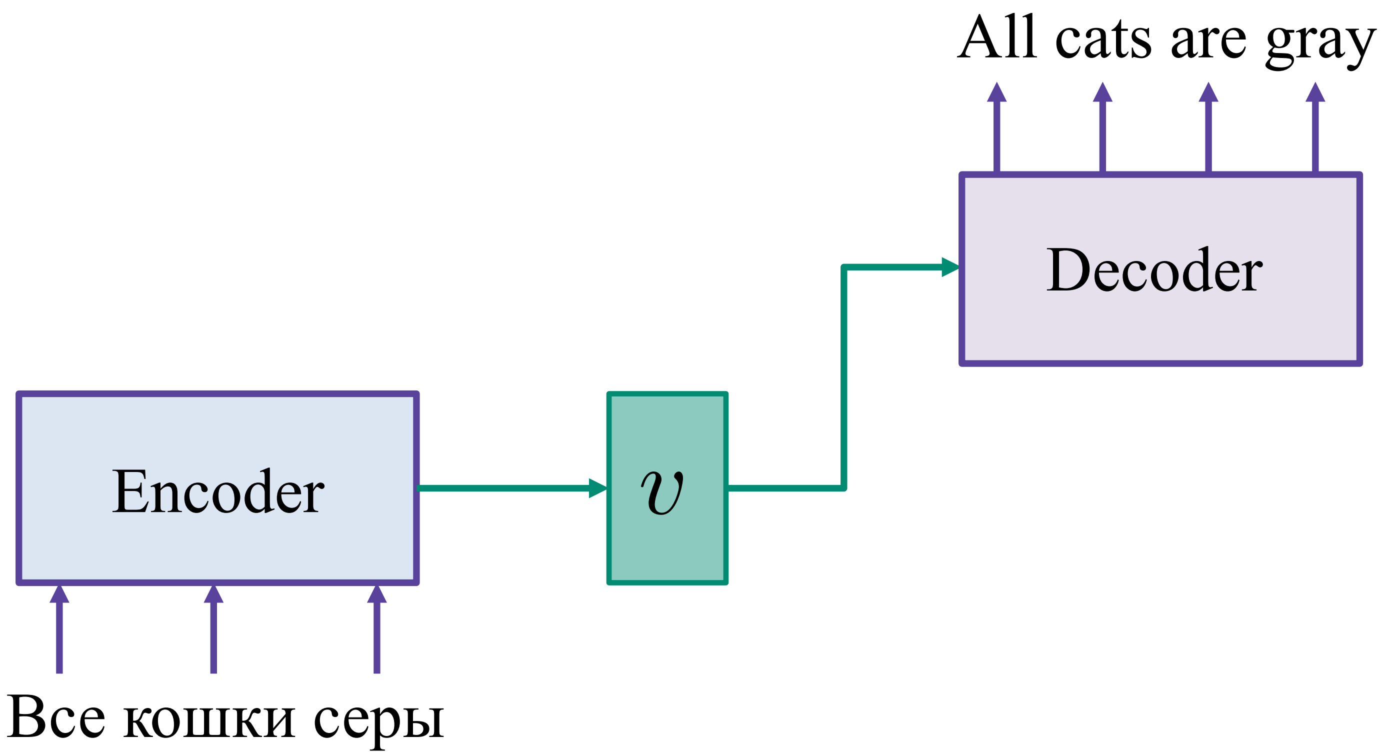
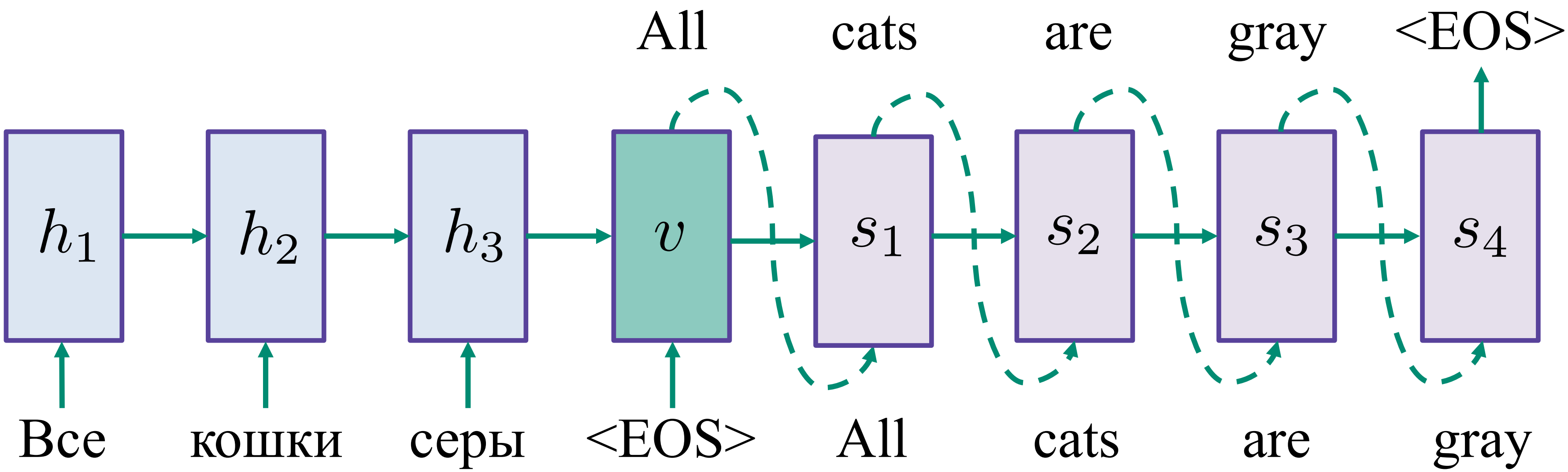
Encoder-decoder architecture

