, but because data is :(



(Luong & Manning, 2016; Ling et al., 2015ab; Kim et al., 2015; Ballesteros et al., 2015; dos Santos & Zadrozny, 2014)

Addresses

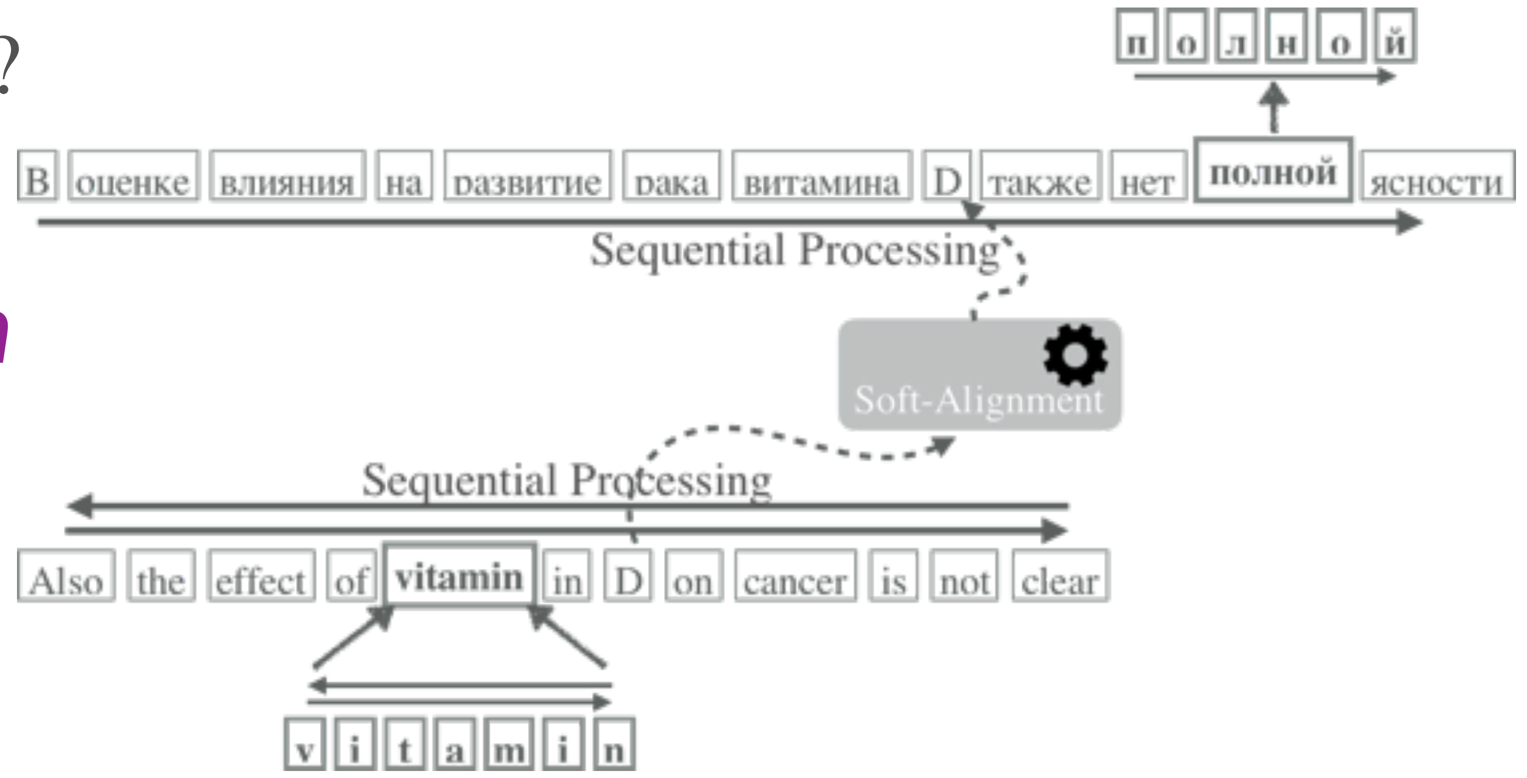
1. Inefficient handling of various morphological variants
   - Sub-optimal segmentation/tokenization
   - “run”, “runs”, “ran”, “running”: one lexeme “run”, but four independent vectors.
2. Lack of generalization to novel/rare morphological variants
   - For instance, ولمركبته in Arabic => “and to his vehicle”

Does not address

3. One vector for compound words?
   - “**kolmi**/vaihe/kilo/watti/tunti/mittari” => one vector?
   - “**kolme**” => one vector?

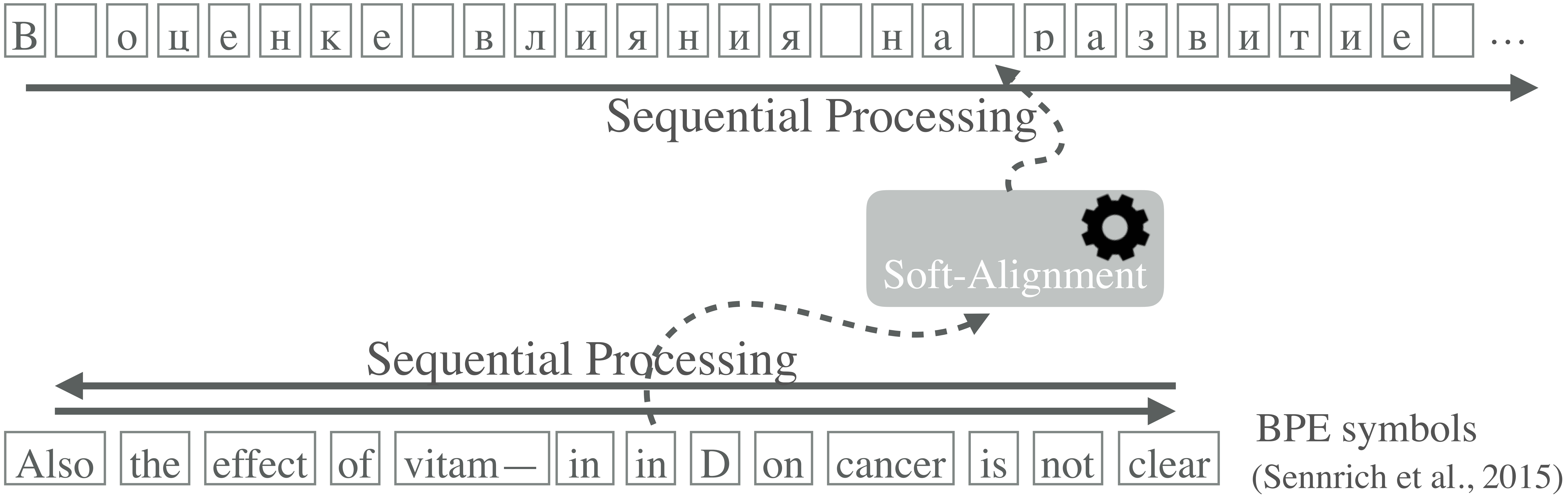
Still relies on

4. Good segmentation/tokenization



So, we decided to answer this question ourselves…

1. Source side: a sequence of BPE-based character n-grams
2. Target side: an unbroken sequence of characters

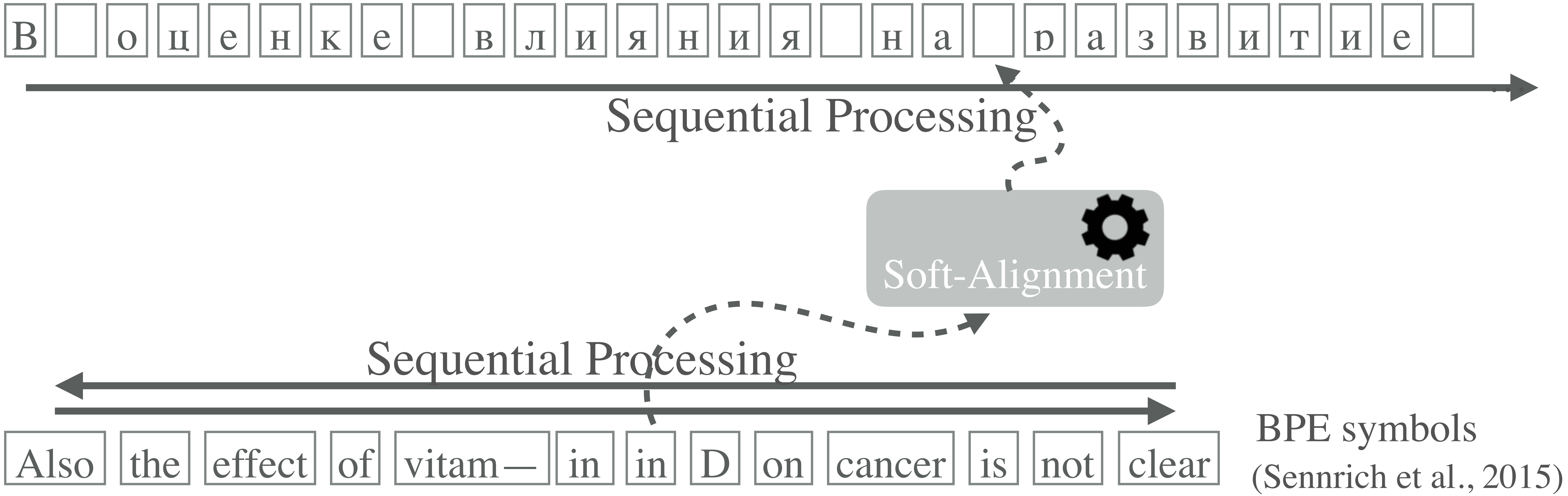


Absolutely the same model we have been using so far…

For a better recurrent decoder for character sequences, stay tuned for ACL'16. JY will tell us more about it.

So, we decided to answer this question ourselves…

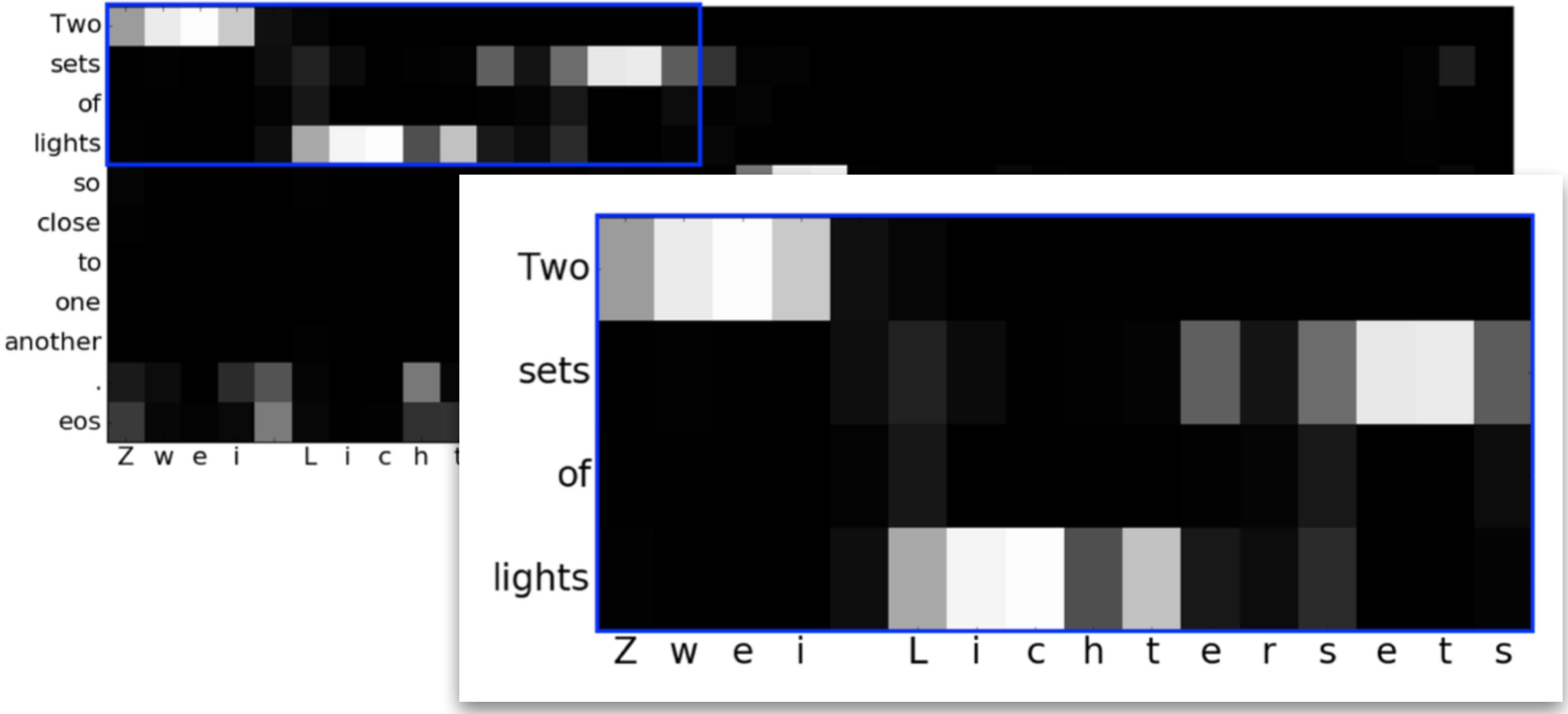
1. Large-scale experiments: *we want a convincing answer!*
2. Multiple languages: En ➞ {Cz, De, Ru, Fi}



