JOOST BASTINGS / NLP II

NEURAL MACHINE TRANSLATION

PHRASE-BASED VS NEURAL MT

Phrase-based SMT (e.g. Moses):

- The basic units are **phrases**
- Similar phrases do **not** share statistical weight
- This leads to **sparsity**: many rare/unseen phrase pairs!

Continuous representations:

- Capture **similarity** (morphological, syntactic, semantic)
- Can be used in language models, overcoming sparsity
- Are sensitive to conditioning information
- Can be constructed for phrases and sentences

NEURAL MACHINE TRANSLATION

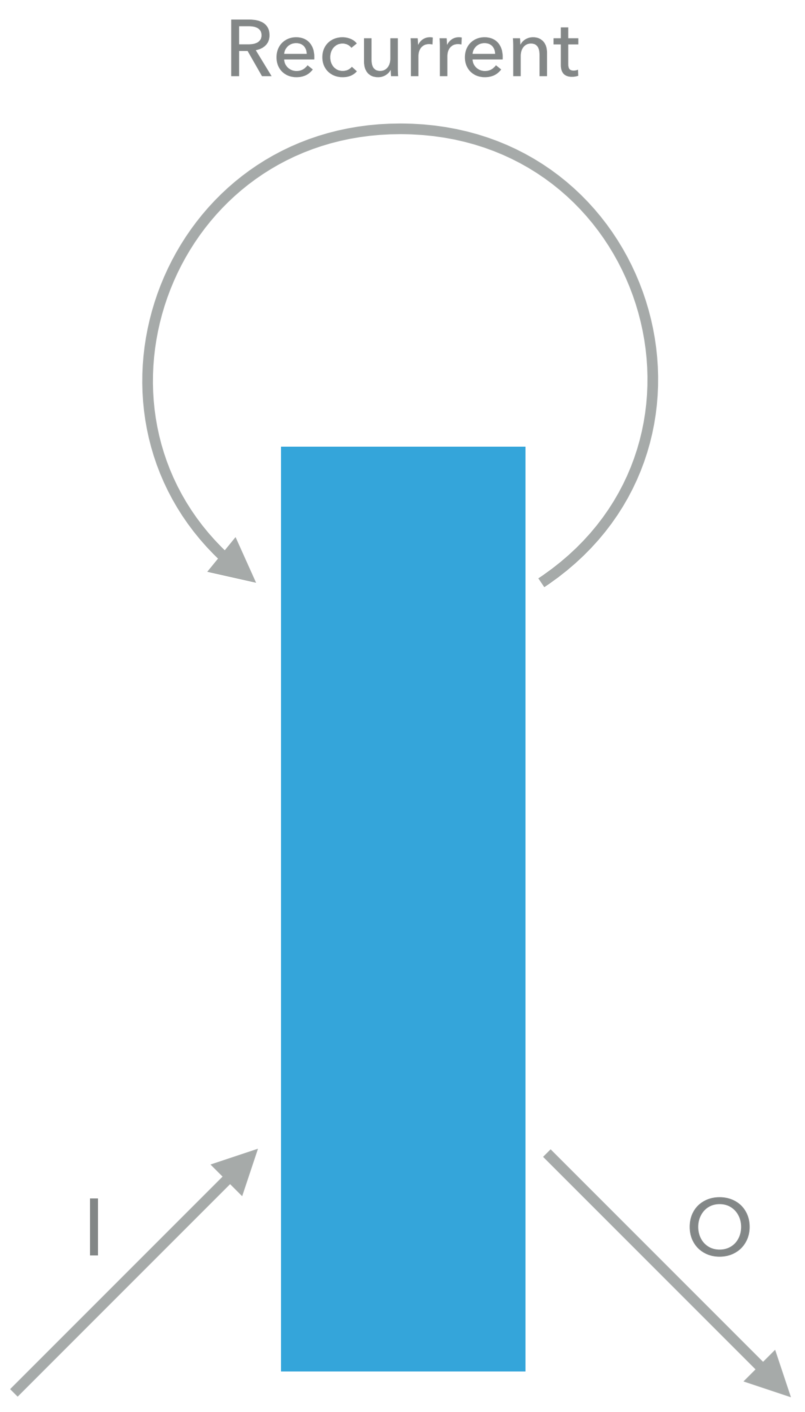
- A **recent** approach
- Build a neural net to **read** in the sentence and **output** a correct translation
- Most approaches consist of:
  - An **encoder** that captures the source sentence into a vector
  - A **decoder** that builds the target sentence from that vector

INTRODUCTION

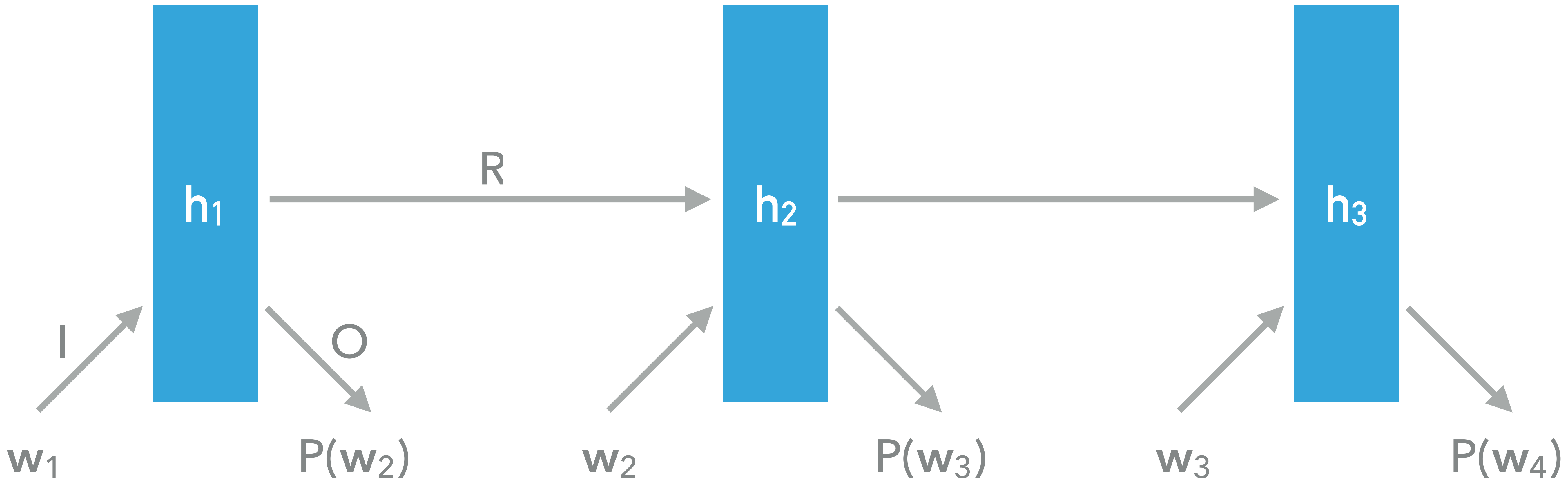
LANGUAGE MODELLING

RECURRENT LANGUAGE MODEL (RLM)

- Gives the probability of a sequence of words $w_1 \ldots w_n$
- Prob. is factorised as the **product** of each word given its previous words
- **I** transforms the input (a one-hot vector) into a word embedding
- **O** transforms the hidden layer into a next word prediction (output)