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Sequence to sequence



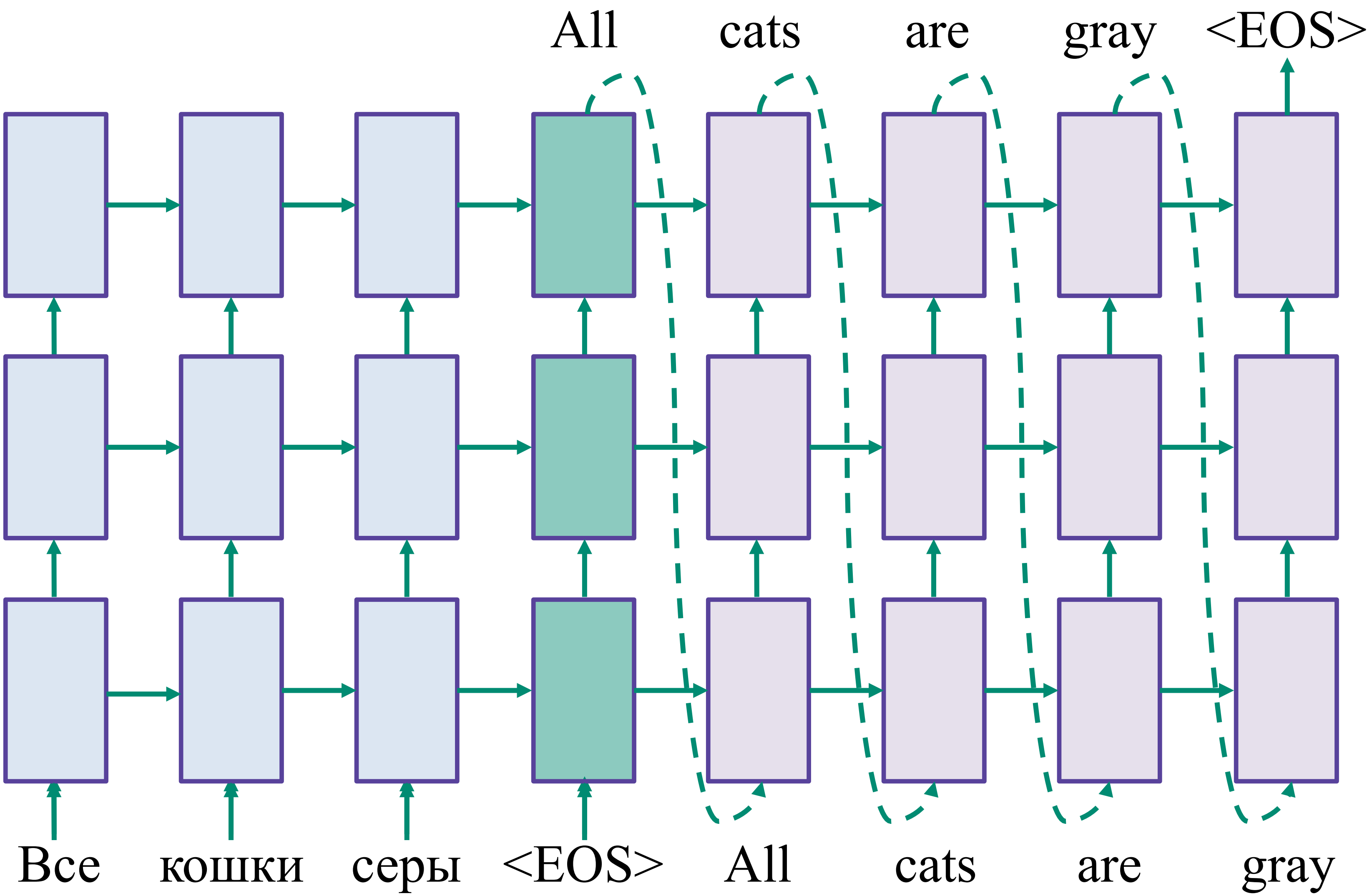
Sequence to sequence



Ilya Sutskever, Oriol Vinyals, Quoc V. Le. Sequence to Sequence Learning with Neural Network, 2014.