For a better recurrent decoder for character sequences, stay tuned for ACL'16. JY will tell us more about it.

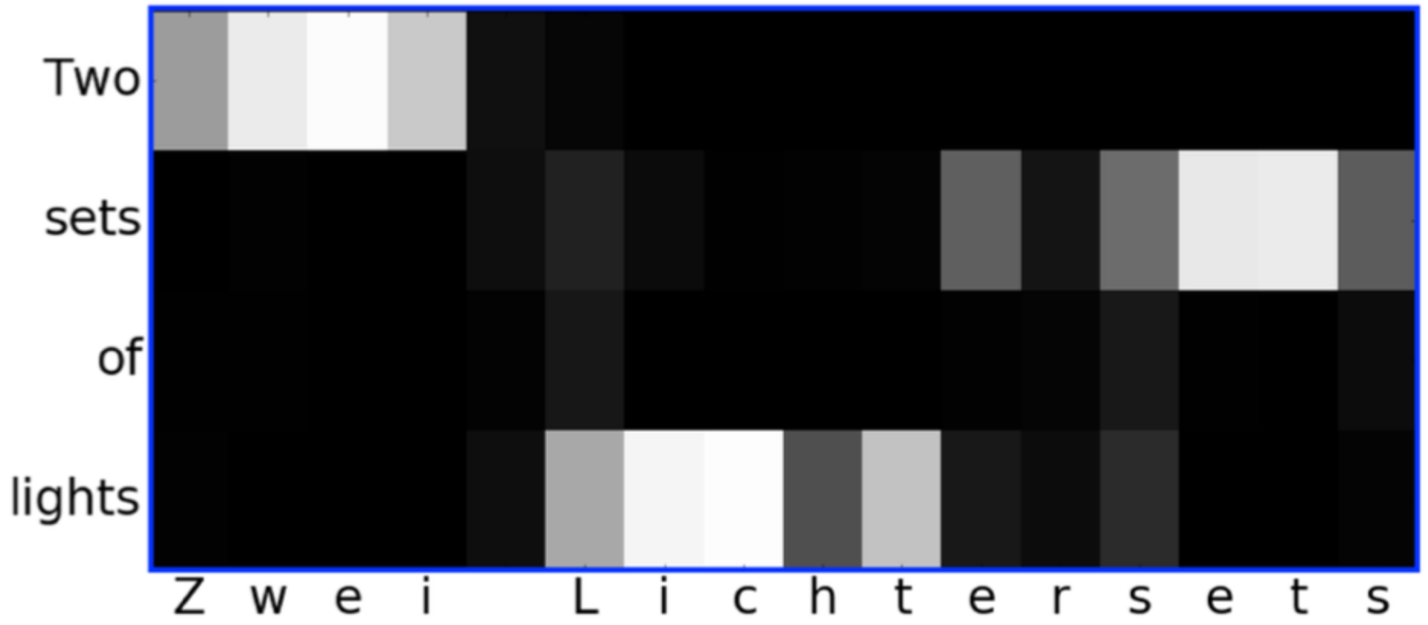
		Src	Trgt	Depth	Attention h^1	Attention h^2	Model	Development Single	Development Ens	$Test_1$ Single	$Test_1$ Ens	$Test_2$ Single	$Test_2$ Ens
En-De	(a)	BPE	BPE	1	✓		Base	20.78	–	19.98	–	21.72	–
	(b)			2	✓	✓		$21.26^{21.45}_{20.62}$	23.49	$20.47^{20.88}_{19.30}$	23.10	$22.02^{22.21}_{21.35}$	24.83
	(c)		Char	2		✓	Base	$21.57^{21.88}_{20.88}$	23.14	$\mathbf{21.33}^{21.56}_{19.82}$	$\underline{23.11}$	$\mathbf{23.45}^{23.91}_{21.72}$	25.24
	(d)			2	✓	✓		20.31	–	19.70	–	21.30	–
	(e)			2		✓	Bi-S	$21.29^{21.43}_{21.13}$	23.05	$21.25^{21.47}_{20.62}$	23.04	$23.06^{23.47}_{22.85}$	$\underline{25.44}$
	(f)			2	✓	✓		20.78	–	20.19	–	22.26	–
	(g)			2	✓			20.08	–	19.39	–	20.94	–
	State-of-the-art Non-Neural Approach*							–		20.60(1)		24.00(2)	
En-Cs	(h)	BPE	BPE	2	✓	✓	Base	$16.12^{16.96}_{15.96}$	19.21	$17.16^{17.68}_{16.38}$	20.79	$14.63^{15.09}_{14.26}$	17.61
	(i)		Char	2		✓	Base	$17.68^{17.78}_{17.39}$	19.52	$19.25^{19.55}_{18.89}$	21.95	$\mathbf{16.98}^{17.17}_{16.81}$	18.92
	(j)			2		✓	Bi-S	$17.62^{17.93}_{17.43}$	19.83	$\mathbf{19.27}^{19.53}_{19.15}$	$\underline{22.15}$	$16.86^{17.10}_{16.68}$	$\underline{18.93}$
	State-of-the-art Non-Neural Approach*							–		21.00(3)		18.20(4)	
En-Ru	(k)	BPE	BPE	2	✓	✓	Base	$18.56^{18.70}_{18.26}$	21.17	$25.30^{25.40}_{24.95}$	29.26	$19.72^{20.29}_{19.02}$	22.96
	(l)		Char	2		✓	Base	$18.56^{18.87}_{18.39}$	20.53	$\mathbf{26.00}^{26.07}_{25.04}$	$\underline{29.37}$	$\mathbf{21.10}^{21.24}_{20.14}$	23.51
	(m)			2		✓	Bi-S	$18.30^{18.54}_{17.88}$	20.53	$25.59^{25.76}_{24.57}$	29.26	$20.73^{21.02}_{19.97}$	$\underline{23.75}$
	State-of-the-art Non-Neural Approach*							–		28.70(5)		24.30(6)	
En-Fi	(n)	BPE	BPE	2	✓	✓	Base	$9.61^{10.02}_{9.24}$	11.92	–	–	$8.97^{9.17}_{8.88}$	11.73
	(o)		Char	2		✓	Base	$11.19^{11.55}_{11.09}$	13.72	–	–	$\mathbf{10.93}^{11.56}_{10.11}$	$\underline{13.48}$
	(p)			2		✓	Bi-S	$10.73^{11.04}_{10.40}$	13.39	–	–	$10.24^{10.63}_{9.71}$	13.32
	State-of-the-art Non-Neural Approach*							–		–		12.70(7)	

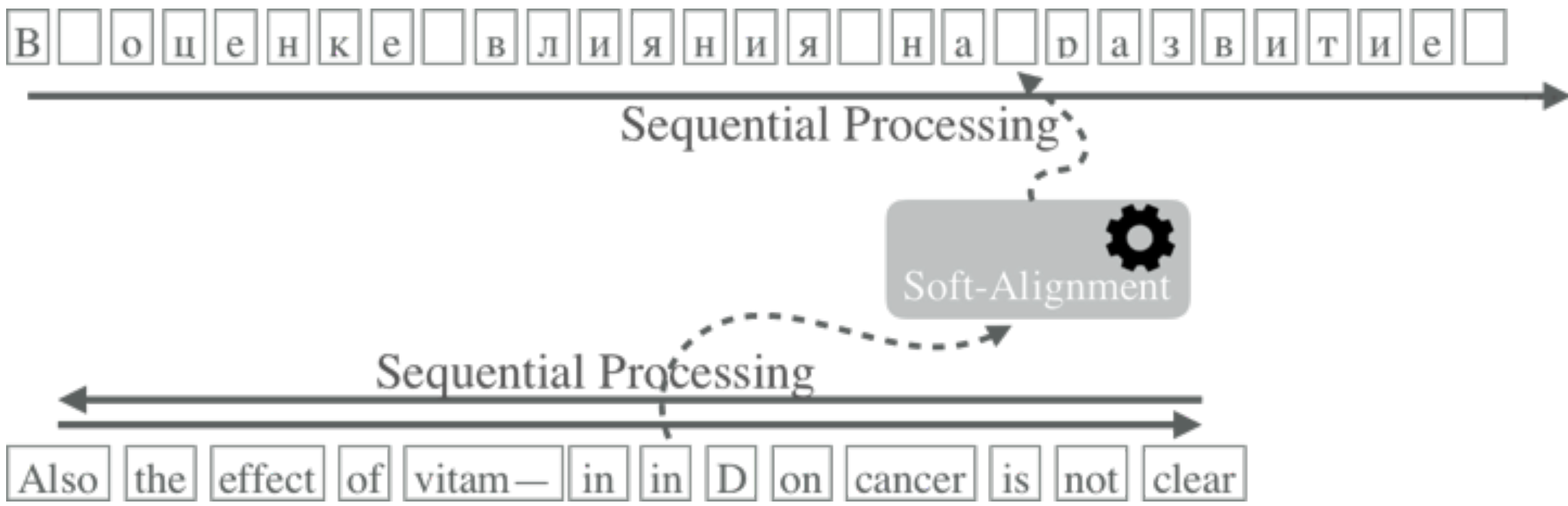
The decoder implicitly learned word-like units automatically.



What have we learned?