COMPUTATION



$$h_1 = \sigma(I \cdot w_1)$$

$$h_{i+1} = \sigma(R \cdot h_i + I \cdot w_{i+1})$$

$$o_{i+1} = O \cdot h_i$$

PART 1

RESCORING

INTRODUCTION TO RCTM

- A **Recurrent Continuous Translation Model** (RCTM) maps a **sentence** from the source language to a **probability distribution over sentences** in the target language
- So they can give us the **probability** of a translation
- There are **2 architectures** that both make use of a **recurrent language model** for the generation of the target sentence
- They differ in how they **condition** that target language model on the source sentence

Kalchbrenner & Blunsom (2013)

RCTM ESTIMATION

The probability of a target sentence $y_1 \ldots y_m$ given a source sentence (captured by vector **c**):

$$P(\mathbf{y} \mid \mathbf{x}) = \prod_{i=1}^{m} P(y_i \mid \mathbf{y}_{1:i-1}, \mathbf{c})$$

AN RCTM HAS 2 PARTS

1. A recurrent language model (RLM) as in Mikolov et al. (2010) that **predicts the next word** given all previous words
2. An architecture that **conditions** that prediction on the source sentence (context) **c**

RCTM I

- RCTM I can be seen as a **Recurrent Language Model** with the source sentence representation **c** provided to each hidden layer
- **c** is built by a **Convolutional Sentence Model** (CSM)

So what is a CSM?
(And what is convolution?)

Example: Convolution

CONVOLUTIONAL SENTENCE MODEL (CSM)

- A **CSM** models the representation of a **sentence** based on the continuous representations of the **words** in that sentence
- Let **X** contain source sentence $\mathbf{x}_1 \ldots \mathbf{x}_m$ (i.e. stacked vectors)
- A sequence of weight matrices K_2, K_3, … will act as **kernels**
- We convolve **X** using the K_i, reducing its size by i – 1 every time
- The vector that results is the sentence representation **c**

