Machine Translation 2
Neural Machine Translation

CS 287

Review: Simple One-to-One Model

1. Language Model; words depend on previous word

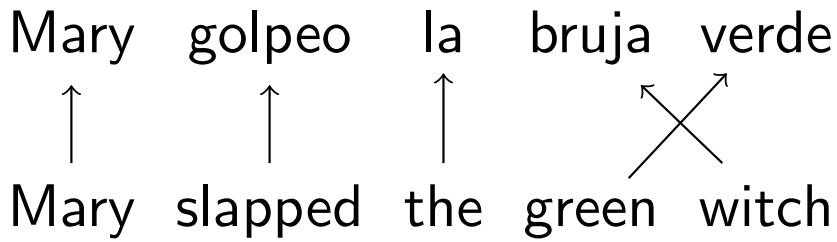
$$p(\mathbf{y}) = \prod_{i=1}^{n} p(\mathbf{y}_i | \mathbf{y}_{i-1})$$

2. Translation Model; source word depends on current position

$$p(\mathbf{x}|\mathbf{y}) = \prod_{i=1}^{n} p(\mathbf{x}_i | \mathbf{x}_{i-1})$$

Review: Alignment

- **a**; alignment mapping each target word to a source word
- Assuming one-to-one



	Mary	golpeo	la	bruja	verde
Mary	X				
slapped		X			
the			X		
green					X
witch				X	

Review: Using Alignments

$$p(\mathbf{y}|\mathbf{x}) \propto \sum_{\mathbf{a}} p(\mathbf{y})p(\mathbf{a}|\mathbf{y})p(\mathbf{x}|\mathbf{a},\mathbf{y})$$

With alignment,

$$p(\mathbf{x}|\mathbf{a},\mathbf{y}) = \prod_{i=1}^{n} p(\mathbf{x}_{a_i}|\mathbf{y}_i)$$

Sum-over-alignment approximated with a max-over-alignment,

$$\arg\max_{j,w_{1:n}^t} \prod_{i=1}^{n} p(\mathbf{x}_{a_i}|\mathbf{y}_i = w_i^t)p(\mathbf{y}_i = w_i^t|\mathbf{y}_{i-1} = w_{i-1}^t)p(a_i = c_i|a_{i-1} = c_{i-1}, i)$$

Quiz: CRF

Note: Nothing in our definition of CRFs relied on $\mathbf{y}_i$ to align with $\mathbf{x}_i$ (conditioned on full sequence).

For this quiz, imagine we have an input sequence $\mathbf{x}$, and we want to find the optimal output sequence $\mathbf{y}$ but we do not fix $n < N$. For instance finding the best word segmentation of an unsegmented input $\mathbf{x}$.

- How would you find

$$\arg\max_{n, c_{1:n}} f(\mathbf{x}, c_{1,n})?$$

- How would you train

$$f(\mathbf{x}, c_{1,n}; \theta)?$$

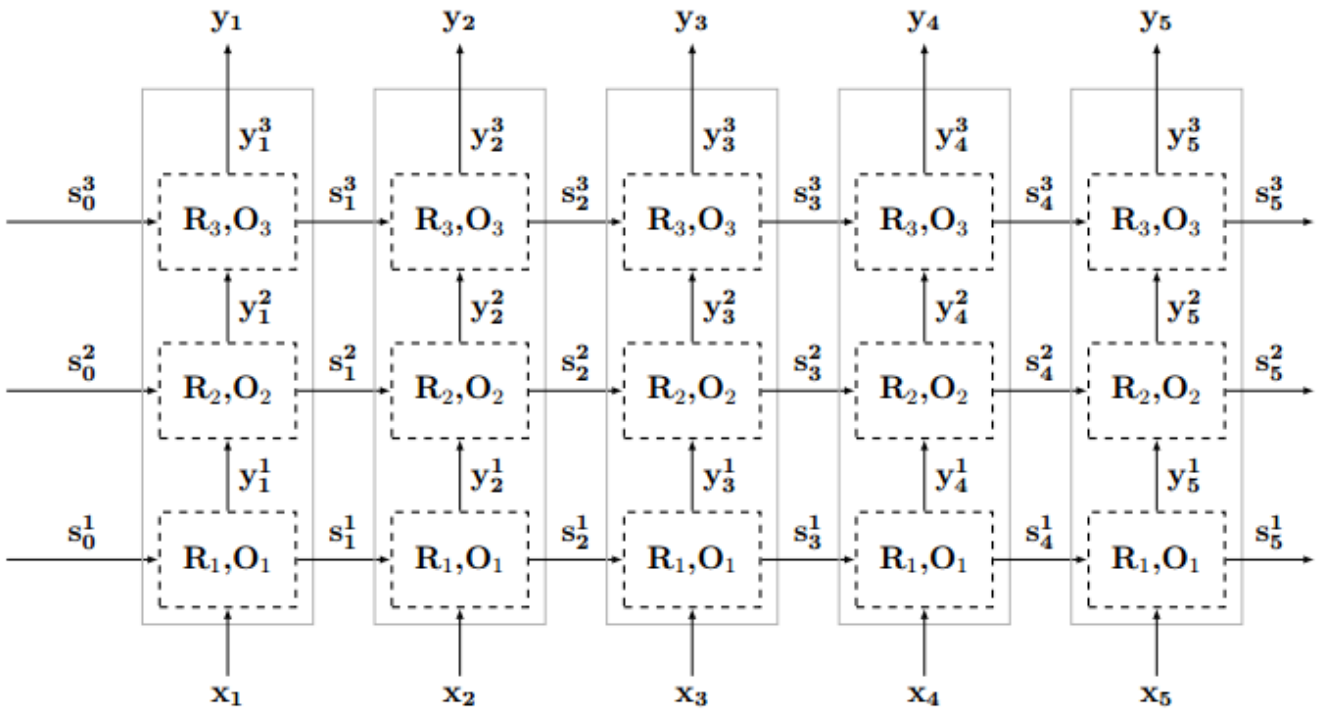
Preliminaries

- How alignment is used in SMT.
- Language Model RNNs
- Bidirectional RNNs
- Stacked RNNs