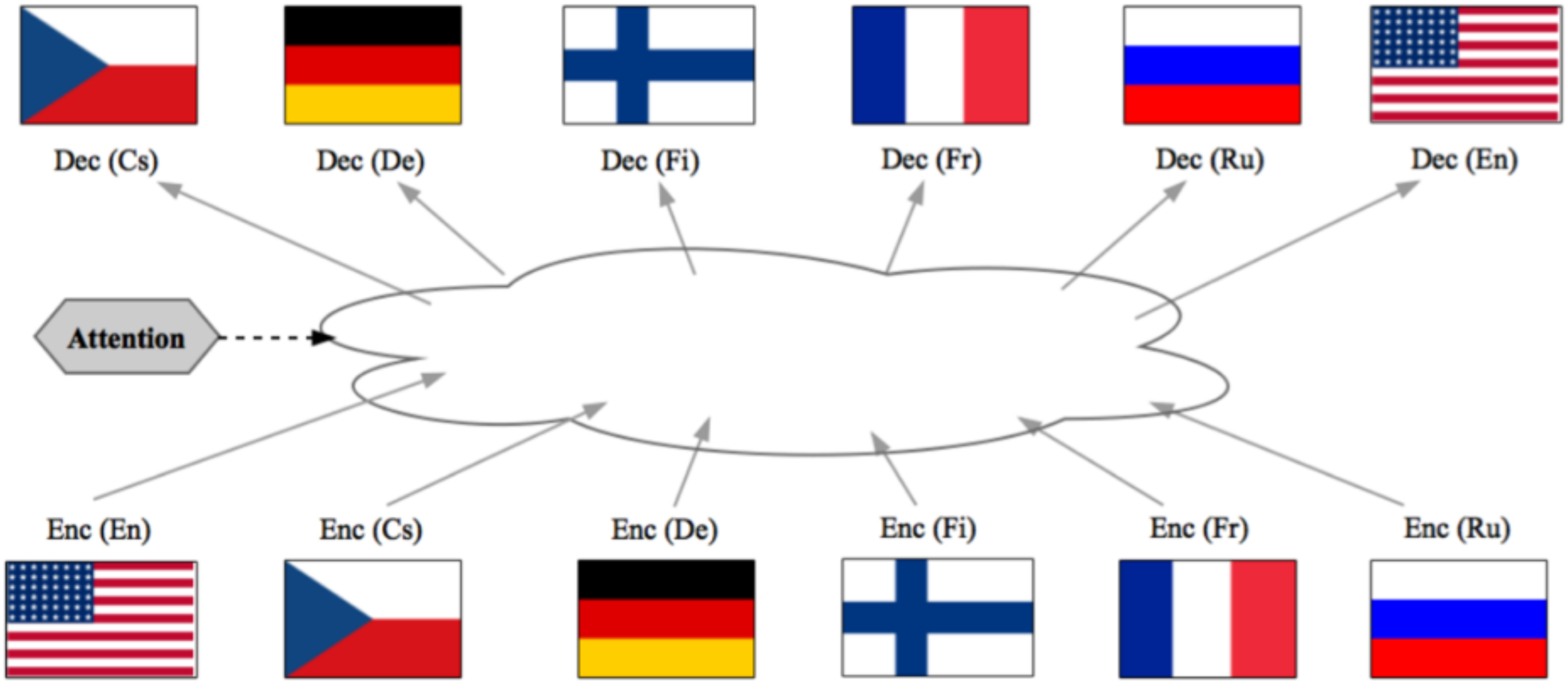
1. Neural MT works with character sequences.
    - At least on the target side (though, it works also on the source side ;))
2. A recurrent network implicitly segments a character sequence automatically.
3. *We should've asked this question at the very beginning..*





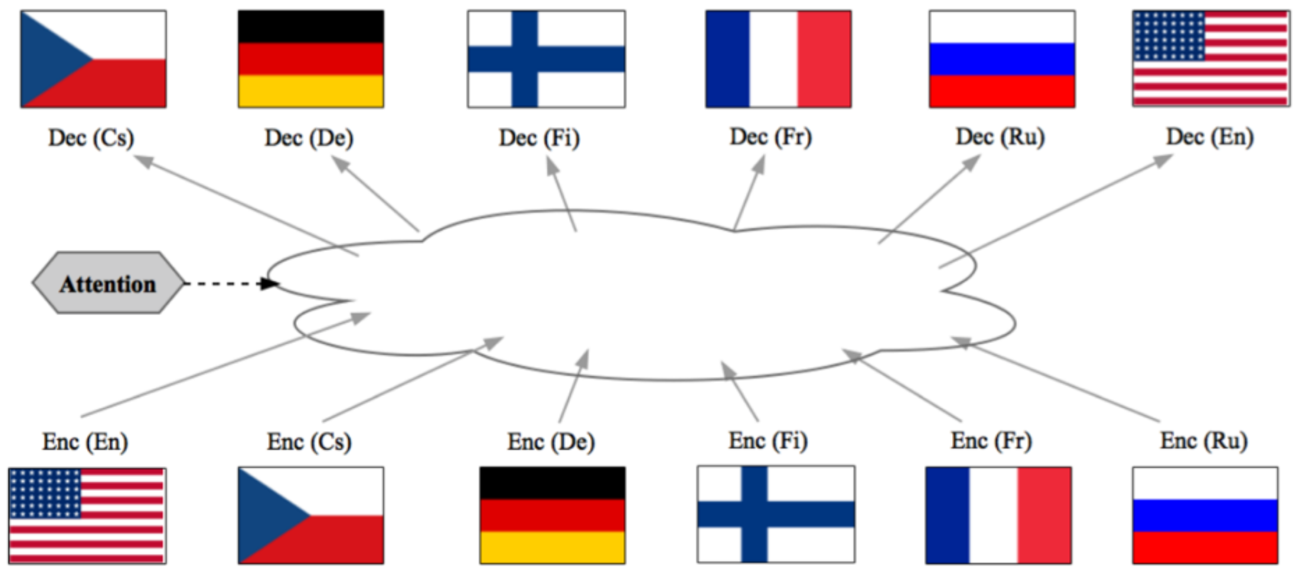
Deep Natural Language Processing
(2) Multilingual Modelling

Multilingual Translation

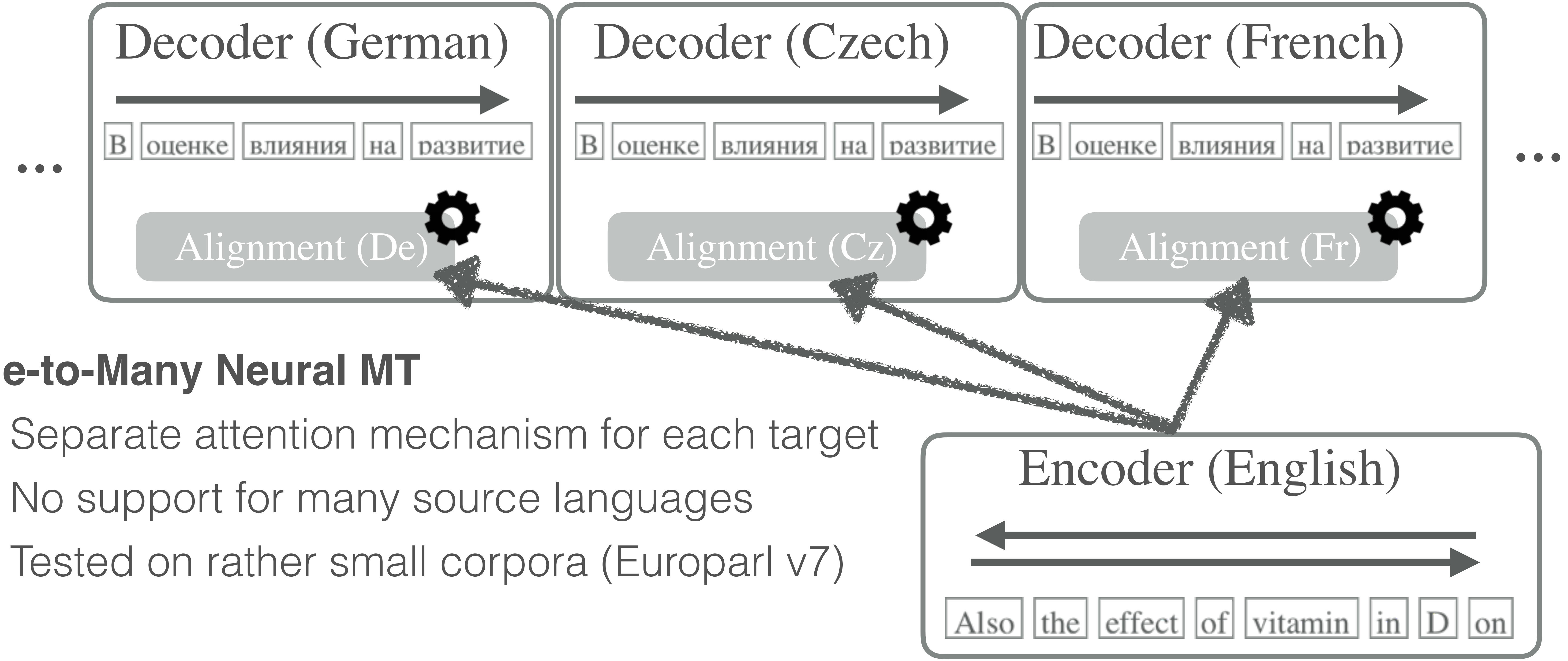


Multilingual Translation: Benefits

1. Positive language transfer across many language pairs/directions
   - Solution to low/zero-resource machine translation
2. # of parameters grows linearly w.r.t. the # of languages
   - as opposed to the quadratic explosion when training many single-pair models.
3. Multi-source translation without requiring any multi-way parallel text
   - inspired by but contrary to Zoph & Knight (2016)
4. **Super fun and cool!**
   - **Most important reason..**



DONG ET AL. (ACL 2015)



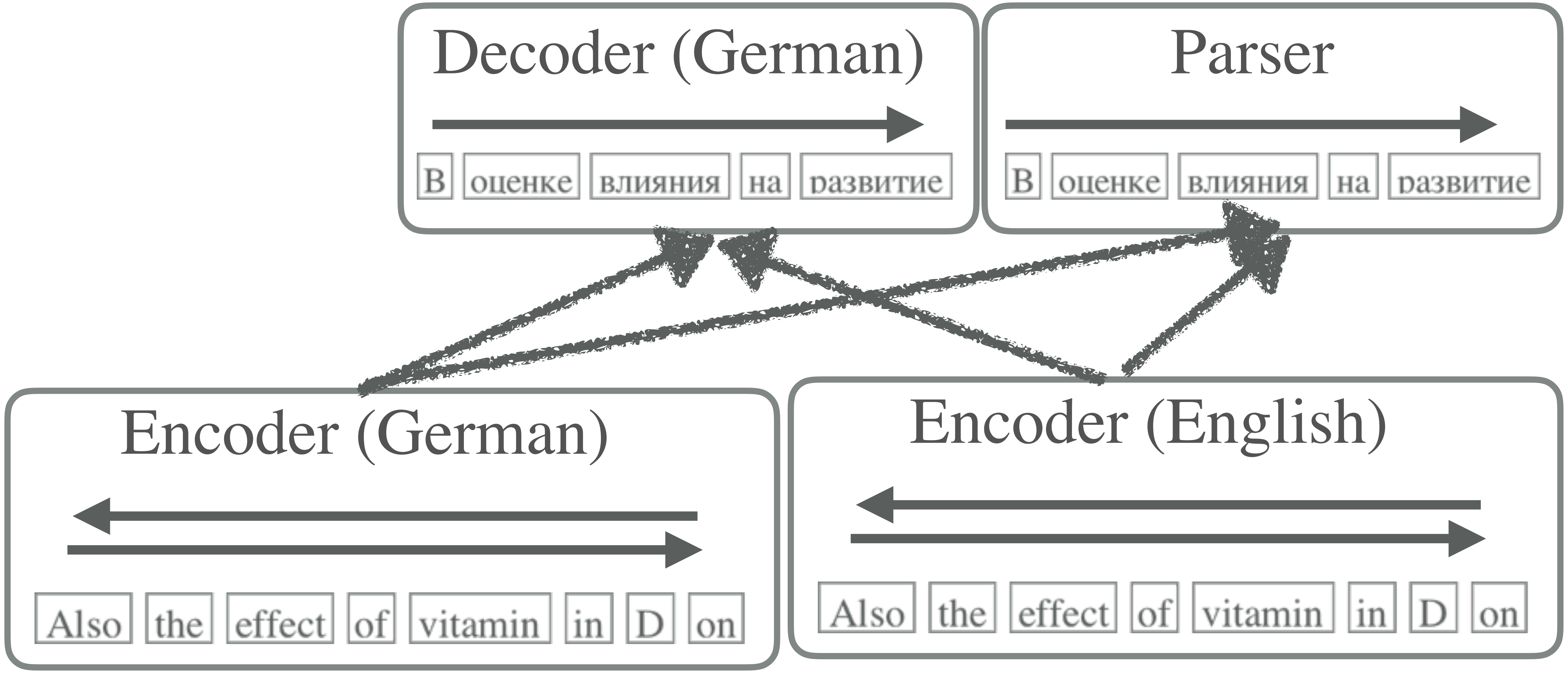
One-to-Many Neural MT

1. Separate attention mechanism for each target
2. No support for many source languages
3. Tested on rather small corpora (Europarl v7)