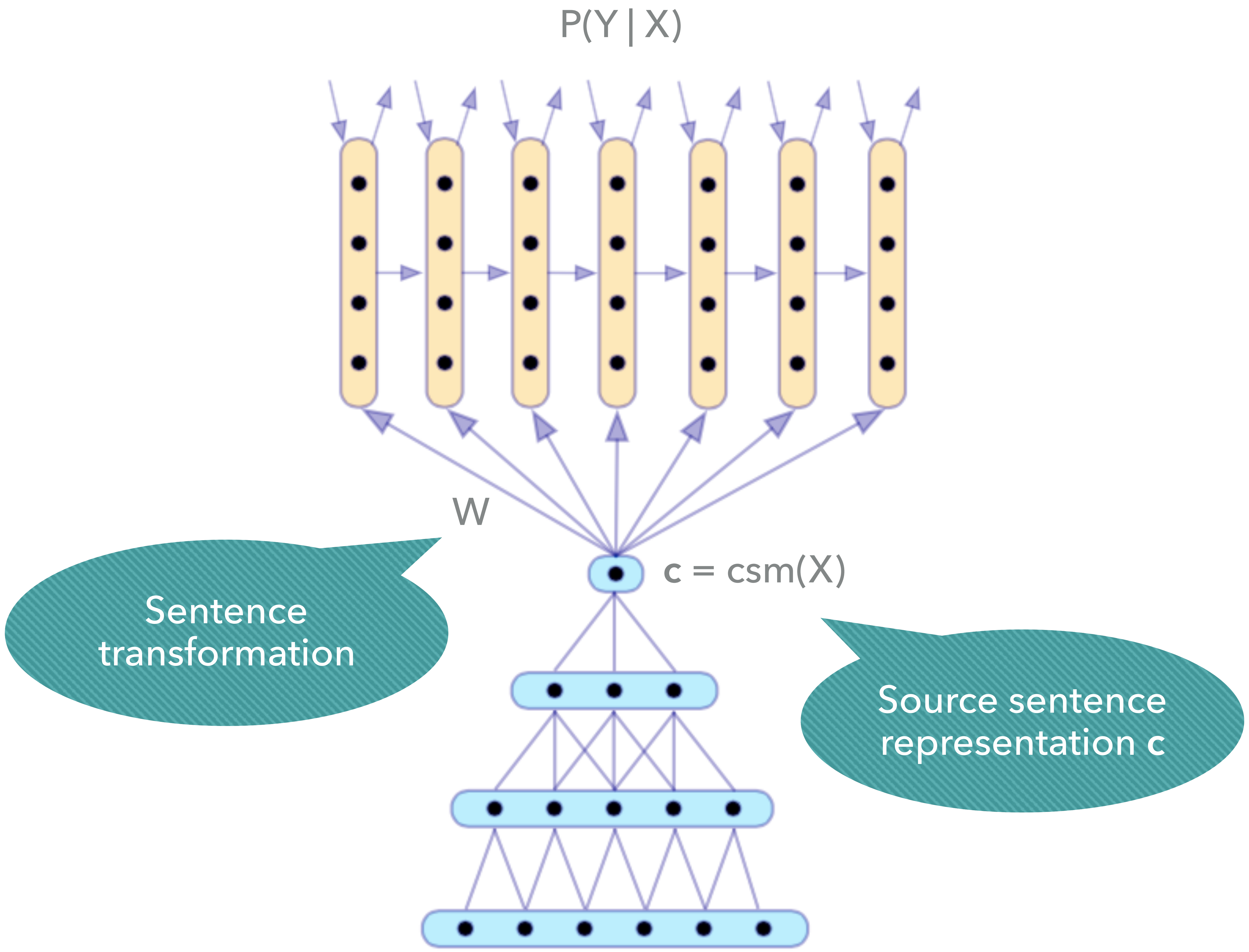
RCTM I COMPUTATION

$$\mathbf{h}_1 = \sigma(I \cdot y_1 + W \cdot \mathbf{c})$$

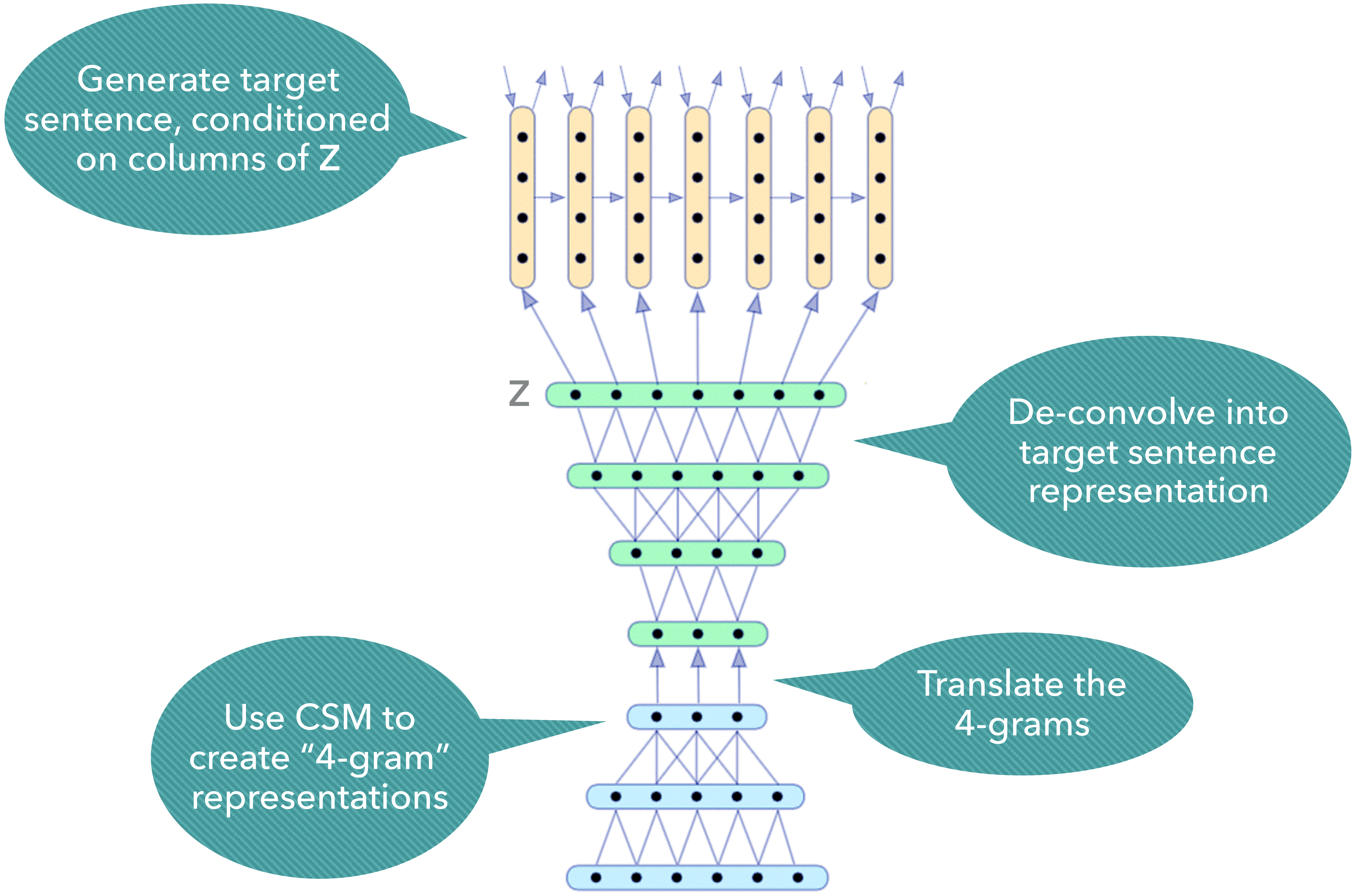
$$\mathbf{h}_{i+1} = \sigma(R \cdot \mathbf{h}_i + I \cdot y_{i+1} + W \cdot \mathbf{c})$$

$$\mathbf{o}_{i+1} = O \cdot \mathbf{h}_i$$

Almost the same computation as RLM!



RCTM II


Generate target sentence, conditioned on columns of Z
Z
De-convolve into target sentence representation
Translate the 4-grams
Use CSM to create “4-gram” representations

RESCORING

- The space for P(Y|X) is huge! (How huge?) V*
- We can take the **k best translations** from an SMT system and score them with the RCTM
- Then we don't need to search in the space of possible translations ourselves!

K-BEST LIST

1. Entre le début des années 70 , lorsque le Boeing 747 jumbo défini les voyages long-courriers , moderne et le tournant du siècle , le poids de l' Américain moyen des hommes de 40 à 49 ans a augmenté de 10 % , selon les données du ministère de la Santé américains (-20.9978)
2. Entre le début des années 1970 , lorsque le Boeing 747 jumbo défini les voyages long-courriers , moderne et le tournant du siècle , le poids de l' Américain moyen de sexe masculin de 40 à 49 ans a augmenté de 10 % , selon les données du ministère de la Santé américains . (-21.0182)
3. …

RESCORING WITH RCTMS

- The RCTMs are able to rescore *almost* **just as well as cdec** which uses 12 engineered features (5 translation models, 2 LMs, and a word penalty)

WMT-NT	2009	2010	2011	2012
RCTM 1	19.7	21.1	22.5	21.5
RCTM 2	19.8	21.1	22.5	21.7
cdec	19.9	21.2	22.6	21.8

DISCUSSION

- The **perplexity** of the RCTMs is much lower than IBM Model 1, even though they do not use word alignments
- RCTM II is shown to be **sensitive** to word order (worse perplexity when permuting the source sentences)
- Sampling from RCTM II gives **well-formed sentences**, showing sensitivity to syntax and semantics
- RCTMs were used successfully to **rescore** (rerank) 1000-best candidate translations from cdec, showing that they learned both translation and language modelling distributions

PART II

DECODING FEATURE

MOTIVATION

- So far we have used the neural net only **after** a phrase-based SMT system outputs its **k-best list**
- If we can use the neural net **during decoding** we can influence a much bigger space of translations!
- Devlin et al. (2014) introduce a **Neural Network Joint Model** with Bengio's NN Language Model as a starting point
- Technological contributions: **10.000x speed-up** by pre-computation & **self-normalisation**

NEURAL NETWORK JOINT MODEL

$$P(Y \mid X) = \prod P(y_i \mid y_{i-1:i-n+1}, S_i)$$

- S_i is the **source window** that is most relevant to y_i
- y_i is conditioned only on the $n - 1$ previous words

DEFINING SOURCE WINDOW S_i

- Each target word y_i is affiliated with 1 source word at index a_i
- S_i is then the **m-word source window** centred at a_i

  $\ldots, x_{a_i-1}, x_{a_i}, x_{a_i+1}, \ldots$

- There must **always** be **1 affiliated source word**
  - If multiple aligned words, take the middle one
  - If unaligned, use the affiliation of the closest aligned word, with preference to the right