Bit-Set Beam Search

[Describe on board]

Stacked RNN


y_1 y_2 y_3 y_4 y_5
y_1^3 y_2^3 y_3^3 y_4^3 y_5^3
s_0^3 R_3,O_3 s_1^3 R_3,O_3 s_2^3 R_3,O_3 s_3^3 R_3,O_3 s_4^3 R_3,O_3 s_5^3
y_1^2 y_2^2 y_3^2 y_4^2 y_5^2
s_0^2 R_2,O_2 s_1^2 R_2,O_2 s_2^2 R_2,O_2 s_3^2 R_2,O_2 s_4^2 R_2,O_2 s_5^2
y_1^1 y_2^1 y_3^1 y_4^1 y_5^1
s_0^1 R_1,O_1 s_1^1 R_1,O_1 s_2^1 R_1,O_1 s_3^1 R_1,O_1 s_4^1 R_1,O_1 s_5^1
x_1 x_2 x_3 x_4 x_5

RNN for Language Modeling

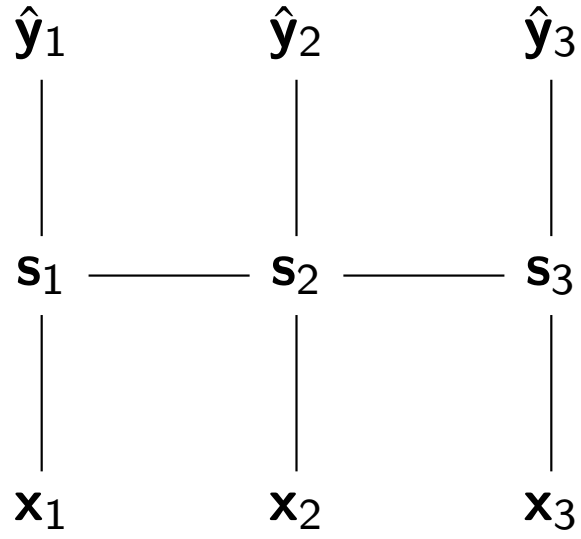
- Recent popularization of RNNs has been based on language modeling (Mikolov, 2012)
- In particular RNNs allow for non-Markovian models

$$p(w_i|w_1, \ldots, w_{i-1}; \theta) = O(\mathbf{s}_i)$$

- Compare this to the feed-forward windowed approach.

$$p(w_i|w_{i-n+1}, \ldots, w_{i-1}; \theta) = O(\mathbf{s}_i)$$

RNN as Transducer



- Can reuse hidden state each time

$$p(w_i|w_1,\ldots,w_{i-1};\theta) = O(\mathbf{s}_i) = O(R(\mathbf{s}_{i-1},\mathbf{x}_i))$$

$$p(w_{i+1}|w_1,\ldots,w_i;\theta) = O(R(\mathbf{s}_i,\mathbf{x}_{i+1}))$$

Bidirectional Models

- For all $i \in \{1, \ldots, n\}$

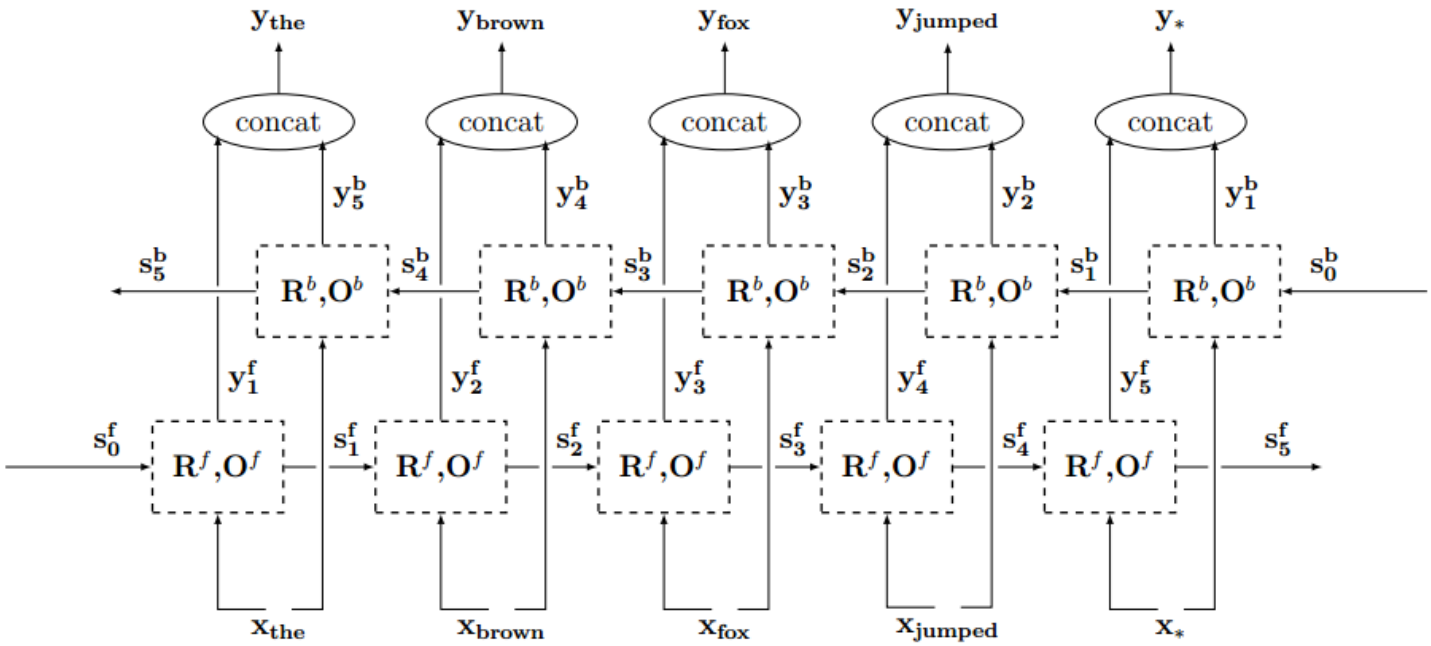
$$\mathbf{s}_i^f = R^f(\mathbf{s}_{i-1}, \mathbf{x}_i)$$