LUONG ET AL. (ICLR, NOV 2015)

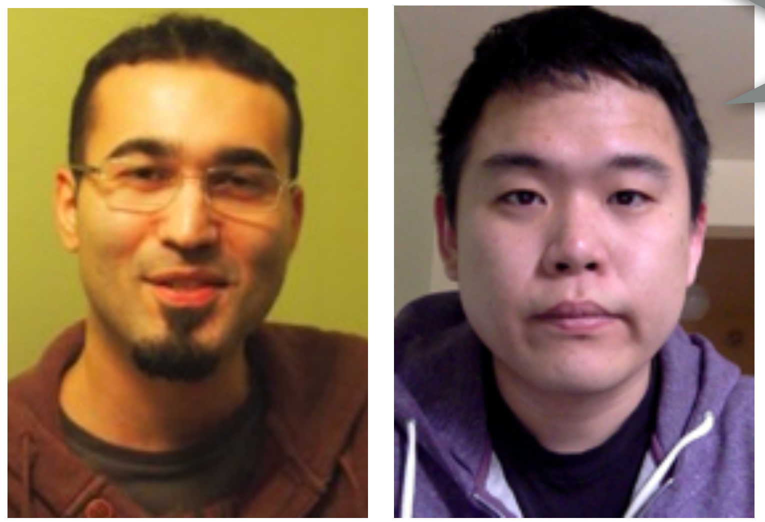
Many-to-Many Sequence-to-Sequence Learning

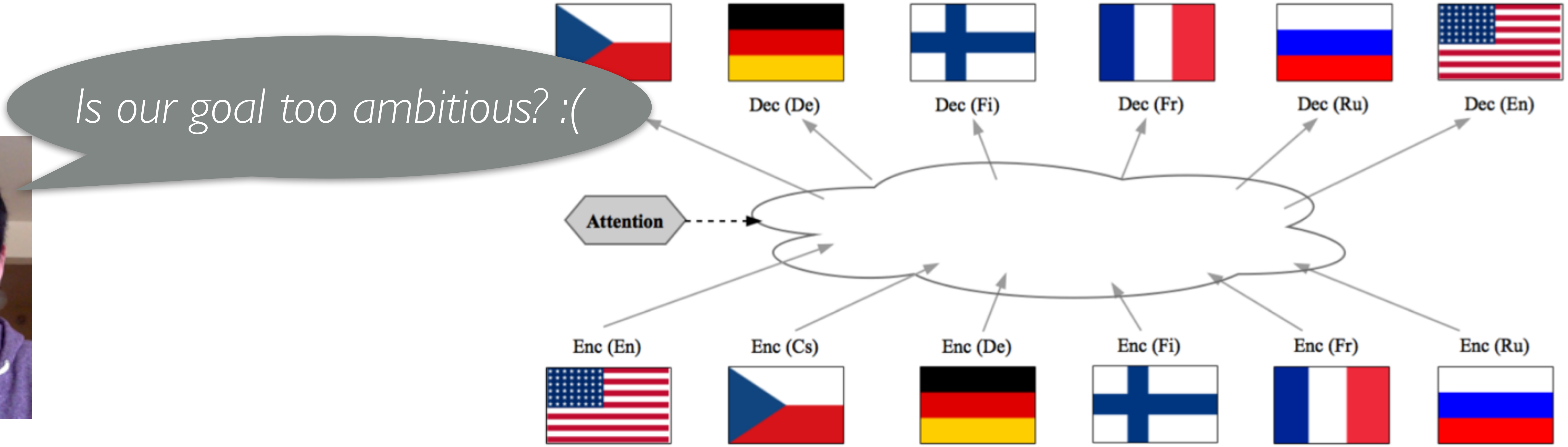
1. No attention: a single vector space shared across source and target languages/tasks.
2. Limited set of languages tested: English, German + **many other tasks**



CHALLENGES

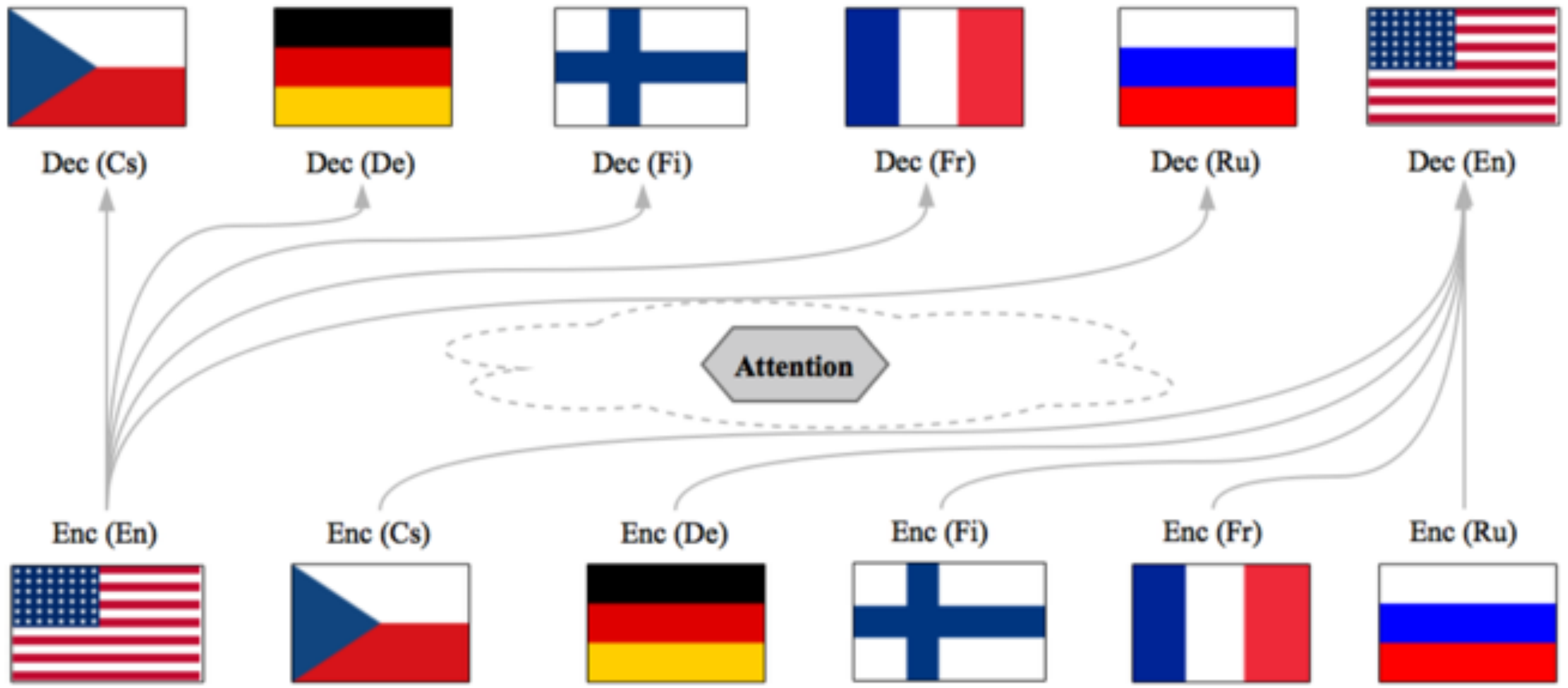
1. We have a strong belief that (soft-)alignment is specific to a language pair.
2. Even if not, there's a gigantic model space. How can we design a network?
3. 6 languages (En, Cs, De, Fi, Fr, Ru)
   - 60+ million bilingual sentence pairs for training
   - The entire model does not fit on one GPU



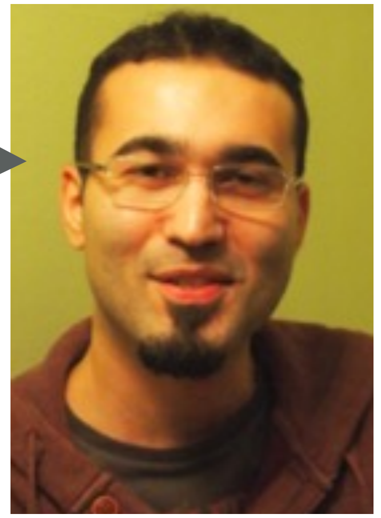


MULTI-WAY, MULTILINGUAL TRANSLATION

1. 10 language pair—directions from WMT'15
   - En → {Cs, De, Fi, Fr, Ru}, {Cs, De, Fi, Fr, Ru} → En
2. One alignment model for all the ten pair—directions.
3. Trained with bilingual parallel pairs only
4. The model was distributed over two GPU's



For details, find this guy!

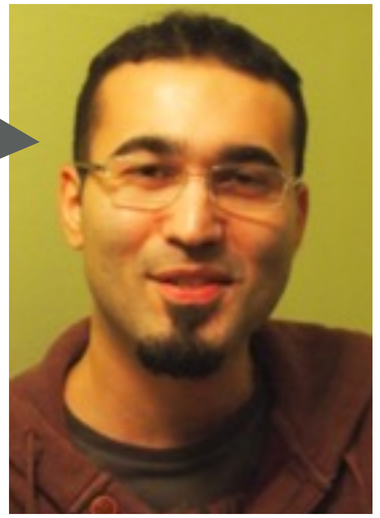


(Firat et al., 2016a)

MULTI-WAY, MULTILINGUAL TRANSLATION

		Dir	Fr (39m) → En	Fr (39m) En →	Cs (12m) → En	Cs (12m) En →	De (4.2m) → En	De (4.2m) En →	Ru (2.3m) → En	Ru (2.3m) En →	Fi (2m) → En	Fi (2m) En →
(a) BLEU	Dev	Single	27.22	26.91	21.24	15.9	24.13	20.49	21.04	18.06	13.15	9.59
		Multi	26.09	25.04	21.23	14.42	23.66	19.17	21.48	17.89	12.97	8.92
	Test	Single	27.94	**29.7**	20.32	**13.84**	24	**21.75**	22.44	**19.54**	12.24	**9.23**
		Multi	**28.06**	27.88	**20.57**	13.29	**24.20**	20.59	**23.44**	19.39	**12.61**	8.98
(b) LL	Dev	Single	-50.53	-53.38	-60.69	-69.56	-54.76	-61.21	-60.19	-65.81	-88.44	-91.75
		Multi	-50.6	-56.55	-54.46	-70.76	-54.14	-62.34	-54.09	-63.75	-74.84	-88.02
	Test	Single	-43.34	**-45.07**	-60.03	**-64.34**	-57.81	**-59.55**	-60.65	-60.29	-88.66	-94.23
		Multi	**-42.22**	-46.29	**-54.66**	-64.80	**-53.85**	-60.23	**-54.49**	**-58.63**	**-71.26**	**-88.09**

For details, find this guy!