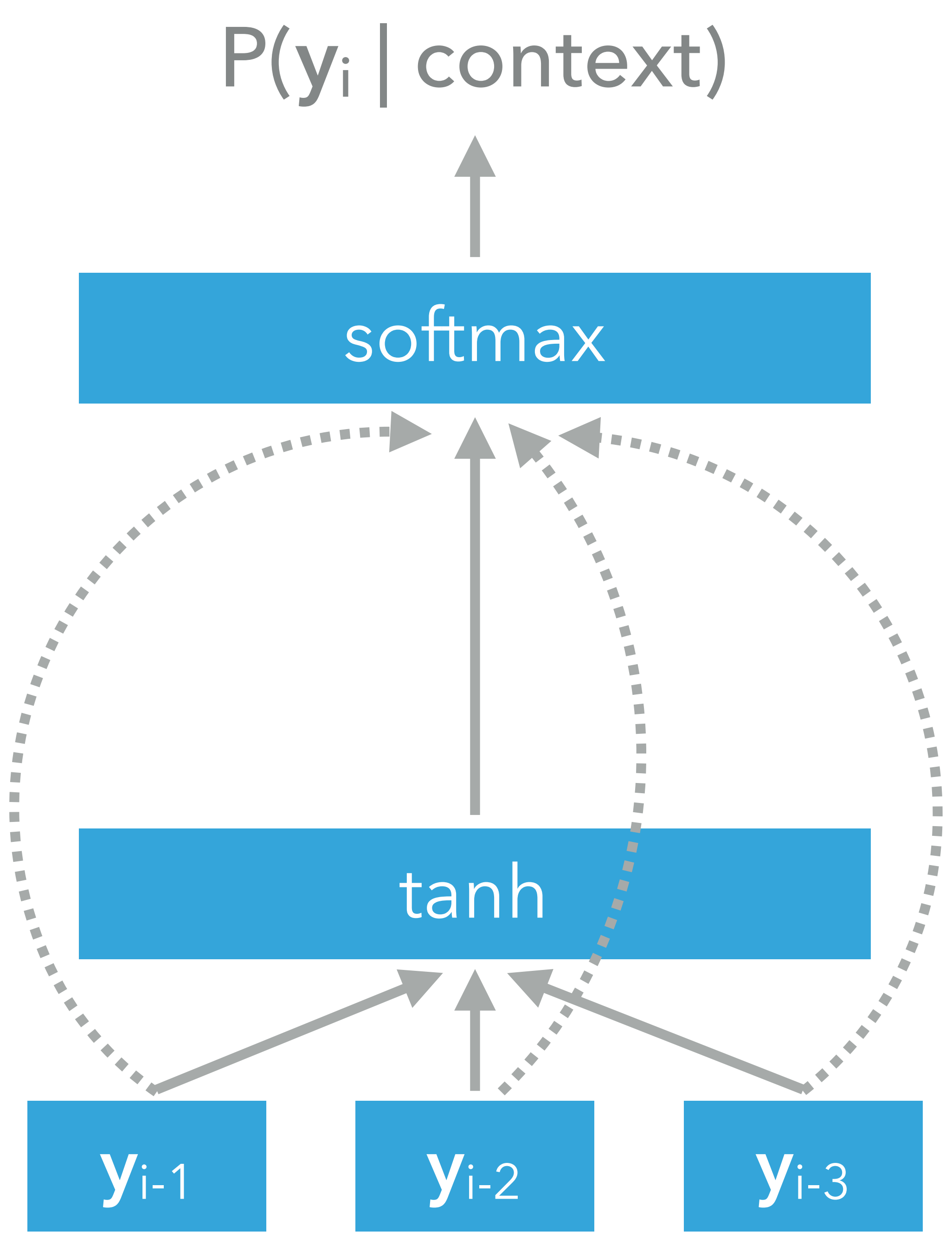
EXAMPLE: CHINESE-TO-ENGLISH

S:	我	3 就	4 取	5 钱	6 给	7 了	她们
	i	*will*	*get*	*money*	*to*	perf.	*them*
T:	2 i	1 will	0 get	the	money	to	them

P(the | get, will, i, 就, 取, 钱, 给, 了)

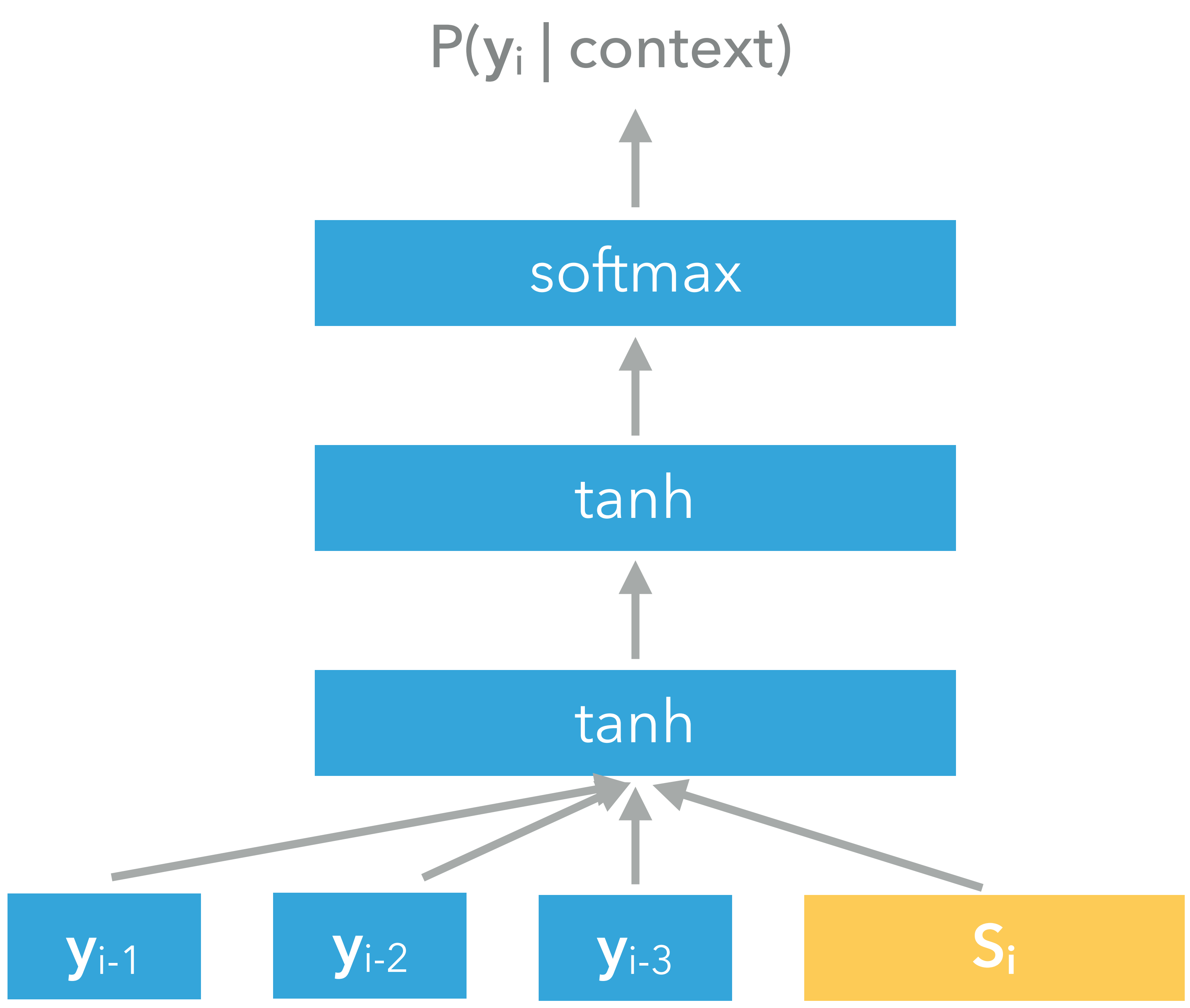
- This is essentially an (n+m)-gram **language model!**
- In all experiments n=4 and m=11
- A traditional LM would be extremely **sparse** with this amount of context

RECAP: BENGIO'S NEURAL NETWORK LANGUAGE MODEL



ARCHITECTURE

- Like Bengio et al. (2003)
- Input is 3 target, 11 source word vectors
- Input vocabulary: 16000 src & 16000 target
- Output vocabulary: 32000 target words