- For all $i \in \{1, \ldots, n\}$

$$\mathbf{s}_i^b = R^b(\mathbf{s}_{i+1}, \mathbf{x}_i)$$

- For all $i \in \{1, \ldots, n\}$

$$\hat{\mathbf{y}}_i = O([\mathbf{s}_i^b, \mathbf{s}_i^f]) = [\mathbf{s}_i^b, \mathbf{s}_i^f]\mathbf{W} + \mathbf{b}$$


$\mathbf{y_{the}}$
$\mathbf{y_{brown}}$
$\mathbf{y_{fox}}$
$\mathbf{y_{jumped}}$
$\mathbf{y_*}$
concat
concat
concat
concat
concat
$\mathbf{y_5^b}$
$\mathbf{y_4^b}$
$\mathbf{y_3^b}$
$\mathbf{y_2^b}$
$\mathbf{y_1^b}$
$\mathbf{s_5^b}$
$\mathbf{s_4^b}$
$\mathbf{s_3^b}$
$\mathbf{s_2^b}$
$\mathbf{s_1^b}$
$\mathbf{s_0^b}$
$\mathbf{R}^b,\mathbf{O}^b$
$\mathbf{R}^b,\mathbf{O}^b$
$\mathbf{R}^b,\mathbf{O}^b$
$\mathbf{R}^b,\mathbf{O}^b$
$\mathbf{R}^b,\mathbf{O}^b$
$\mathbf{y_1^f}$
$\mathbf{y_2^f}$
$\mathbf{y_3^f}$
$\mathbf{y_4^f}$
$\mathbf{y_5^f}$
$\mathbf{s_0^f}$
$\mathbf{s_1^f}$
$\mathbf{s_2^f}$
$\mathbf{s_3^f}$
$\mathbf{s_4^f}$
$\mathbf{s_5^f}$
$\mathbf{R}^f,\mathbf{O}^f$
$\mathbf{R}^f,\mathbf{O}^f$
$\mathbf{R}^f,\mathbf{O}^f$
$\mathbf{R}^f,\mathbf{O}^f$
$\mathbf{R}^f,\mathbf{O}^f$
$\mathbf{x_{the}}$
$\mathbf{x_{brown}}$
$\mathbf{x_{fox}}$
$\mathbf{x_{jumped}}$
$\mathbf{x_*}$

Today's Lecture

1. Sequence-to-Sequence Model (Sutskever et al 2014)
2. Attention-Based Model (Bahdanau et al 2014)

Contents

Neural Machine Translation

- $\mathbf{x} = [w_1^s \ w_2^s \ w_3^s \ldots w_n^s]$
- $\mathbf{y} = [w_1^t \ w_2^t \ w_3^t \ldots w_n^t]$

$$p(w^t|w^s) = \prod_{i=1}^{n} p(w_i^t|w^s, \theta)$$

Intuition:

1. Start with bag-of-source words
2. Pick a source word, translate it to a target word that "fits"
3. Update source bag representation.

Encoder-Decoder Idea

Compute a single vector $\mathbf{s}^t$ encoding the source.

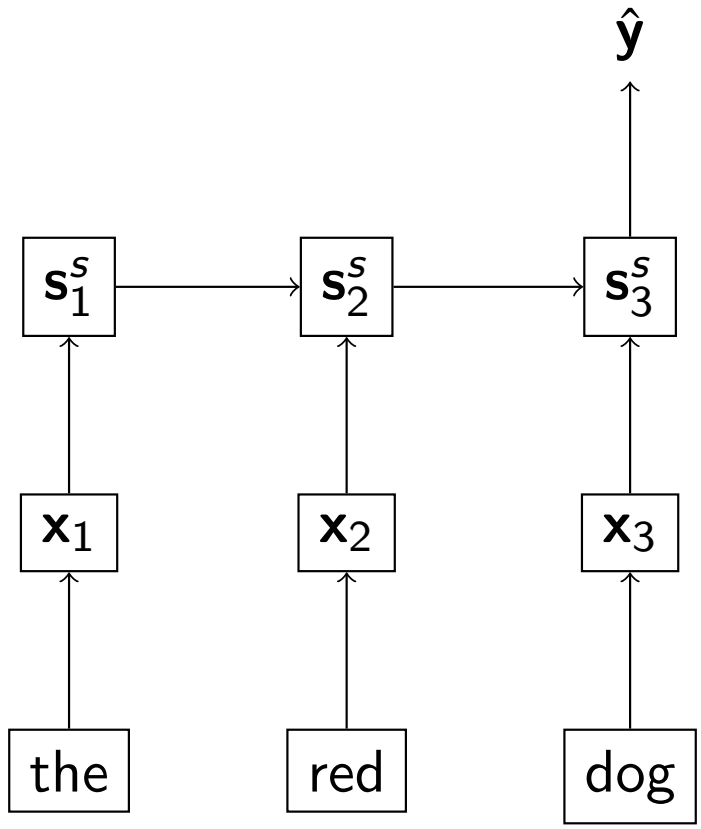
- $$p(w_i^t | w_1^t, \ldots, w_{i-1}^t, \mathbf{s}) = \hat{y}_{w_i}$$

Intuition:

- Start with encoded source
- Use encoded vector to translate to a target word that "fits"
- Update source representation