SETTINGS

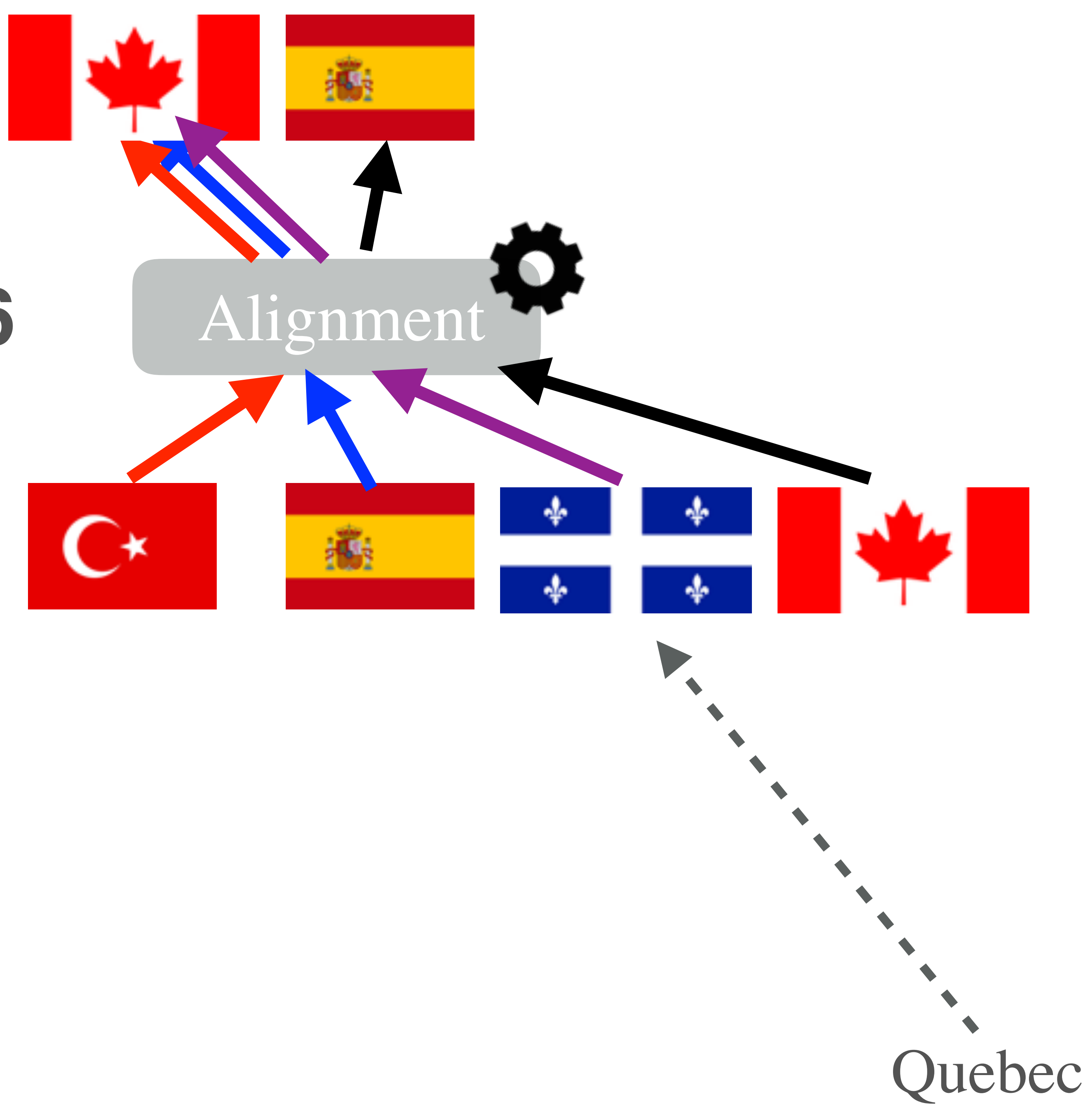
1. Target language pairs
   - Uzbek → English, Turkish → English
2. Auxiliary language pairs
   - French ↔ English, Spanish ↔ English

	# Symbols		# Sentence		
	# En	Other	Train	Dev	Test
En-Uz	1.361m	1.186m	73.66k	948	882
En-Tr	13.17m	12.43m	784.65k	862	940
En-Es	908.1m	924.9m	34.71m	3003	3000
En-Fr	1.837b	1.911b	65.77m	3003	3000

(Firat et al., 2016b; under review)
Work done in collaboration with IBM

TURKISH-TO-ENGLISH

1. Tr-En: 14.21/17.28
2. Tr-En+Es-En: 16.00/**17.75**
3. Tr-En+Es-En+Fr-En: 16.18/**18.13**
4. Tr-En+Es-En+Fr-En+En-Es: 16.28/**18.66**
5. Ensemble: 20.00/**22.56**
   - 3x Tr-En+Es-En+Fr-En
   - 3x Tr-En+Es-En+Fr-En+En-Es



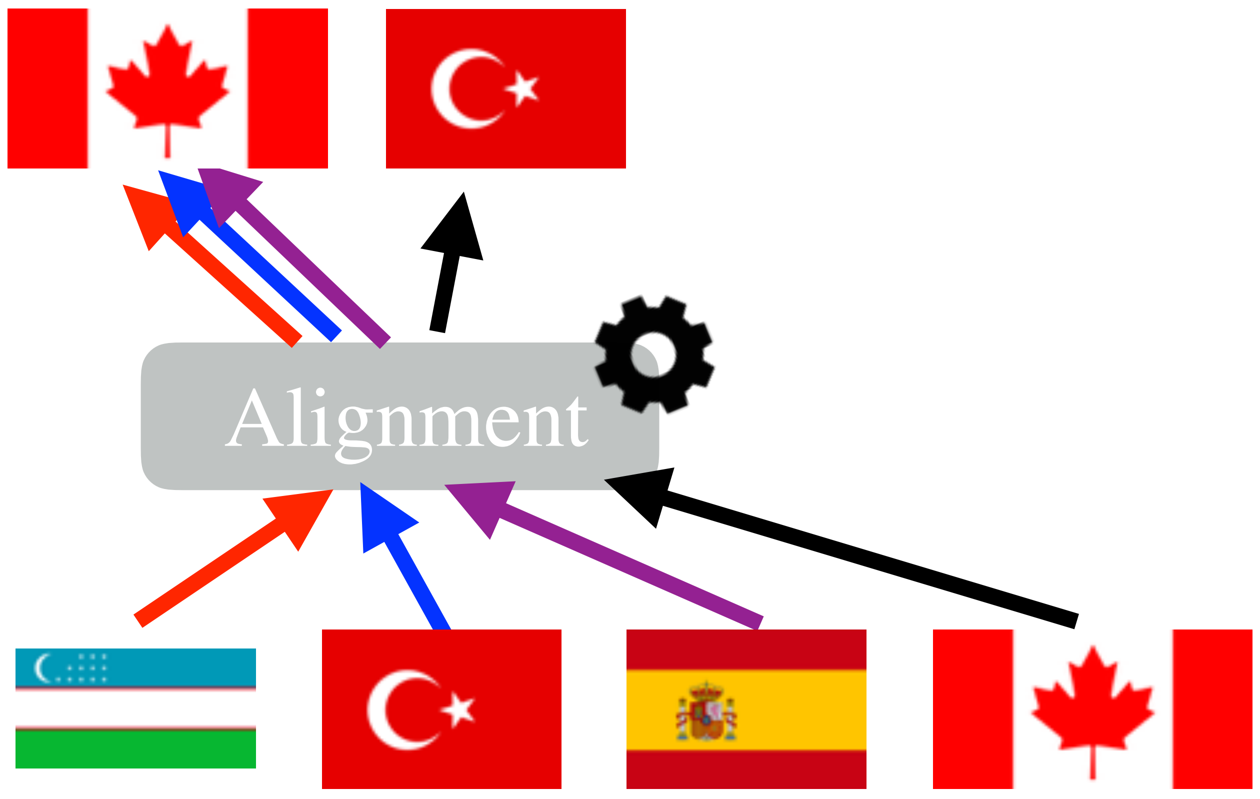
(Firat et al., 2016b; under review)

Work done in collaboration with IBM

UZBEK-TO-ENGLISH

1. Uz-En: 6.63/6.45
2. Uz-En+Tr-En: 8.68/**9.34**
3. Uz-En+Tr-En+Es-En: 9.55/**10.34**
4. Uz-En+Tr-En+Es-En+En-Tr: 8.93/**9.41**

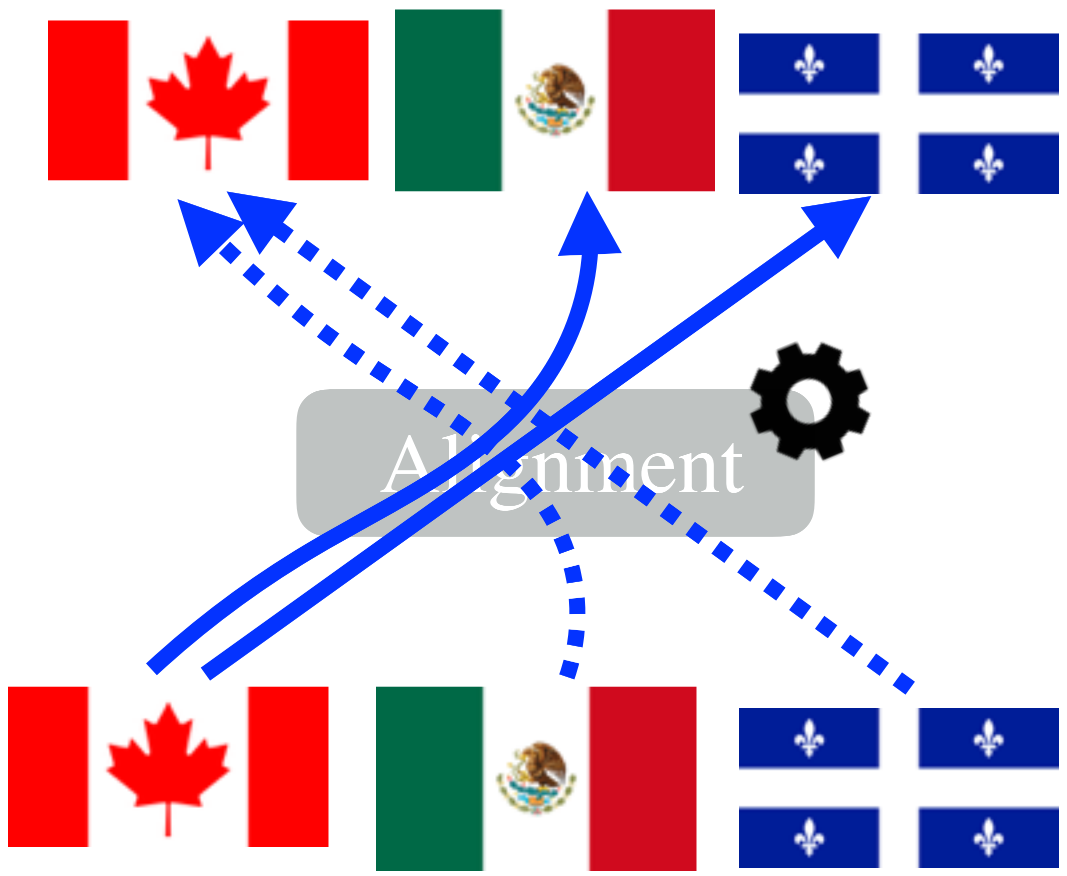
5. Ensemble: 12.17/**12.99**
    - 3x Uz-En+Tr-En+Es-En
    - 3x Uz-En+Tr-En+Es-En+En-Tr