TRAINING

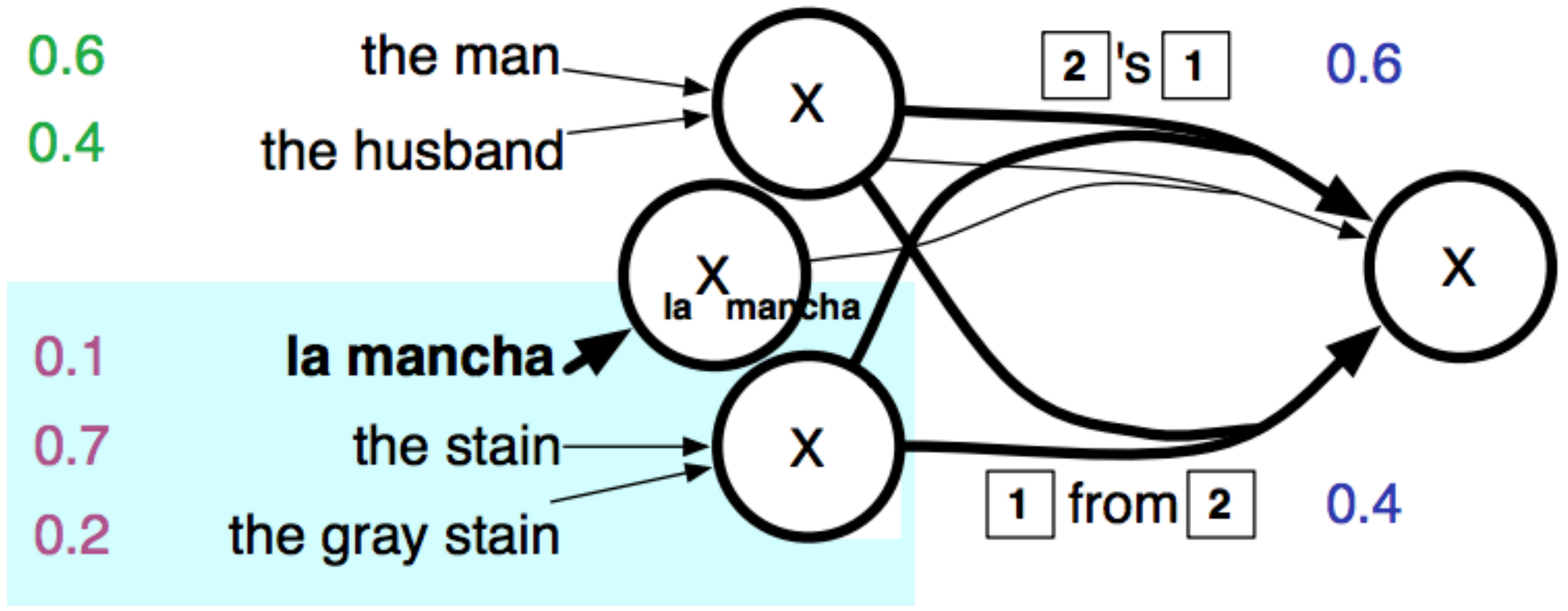
- Training is similar to Bengio et al. (2003)
- This time we use a **parallel corpus**
- We maximise the **log-likelihood** of the target words $\mathbf{y}_i$ in the training data

$L = \sum_i \log P(\mathbf{y}_i)$

- Optimisation: **back-propagation** with stochastic gradient ascent, **mini batches**

DECODING

- Language models are easily **integrated** in an SMT decoder
- We can store the **target word span** in state space, so a LM can compute the probability of the translation



- Now we also store their **affiliated source words**

DISCUSSION

- **+3.0 BLEU** on top of a strong and feature-rich baseline
- **+6.3 BLEU** on top of a simple hierarchical baseline
- The model is simple: no linguistic resources, no feature engineering, and only a few hyper-parameters
- Any thoughts? Criticisms? What are the contributions?

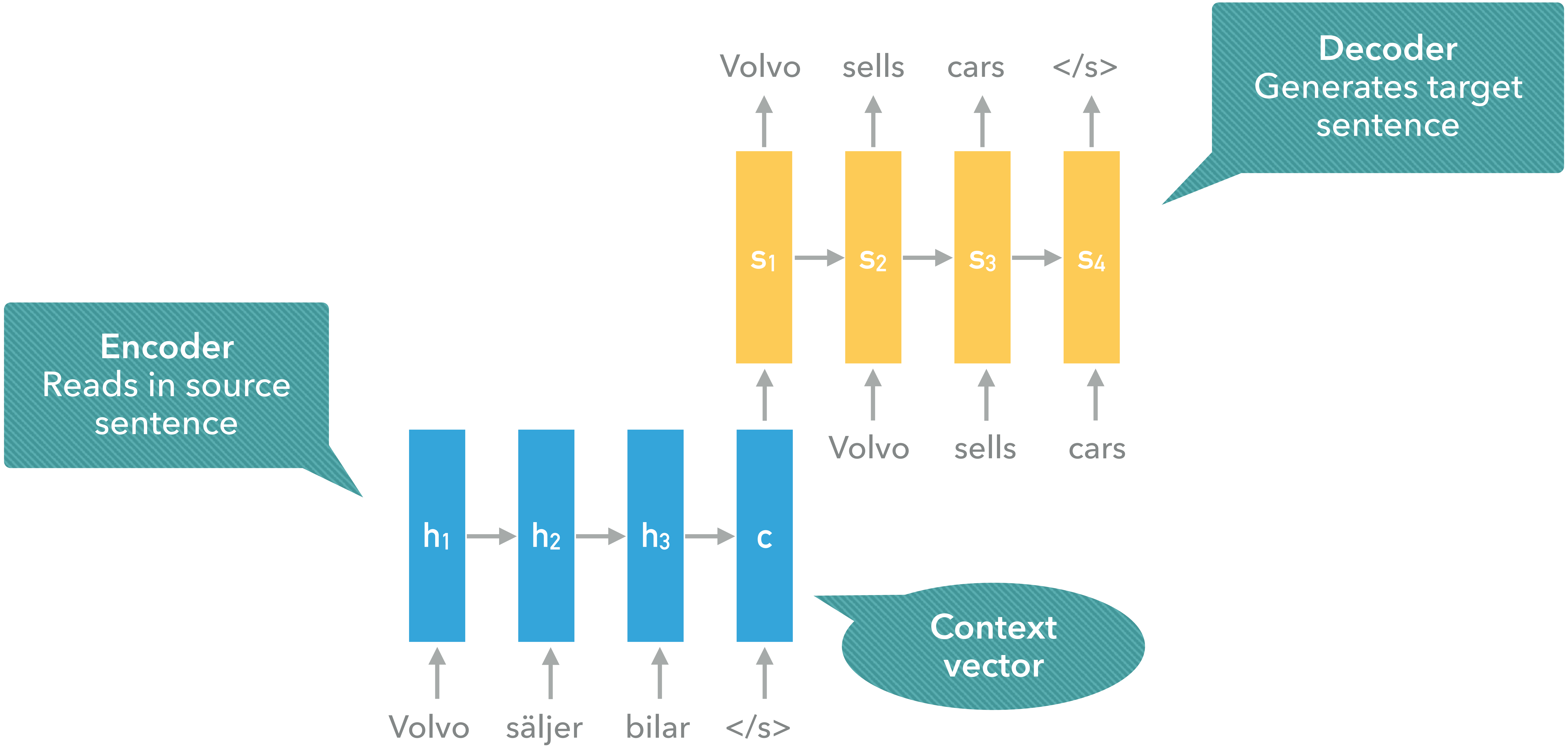
PART III

END-TO-END NMT

END-TO-END NMT

- We still haven't actually **translated** using neural nets!
- Another approach looks at sentences more generally as **sequences (of words)**
- **MT** then basically is **sequence-to-sequence** translation
- **Key problem** (as before): Neural nets require **fixed** input and output dimensions, and with sentences these vary!