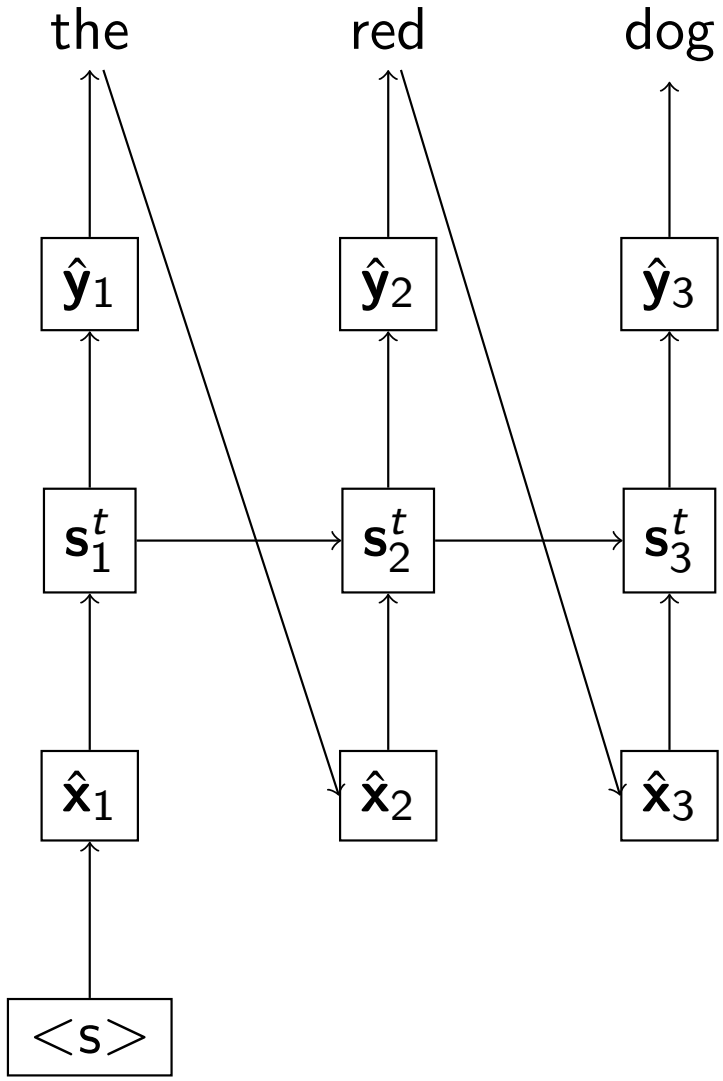
Encoder

- How do you get a source encoding? LSTM (or bi-LSTM)