(Firat et al., 2016b; under review)
Work done in collaboration with IBM

SETTINGS

1. Three languages: English, Spanish and French
   - {En, Es, Fr} ↔ {En, Es, Fr}
2. Bilingual corpora only during training: En → {Es, Fr}, {Es, Fr} ⇢ En
3. Multi-language source during test time



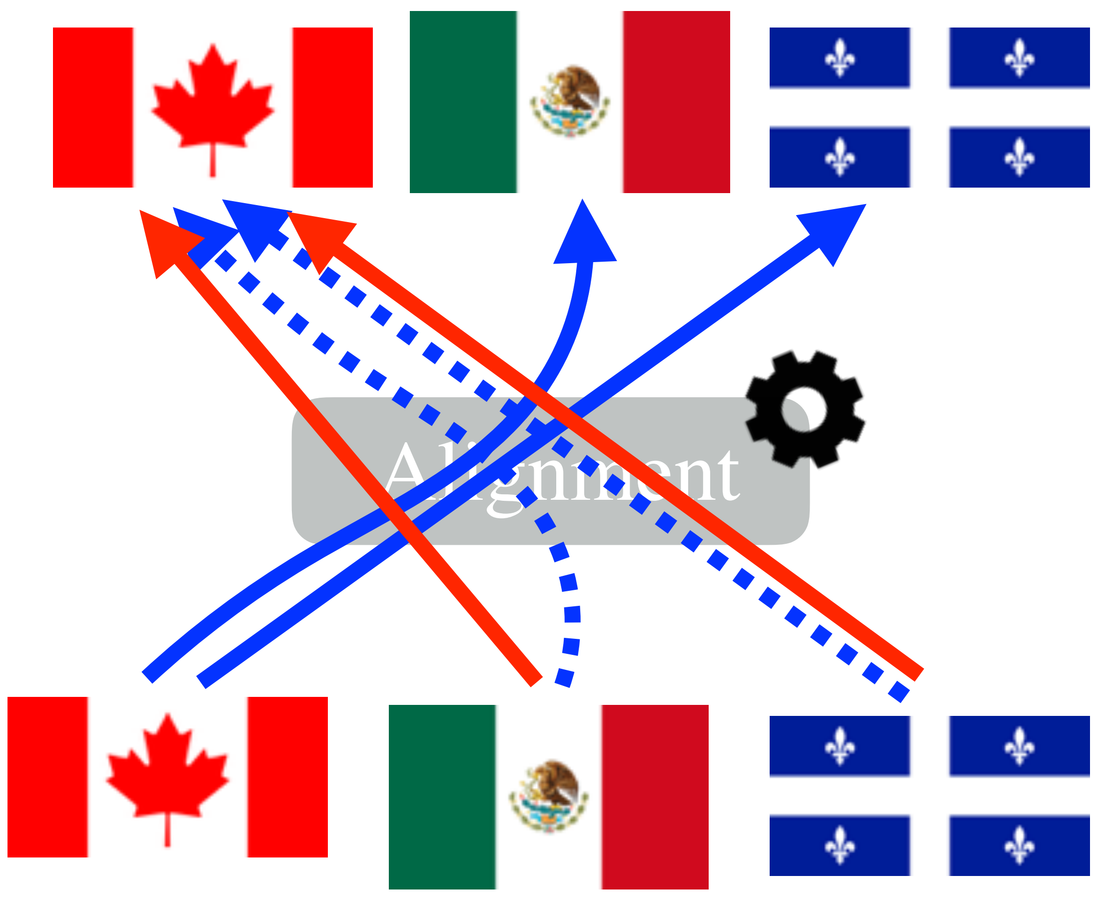
# Sents	Train	Dev†	Test‡
En-Es	34.71m	3003	3000
En-Fr	65.77m	3003	3000

(Firat et al., EMNLP 2016c)
Work done in collaboration with IBM

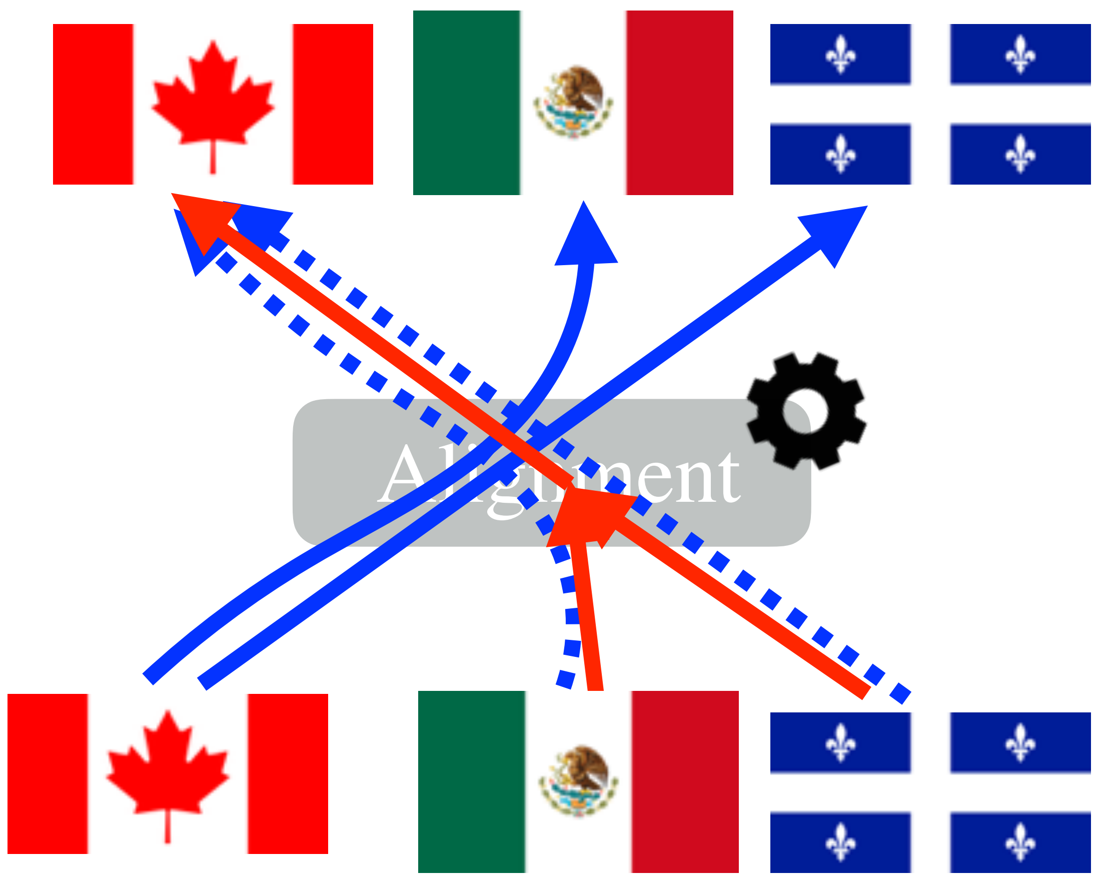
MULTI-SOURCE TRANSLATION?

1. (Es, Fr) → En
2. Two translation strategies

Late Averaging



Early Averaging*



* (Zoph & Knight, 2016)
(Firat et al., EMNLP 2016c)
Work done in collaboration with IBM Watson R&D

MULTI-SOURCE TRANSLATION? - **YES**

Single-source translation

	Src	Trgt	Multi Dev	Multi Test	Single Dev	Single Test
(a)	Es	En	30.73	28.32	29.74	27.48
(b)	Fr	En	26.93	27.93	26.00	27.21
(c)	En	Es	30.63	28.41	31.31	28.90
(d)	En	Fr	22.68	23.41	22.80	24.05

Multi-source translation

		Multi Dev	Multi Test	Single Dev	Single Test
(a)	Early	31.89	31.35	–	–
(b)	Late	32.04	31.57	32.00	31.46
(c)	E+L	32.61	31.88	–	–

But, single-pair models can apparently do multi-source translation..

(Firat et al., EMNLP 2016c)

Work done in collaboration with IBM Watson R&D

Deep Natural Language Processing
(3) Larger-Context Modelling

Context matters

- What does context tell us?
    - **Theme**/**Topic** of a document
- What does context tell us in practice?
    - *What are the words that are more likely to appear in this document?*

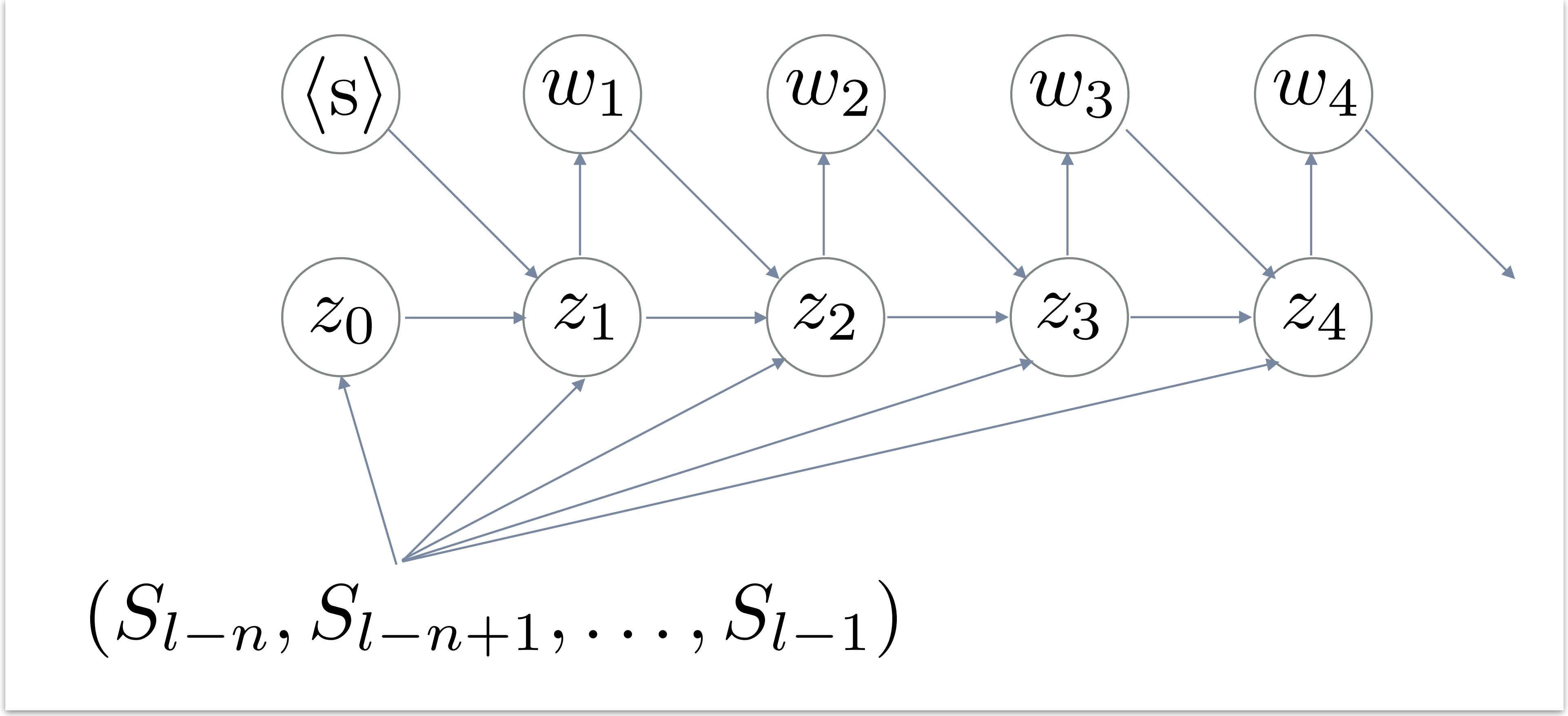
Context	Following Sentence
While it's not flawless, some motivations and scenarios remain somewhat underdeveloped or questionable; Ex_Machina is a stunning Sci-Fi vision that is also a fully formed thinking man's thriller.	With a jaw droopingly good turn from the soon to be megastar Vikander, ____**?**____ is another excellent example of what makes the ____**?**____