ENCODER-DECODER



ENCODER-DECODER (FORMAL)

- Words are one-hot vectors

Example: $\mathbf{x}_1 = [0, 0, 0, 1, 0, 0]^T$

- A sentence is a sequence of words

$$\mathbf{x}_{1:n} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$$

$$\mathbf{y}_{1:m} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m)$$

ENCODER-DECODER (FORMAL)

- Calculate the **hidden states** of the **encoder RNN**:

$$h_1 = f(x_1, h_0)$$
$$h_2 = f(x_2, h_1)$$
$$\vdots$$

$$\mathbf{h}_j = f(\mathbf{x}_j, \mathbf{h}_{j-1})$$

LSTM or GRU

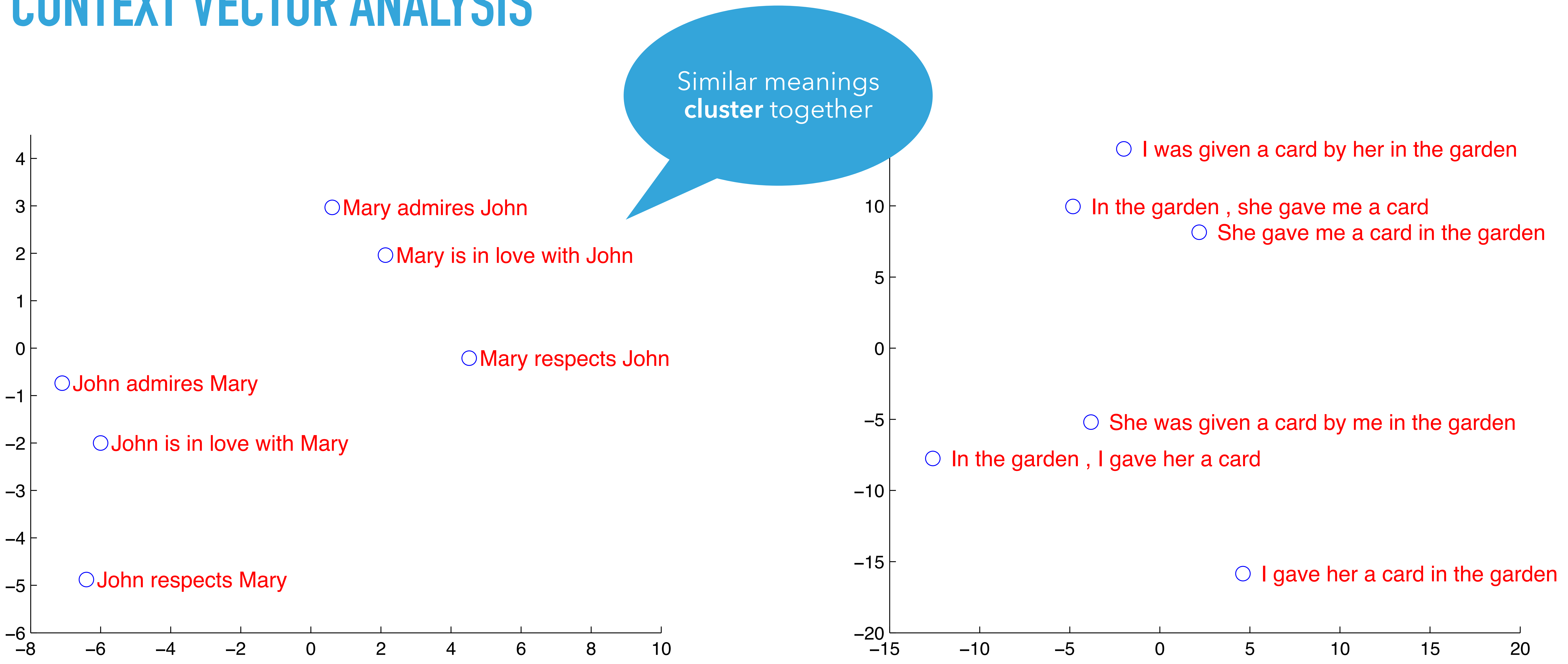
- Use **last hidden state** as context vector, i.e. $\mathbf{c} = \mathbf{h}_n$

- Get **translation probability** from the **decoder RNN**:

$$p(\mathbf{y}_{1:m}) = \prod_{i=1}^{m} g(\mathbf{y}_{i-1}, \mathbf{s}_i, \mathbf{c})$$

softmax

CONTEXT VECTOR ANALYSIS

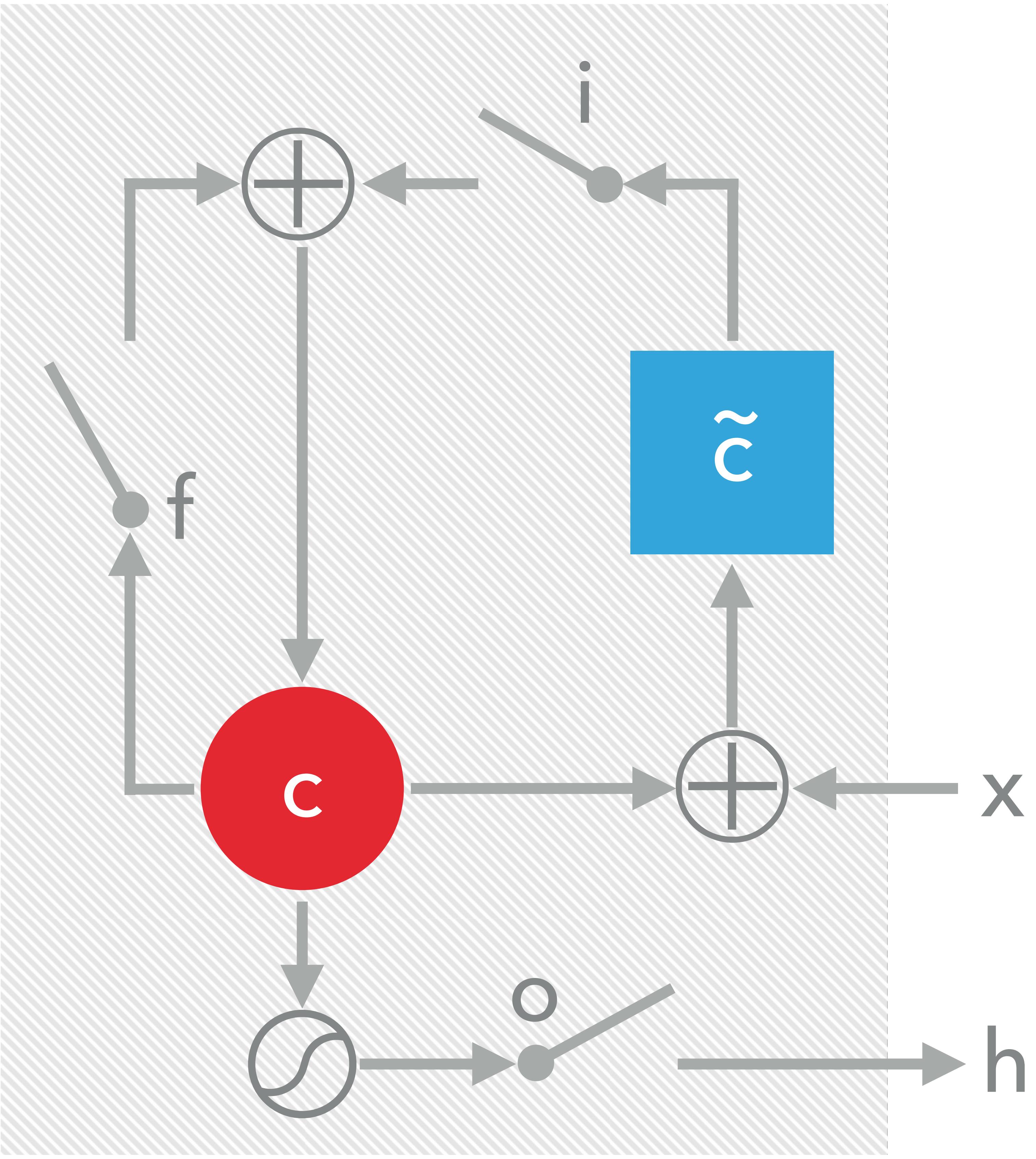


ENCODER-DECODER ISSUES

- One **fixed-size vector** needs to capture the meaning of the **whole** src sentence
- This includes **very long sentences**
- Cho et al. (2014) showed that this is **problematic**
- Sutskever et al. (2014) found that **reversing the input sentence** helps