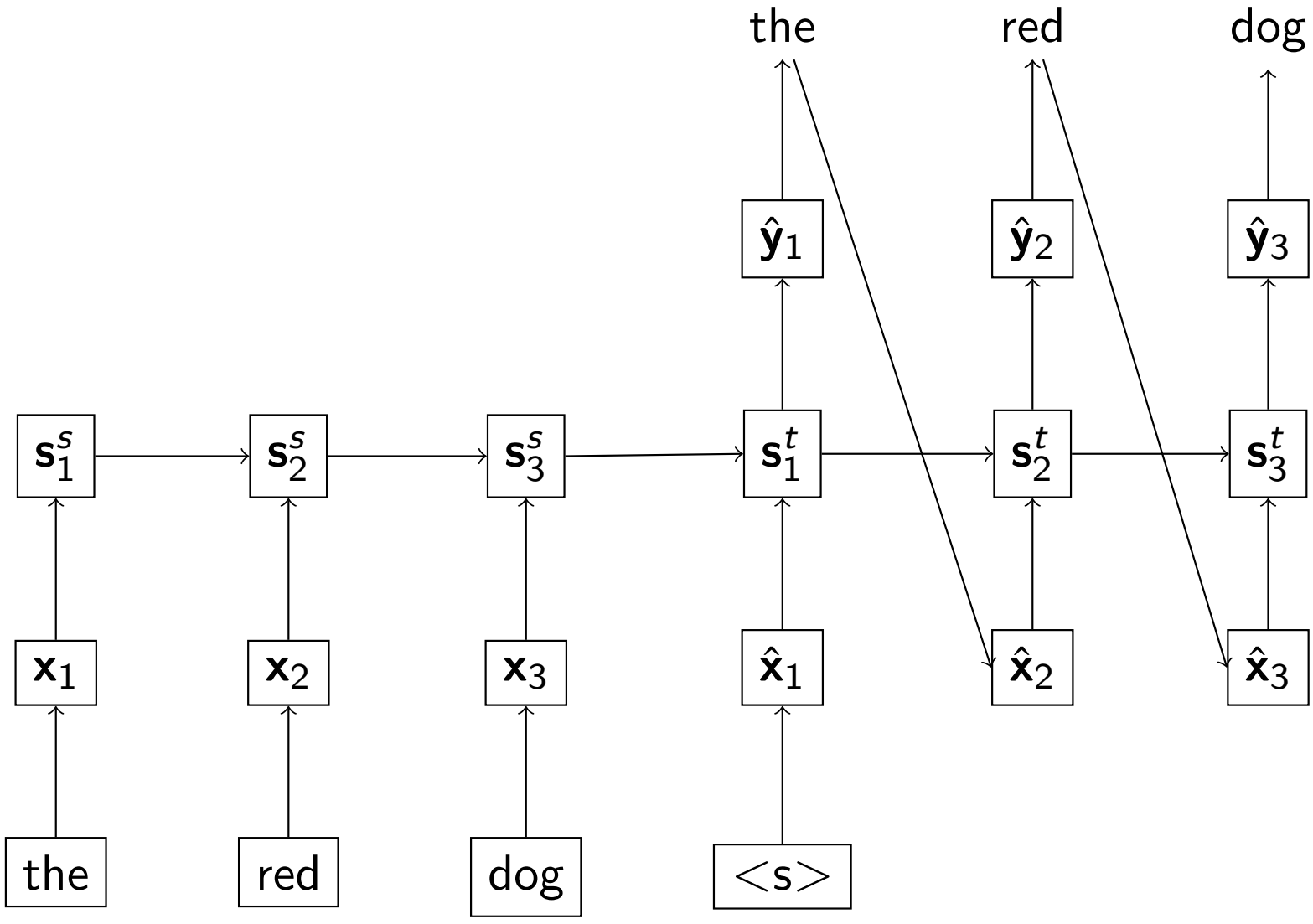


Decoder

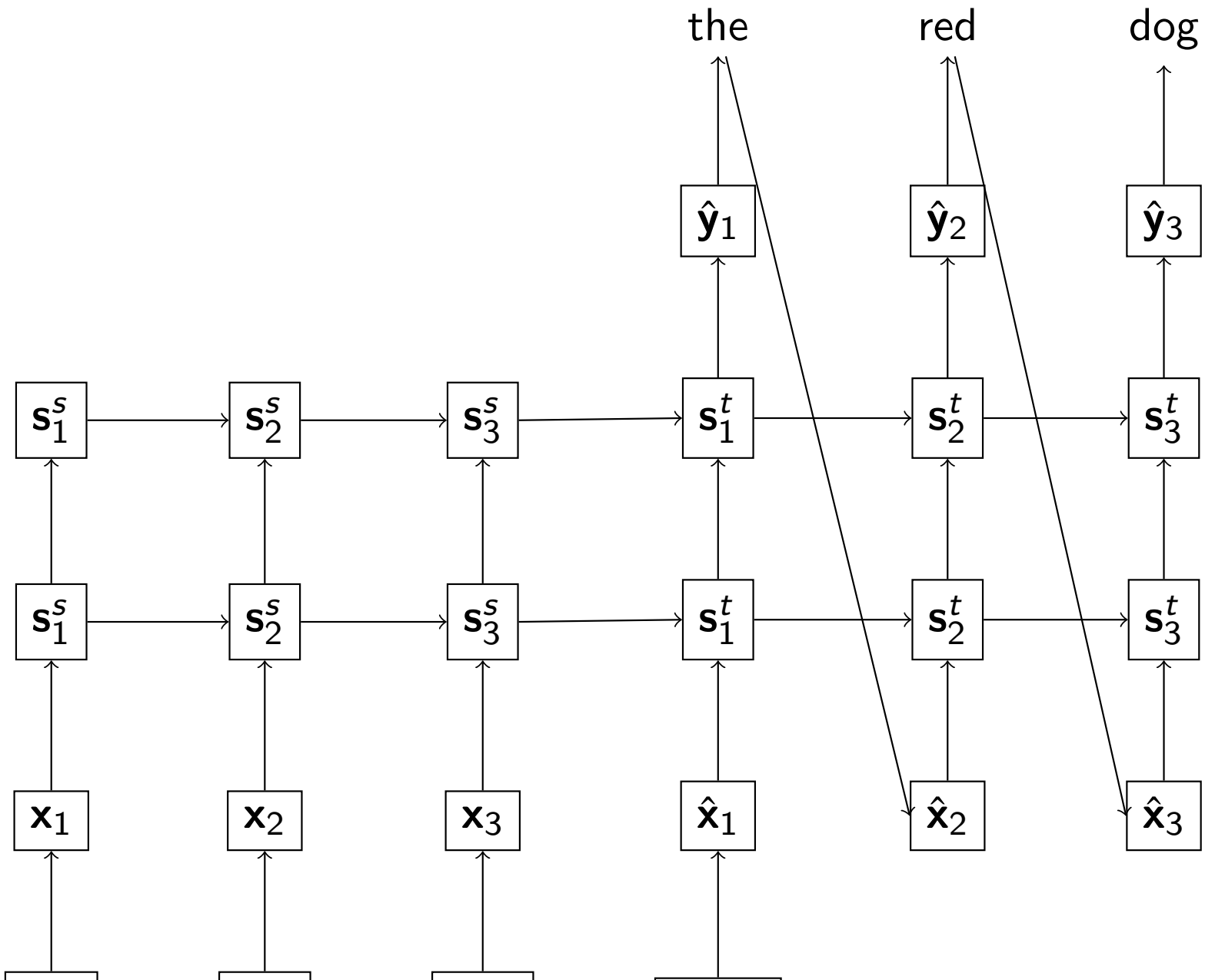
- How do you use this source encoding? LSTM



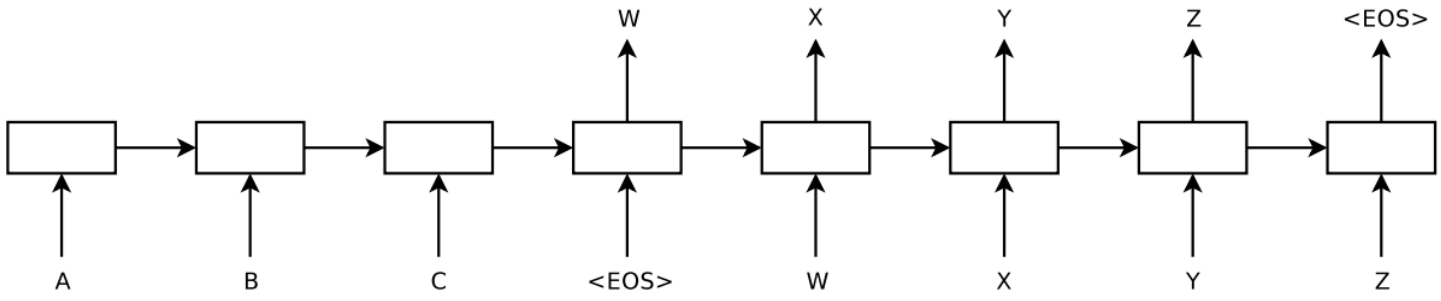
Sequence-to-Sequence



Stacked Sequence-to-Sequence



Paper



While the LSTM is capable of solving problems with long term dependencies, we discovered that the LSTM learns much better when the source sentences are reversed (the target sentences are not reversed). By doing so, the LSTMs test perplexity dropped from 5.8 to 4.7, and the test BLEU scores of its decoded translations increased from 25.9 to 30.6.