Larger-Context Language Modelling

- Language modelling as "document modelling" instead of "sentence modelling" (Wang & Cho, arXiv 2015; Ji et al., arXiv 2015)

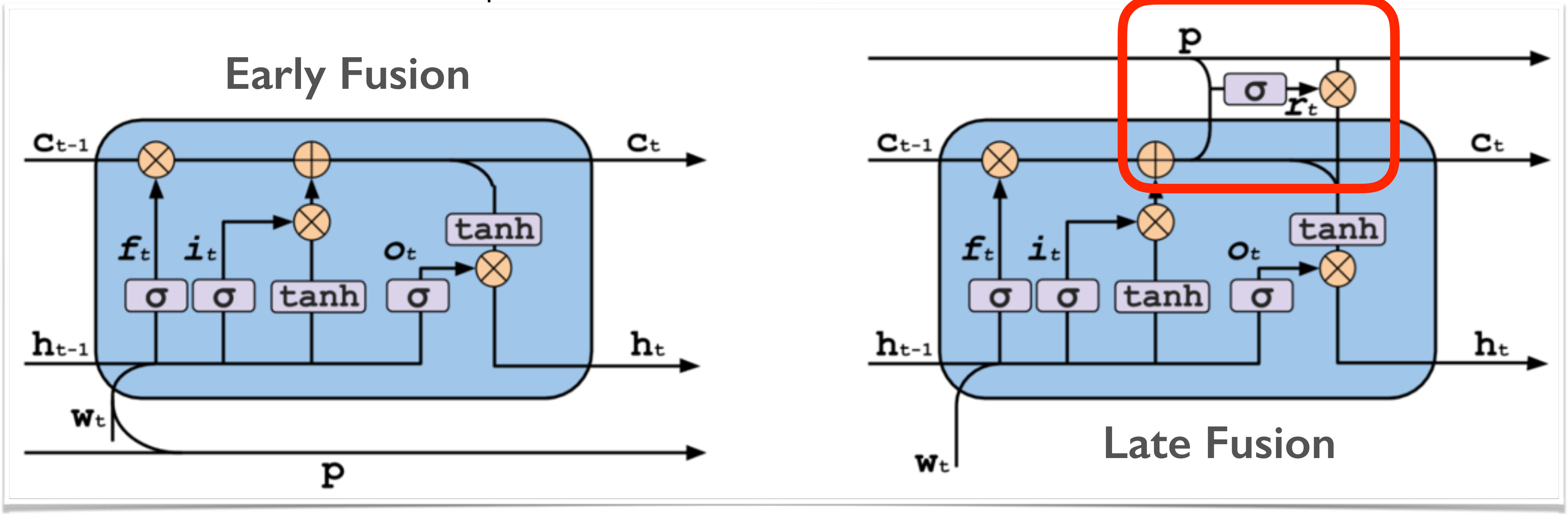
$$P(D) \approx P(S_1)P(S_2)\cdots P(S_N) \text{ vs. } P(D) \approx P(S_1)P(S_2|S_1)\cdots P(S_N|S_{N-n},\ldots,S_{N-1})$$

- Simplest approach (Wang & Cho, arXiv 2015)
  - Bag of all the words from the previous n sentences
  - RNN Language model conditioned on this bag-of-words



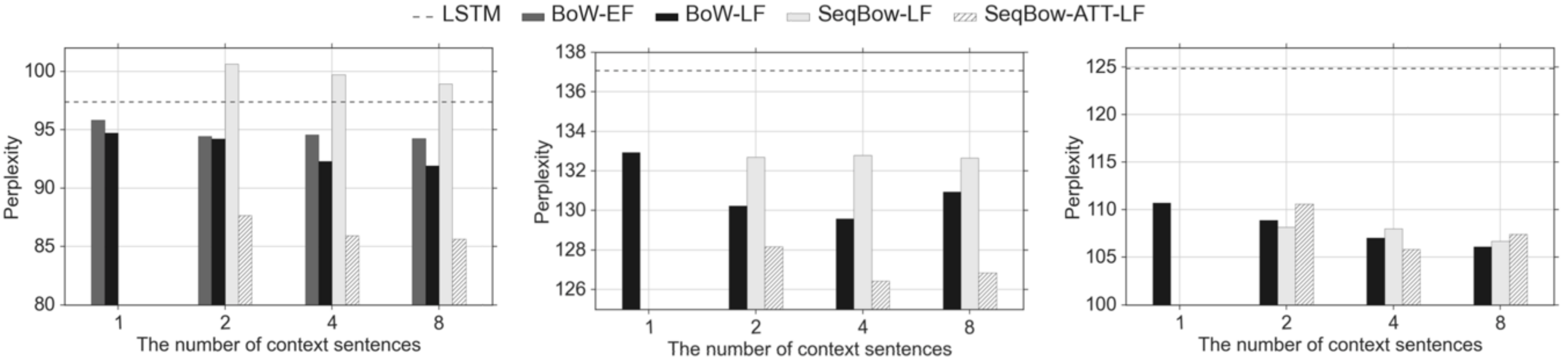
Larger-Context Language Modelling

- Late Fusion of LSTM (Wang & Cho, arXiv 2015)
  - Let the memory cell c model *intra-sentence* dependencies
  - Let the *inter-sentence* dependencies be fused in **later**



Larger-Context Language Modelling

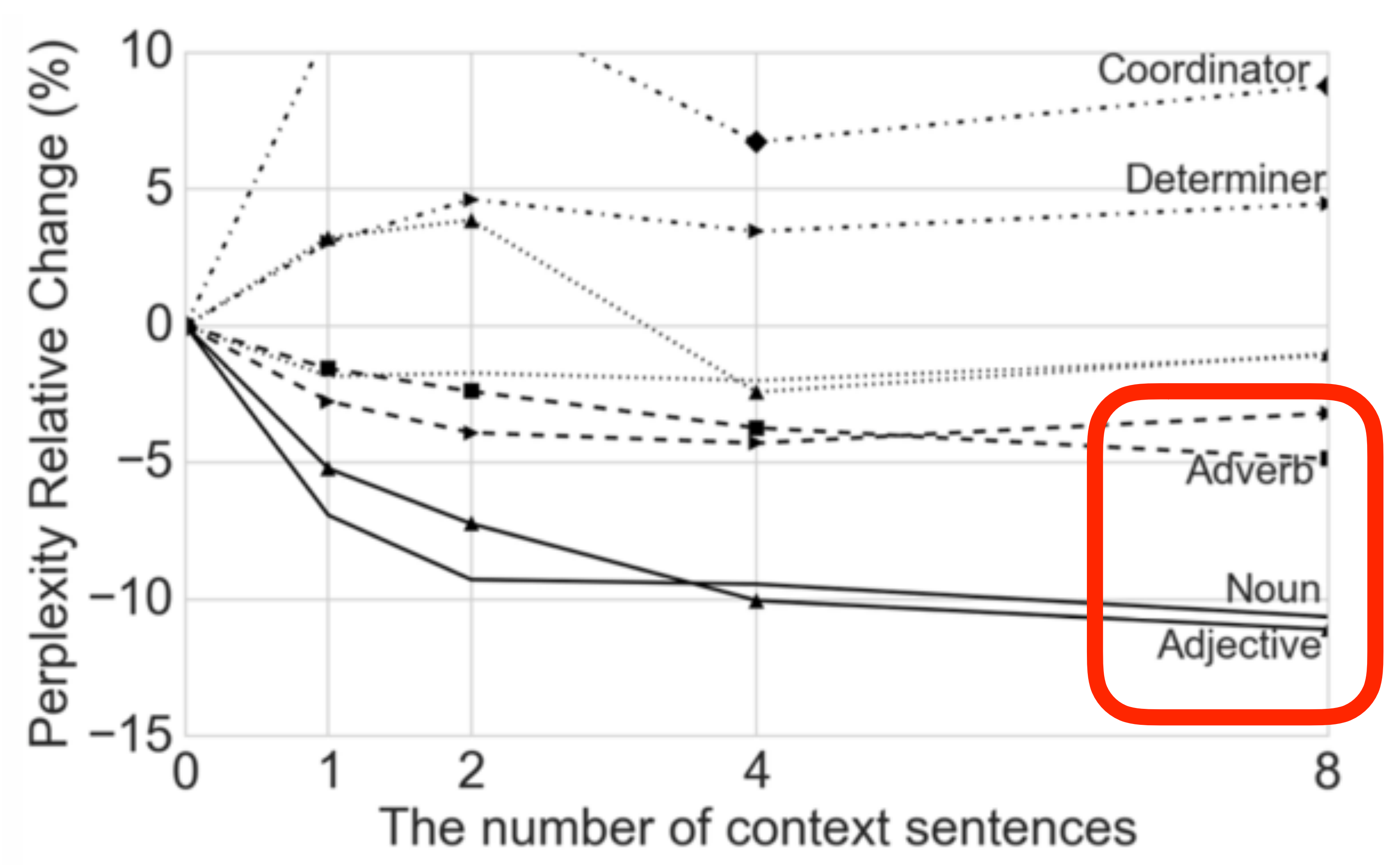
- It helps *obviously* (Wang & Cho, arXiv 2015)
  - Especially with the *late fusion* of context