LONG SHORT-TERM MEMORY (LSTM)

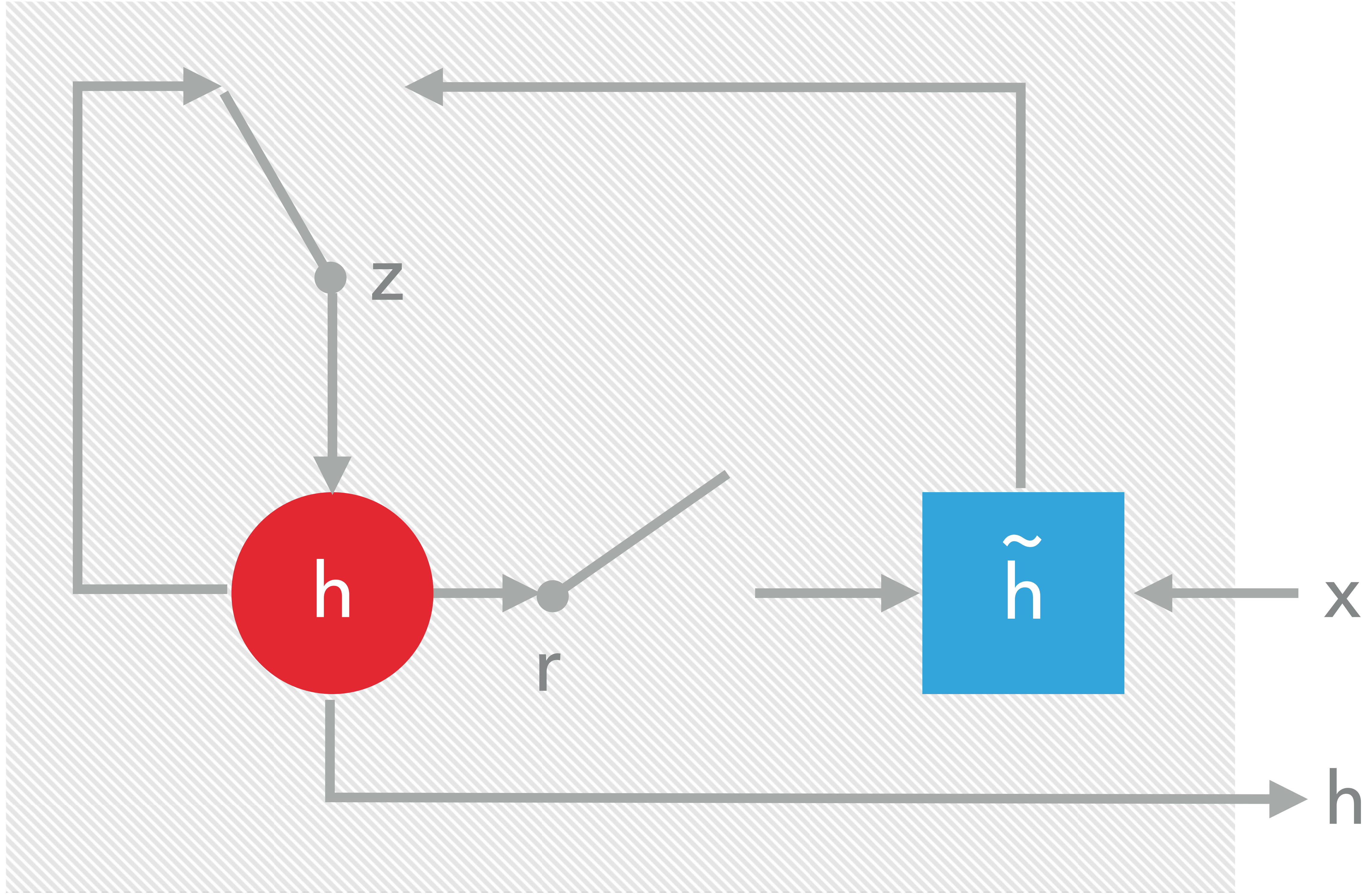
- A type of **RNN** introduced by Hochreiter and Schmidhuber (1997)
- Each unit has **input** (i), **forget** (f) and **output** (o) gates
- The gates of a single node **protect it** from irrelevant inputs, **and protect the other nodes** from a currently irrelevant activation



LSTM

GATED RECURRENT UNIT (GRU)

- Introduced by Cho et al. (2014)
- Simpler version of LSTM
- Each unit has an **update gate** (z) and a **reset gate** (r)



GRU

TRAINING

- Training objective: $\frac{1}{|S|} \sum_{(X,Y) \in S} \log p(Y|X)$
- Sutskever et al. use:
  - **4 layers**, each 1000 cells, and 1000-dimensional word embeddings
  - Input vocabulary **160k**, output vocabulary **80k**
  - A sentence is represented by **8000** real numbers
  - Training with **stochastic gradient descent**, in **mini batches**
    - … **10 days** on an **8-GPU** machine!

DECODING

- After training, we would like to get the **most likely translation**

  $\hat{Y} = \arg\max p(Y \mid X)$

- We can find it with a **beam search**:
  - Go from **left to right**, always keeping **a small number** of partial hypotheses (a prefix of Y)
  - At **each** time step, **expand** with **each** possible word
  - … then keep only the **best B** hypotheses

EXPERIMENTS

- WMT '14 task (English to French)

Baseline	33.30
LSTM	26.17
LSTM (reversed)	30.59
5 reversed LSTMs, beam size 12	34.81
Best WMT result	37.0