We found that the LSTM models are fairly easy to train. We used deep LSTMs with 4 layers, with 1000 cells at each layer and 1000 dimensional word embeddings, with an input vocabulary of 160,000 and an output vocabulary of 80,000. We found deep LSTMs to significantly outperform shallow LSTMs, where each additional layer reduced perplexity by nearly 10%, possibly due to their much larger hidden state. We used a naive softmax over 80,000 words at each output. The resulting LSTM has 380M parameters of which 64M are pure recurrent connections (32M for the encoder LSTM and 32M for the decoder LSTM). The complete training details are given below:

We search for the most likely translation using a simple left-to-right beam search decoder which maintains a small number B of partial hypotheses, where a partial hypothesis is a prefix of some translation. At each timestep we extend each partial hypothesis in the beam with every possible word in the vocabulary. This greatly increases the number of the hypotheses so we discard all but the B most likely hypotheses according to the models log probability. As soon as the EOS symbol is appended to a hypothesis, it is removed from the beam and is added to the set of complete hypotheses. While this decoder is approximate, it is simple to implement. Interestingly, our system performs well even with a beam size of 1, and a beam of size 2 provides most of the benefits of beam search (Table 1).

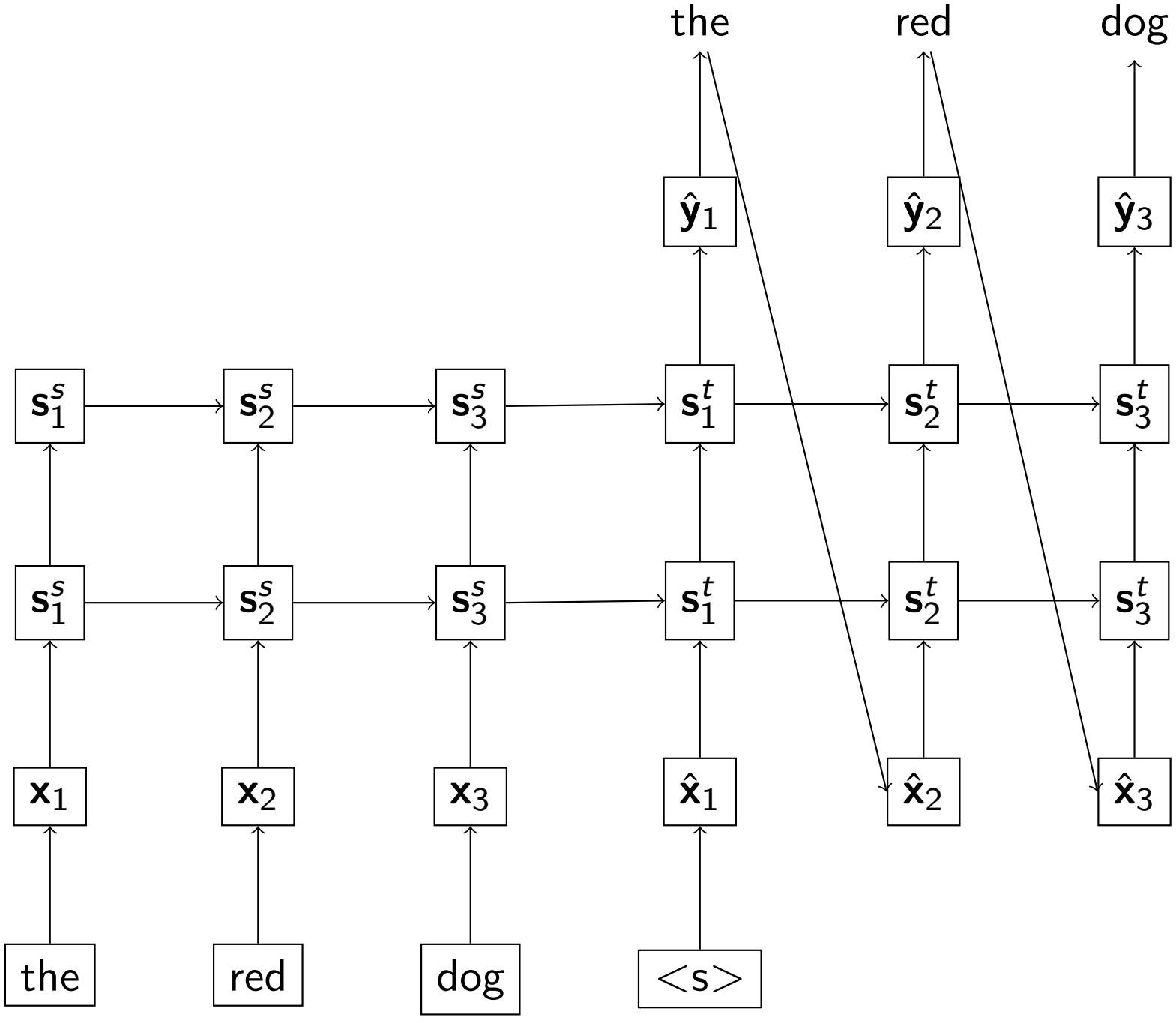
Paper

Method	test BLEU score (ntst14)
Baseline System [29]	33.30
Cho et al. [5]	34.54
State of the art [9]	**37.0**
Rescoring the baseline 1000-best with a single forward LSTM	35.61
Rescoring the baseline 1000-best with a single reversed LSTM	35.85
Rescoring the baseline 1000-best with an ensemble of 5 reversed LSTMs	**36.5**
Oracle Rescoring of the Baseline 1000-best lists	~45

Contents

Sequence-To-Sequence

- True Encoder-Decoder stores full sentence in single vector.
- In practice we have access to the original at decoding time.


the
red
dog
$\hat{\mathbf{y}}_1$
$\hat{\mathbf{y}}_2$
$\hat{\mathbf{y}}_3$
$\mathbf{s}_1^s$
$\mathbf{s}_2^s$
$\mathbf{s}_3^s$
$\mathbf{s}_1^t$
$\mathbf{s}_2^t$
$\mathbf{s}_3^t$
$\mathbf{s}_1^s$
$\mathbf{s}_2^s$
$\mathbf{s}_3^s$
$\mathbf{s}_1^t$
$\mathbf{s}_2^t$
$\mathbf{s}_3^t$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\hat{\mathbf{x}}_1$
$\hat{\mathbf{x}}_2$
$\hat{\mathbf{x}}_3$
the
red
dog
<s>

Alignment

1. Start with bag-of-source words
2. Pick a **aligned** source word, translate it to a target word that "fits"
3. Update source bag representation.