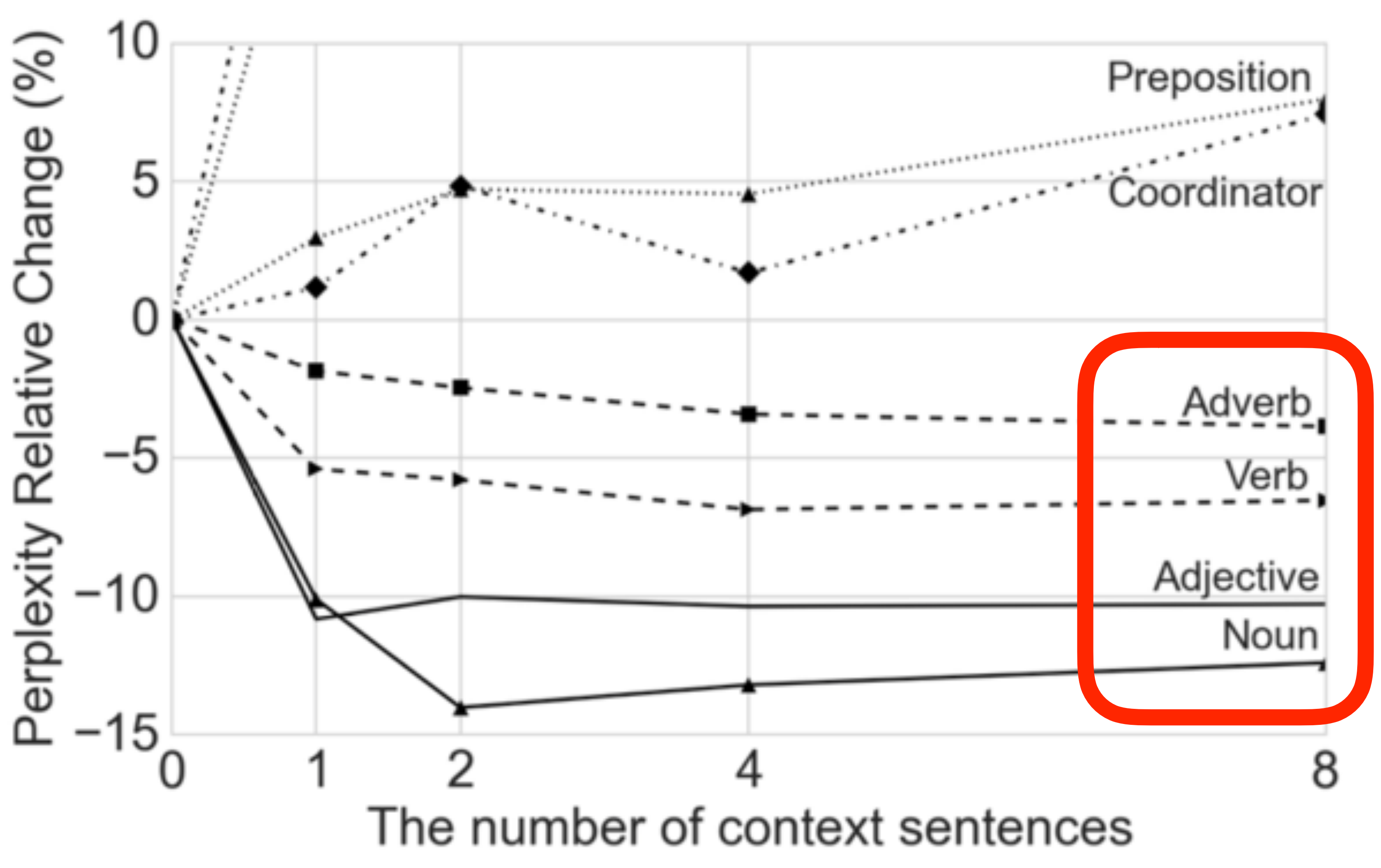
(a) IMDB (b) Penn Treebank (c) BBC

Larger-Context Language Modelling

- "*What are the words that are more likely to appear in this document?*"
    - Open-class words: nouns, adjectives, verbs and adverbs



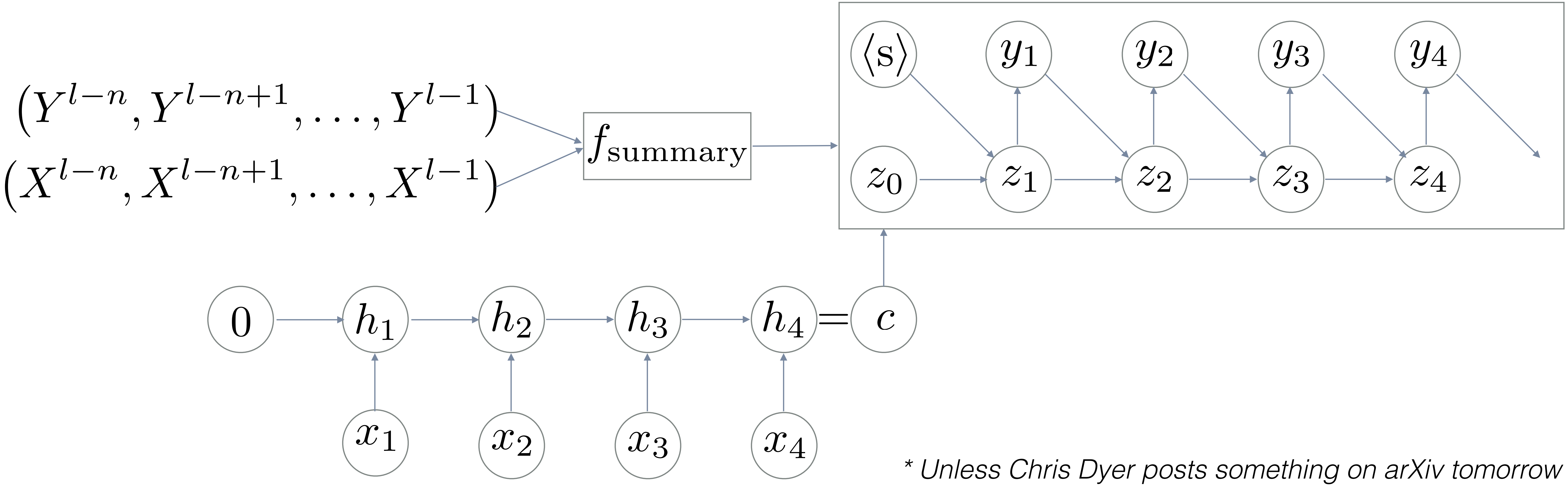
IMDB



PTB

Larger-Context Machine Translation

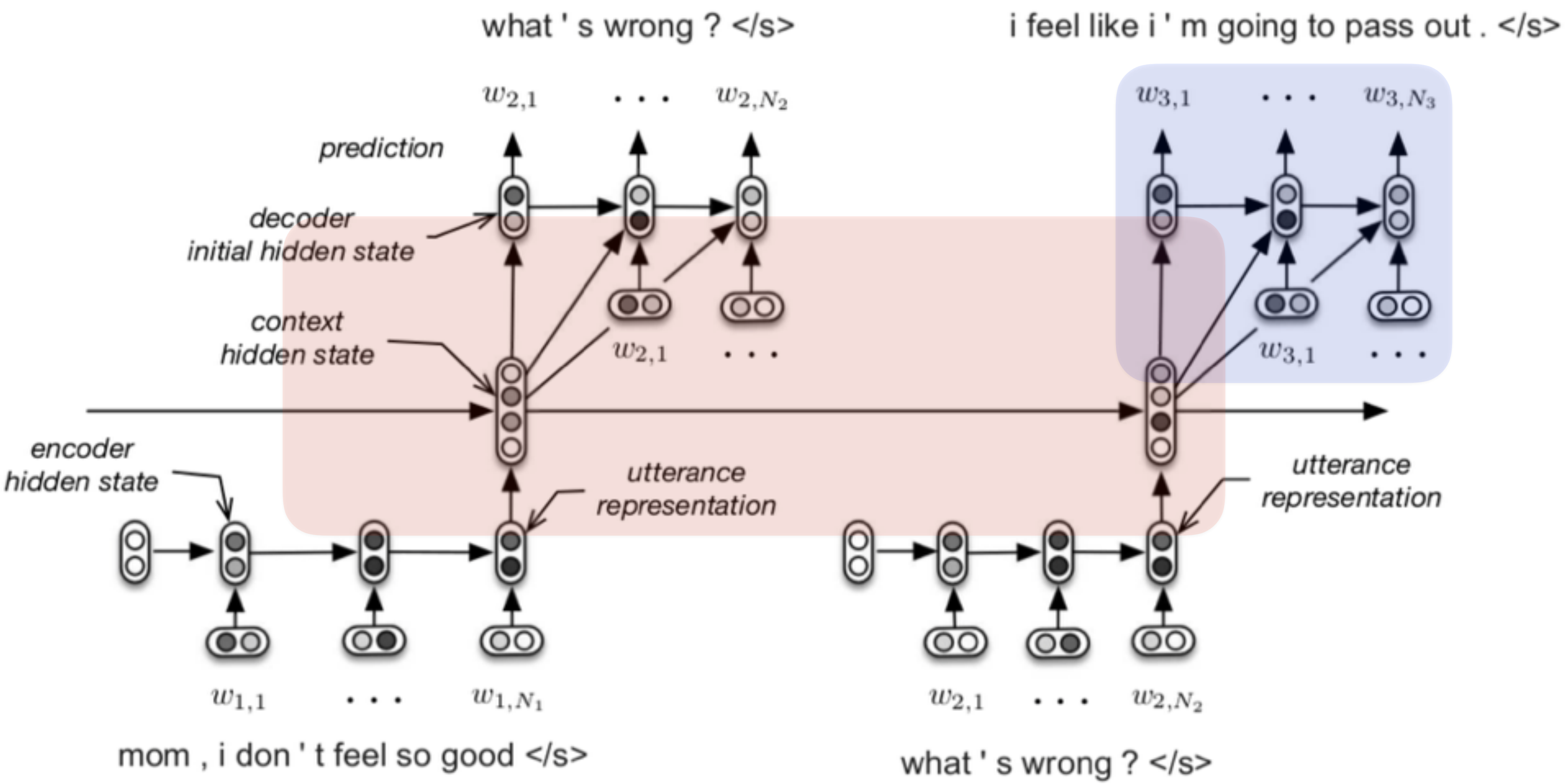
- Toward Larger-Context Machine Translation (Jean & Cho, Work in Progress*)
  - How to represent the source and target contexts?
  - Which is conditioned on the context, encoder, decoder or both?



** Unless Chris Dyer posts something on arXiv tomorrow*

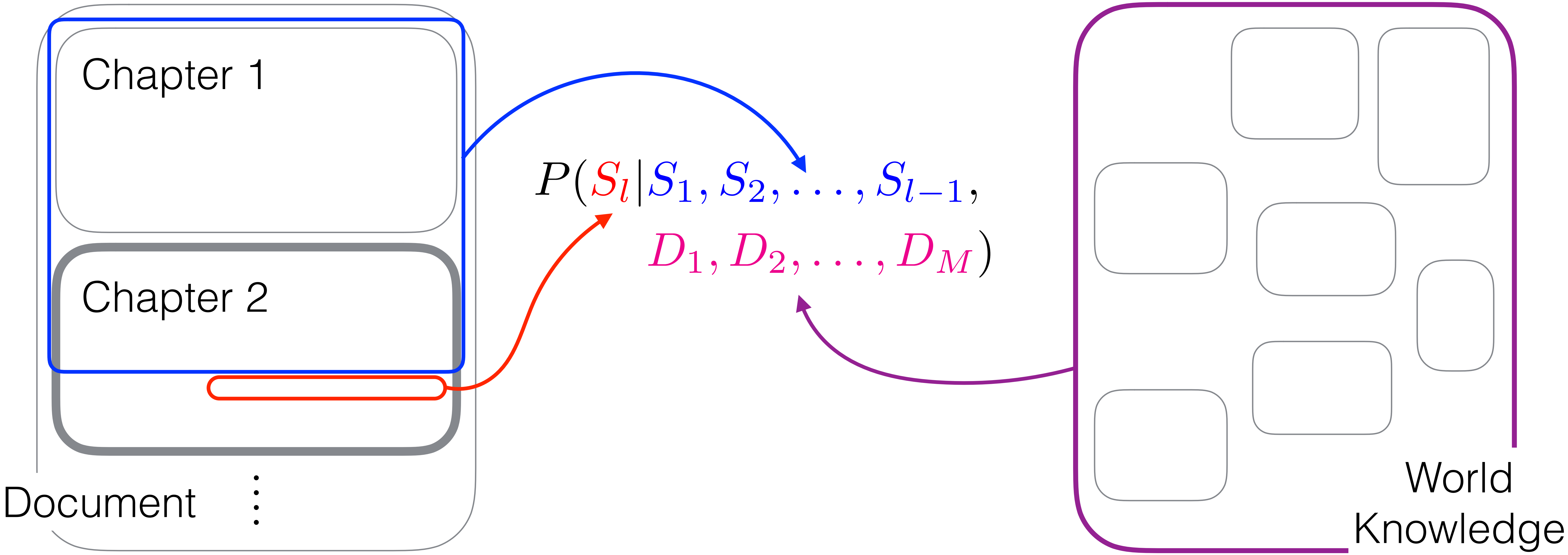
Dialogue-level Machine Translation

- Hierarchical Model for Dialogue Modelling (Serban et al., 2015; Sordoni et al., 2015)
  Utterance-level RNN + **Dialogue-level RNN**



World-Context Machine Translation

- Beyond Document-Level Language Processing
  - How do we *blend intra-document context and world knowledge?*



Thank You!