- Rescoring the baseline with the best setup gives BLEU = 36.5

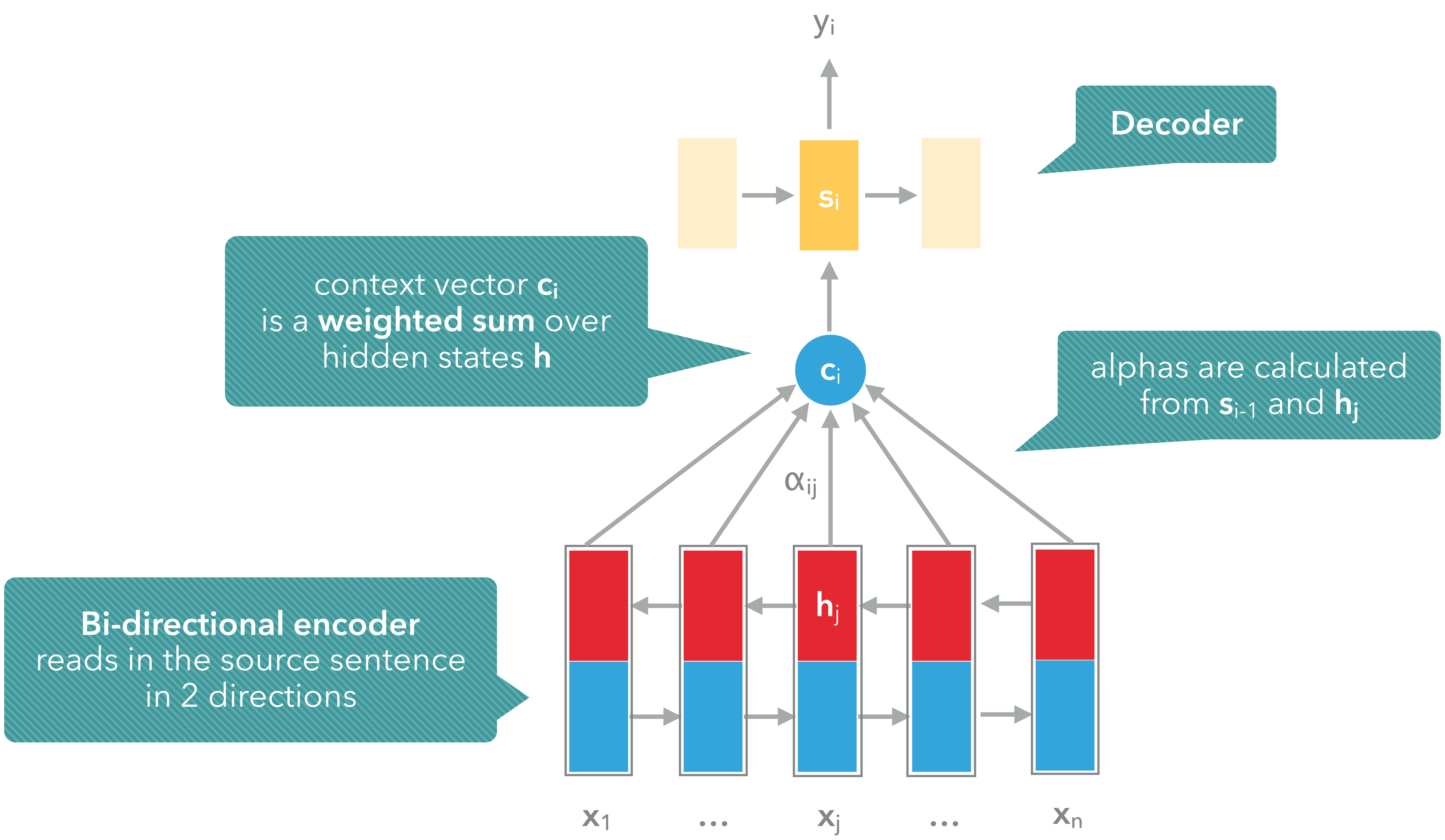
DISCUSSION

- The decoded translations did not outperform the state-of-the-art on WMT'14
- … but this was the **first time** that a pure neural MT system outperformed a phrase-based SMT baseline on a large scale MT task
- Surprising: LSTM did well on long sentences
- Initial results reported only on the (relatively easy) **English-French** task
- **In WMT '15, NMT systems ranked first on En→Cs and En→De**

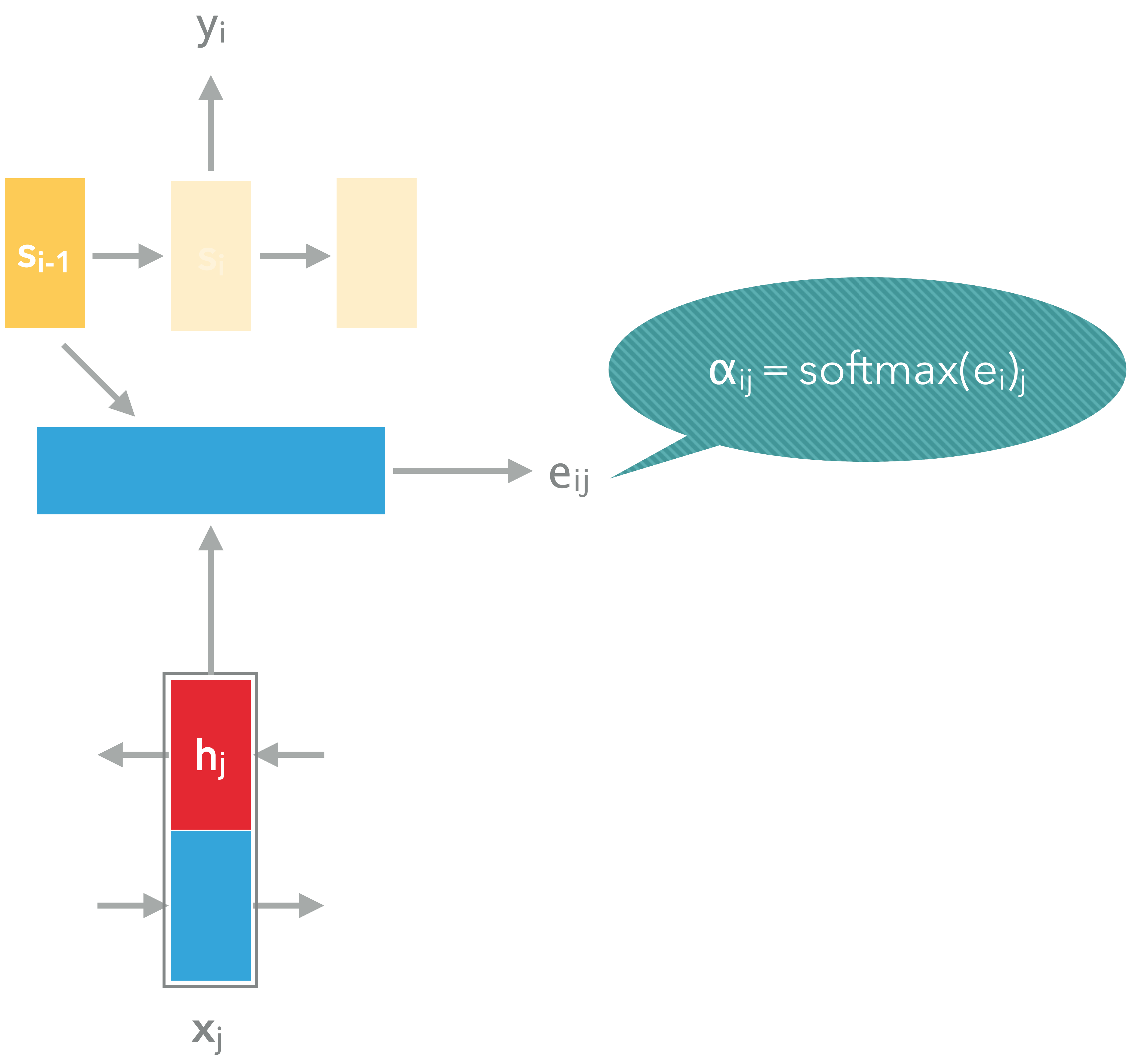
SUMMARY

- Neural networks can be used for:
  1. Rescoring k-best lists
  2. As a powerful decoding feature
  3. End-to-end Machine Translation
- Only Devlin et al. (2014)'s model can be easily **integrated** in a beam-search decoder, because it uses a limited source window
- Sadly, not all papers report pure MT results, so we cannot compare the various approaches very well
- **Recent work** focuses on incorporating language models, sub-word units, and using attention

NMT WITH ATTENTION (BAHDANAU ET AL., 2015)



CALCULATING ATTENTION WEIGHTS (FFNN)



REFERENCES

- Bahdanau et al. (2015). Neural Machine Translation by Jointly Learning to Align and Translate.
- Bengio et al. (2003). A Neural Probabilistic Language Model.
- Cho et al. (2014). Learning Phrase Representations using RNN Encoder-Decoder for SMT.
- Devlin et al. (2014). Fast and Robust Neural Network Joint Models for SMT.
- Kalchbrenner and Blunsom (2013). Recurrent Continuous Translation Models.
- Sutskever et al. (2014). Sequence to Sequence Learning with Neural Networks.