	Mary	golpeo	la	bruja	verde
Mary	X				
slapped		X			
the			X		
green					X
witch				X	

Hard Alignments

- Have source vectors $\mathbf{s}_j^t$
- Could find greedy alignment at each step.
- Idea: learn scoring MLP, and use best aligned vector.
- At generation position i, For all source positions j,

$$z_j = \tanh([\mathbf{s}_i^t, \mathbf{s}_j^s]\mathbf{W} + \mathbf{b})$$

$$j = \arg\max_j z_j$$

$$w_{i+1}^t = \arg\max_w O(w, \mathbf{s}_{i+1}^t, \mathbf{s}_j^s)$$

What is the issue here?

Attention: Soft Alignments

At generation position i, For all source positions j,

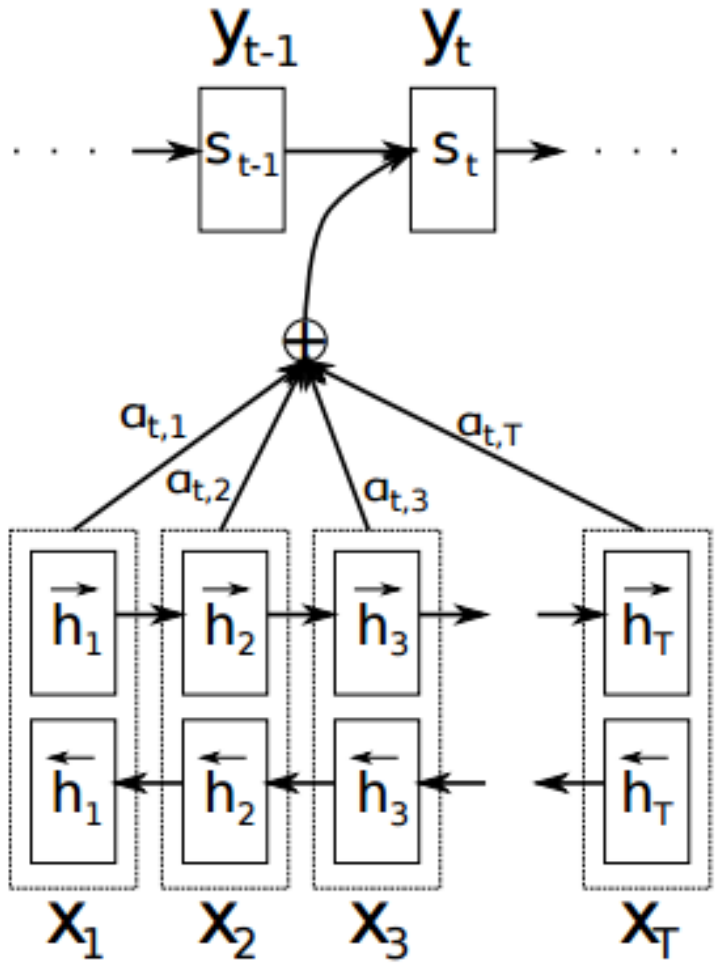
$$z_j = \tanh([\mathbf{s}_i^t, \mathbf{s}_j^s]\mathbf{W} + \mathbf{b})$$

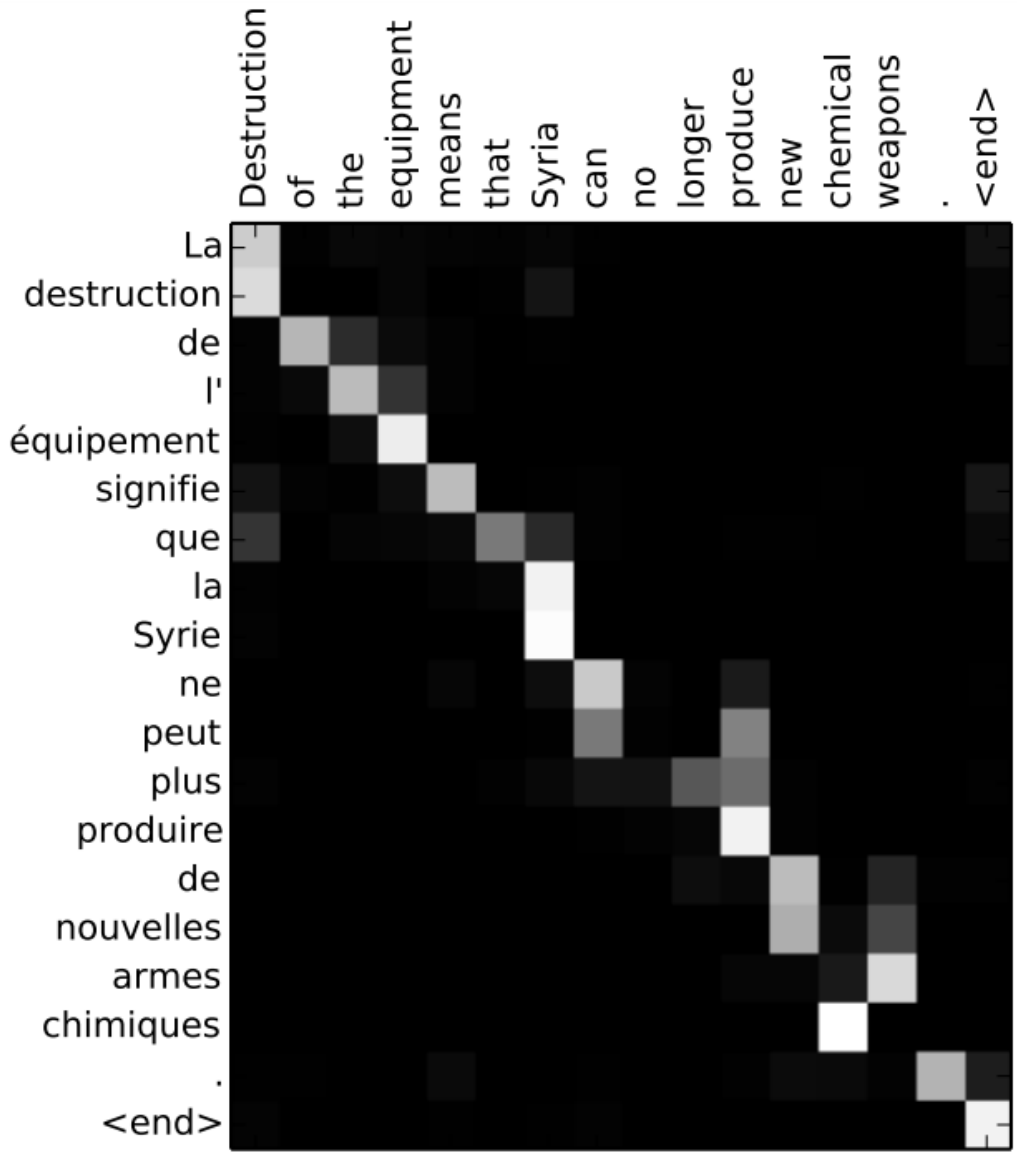
$$\boldsymbol{\alpha} = \text{softmax}(\mathbf{z})$$

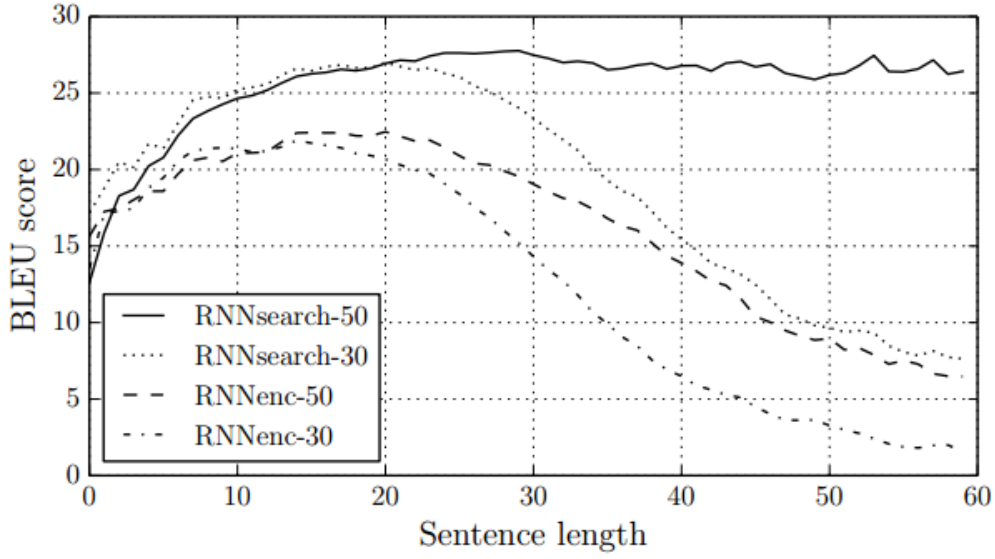
$$\mathbf{c} = \sum_j \alpha_j \mathbf{s}_j^s$$

$$w_{i+1}^t = \arg\max_w O(w, \mathbf{s}_{i+1}^t, \mathbf{c})$$

- α_j is the "attention" paid to source position j


y_{t-1}
y_t
s_{t-1}
s_t
$\alpha_{t,1}$
$\alpha_{t,2}$
$\alpha_{t,3}$
$\alpha_{t,T}$
$\overrightarrow{h}_1$
$\overrightarrow{h}_2$
$\overrightarrow{h}_3$
$\overrightarrow{h}_T$
$\overleftarrow{h}_1$
$\overleftarrow{h}_2$
$\overleftarrow{h}_3$
$\overleftarrow{h}_T$
x_1
x_2
x_3
x_T


Destruction
of
the
equipment
means
that
Syria
can
no
longer
produce
new
chemical
weapons
.
<end>
La
destruction
de
l'
équipement
signifie
que
la
Syrie
ne
peut
plus
produire
de
nouvelles
armes
chimiques
.
<end>


30
25
20
15
10
5
0
BLEU score
0
10
20
30
40
50
60
Sentence length
RNNsearch-50
RNNsearch-30
RNNenc-50
RNNenc-30

While the LSTM is capable of solving problems with long term dependencies, we discovered that the LSTM learns much better when the source sentences are reversed (the target sentences are not reversed). By doing so, the LSTMs test perplexity dropped from 5.8 to 4.7, and the test BLEU scores of its decoded translations increased from 25.9 to 30.6.

We use a minibatch stochastic gradient descent (SGD) algorithm together with Adadelta (Zeiler, 2012) to train each model. Each SGD update direction is computed using a minibatch of 80 sentences. We trained each model for approximately 5 days. Once a model is trained, we use a beam search to find a translation that approximately maximizes the conditional probability (see, e.g., Graves, 2012; Boulanger-Lewandowski et al., 2013). Sutskever et al. (2014) used this approach to generate translations from their neural machine translation model.

WMT 14 contains the following English-French parallel corpora: Europarl (61M words), news commentary (5.5M), UN (421M) and two crawled corpora of 90M and 272.5M words respectively, totaling 850M words. Following the procedure described in Cho et al. (2014a), we reduce the size of the combined corpus to have 348M words using the data selection method by Axelrod et al. (2011).5 We do not use any monolingual data other than the mentioned parallel corpora, although it may be possible to use a much larger monolingual corpus to pretrain an encoder.