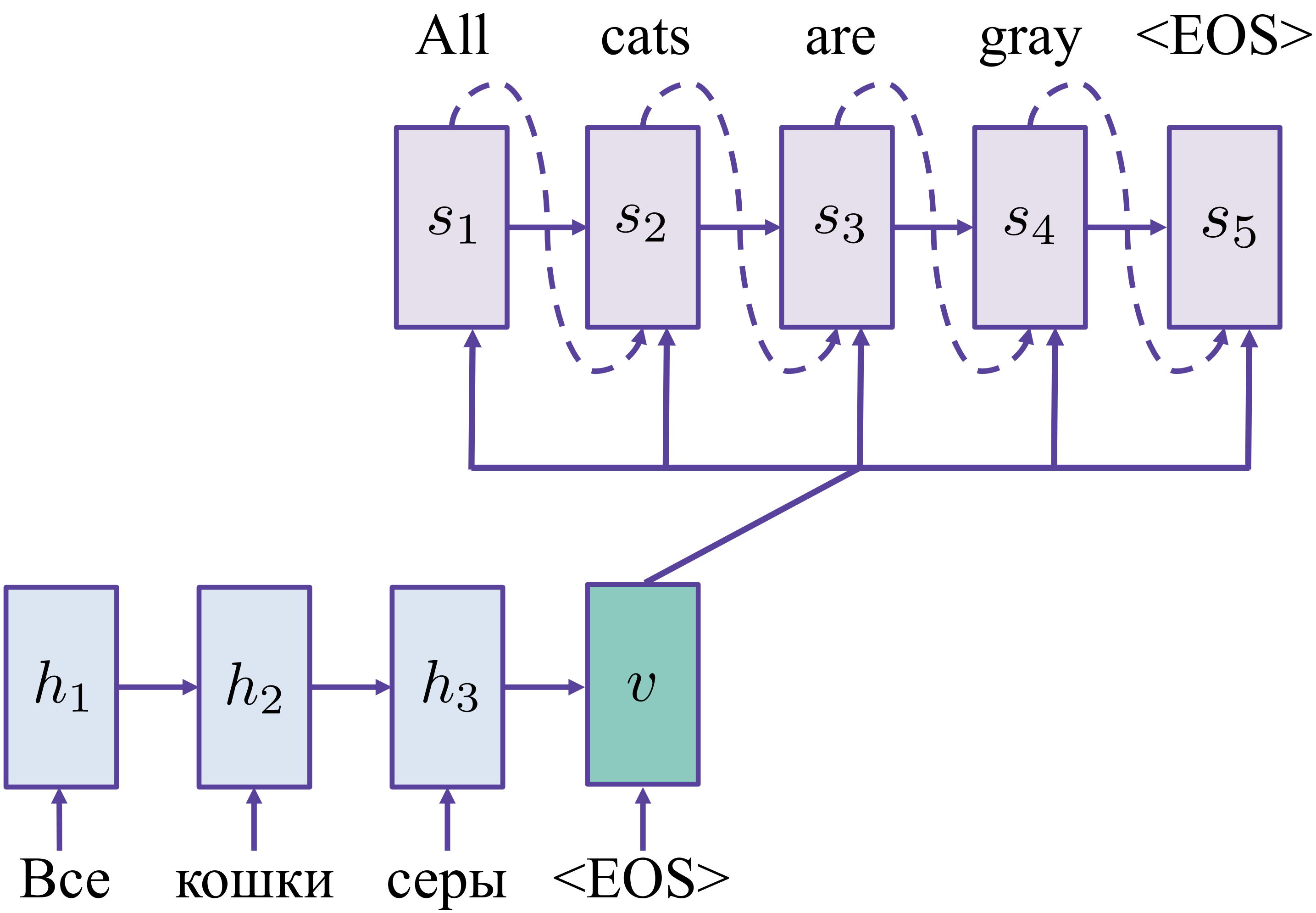
Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Sequence to sequence



Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Sequence to sequence



Cho et. al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, 2014.

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Sequence to sequence

$$p(y_1, \dots y_J | x_1, \dots x_I) = \prod_{j=1}^{J} p(y_j | v, y_1, \dots y_{j-1})$$

- **Encoder:** maps the source sequence to the hidden vector

RNN: $h_i = f(h_{i-1}, x_i)$ $\qquad v = h_I$

- **Decoder:** performs language modeling given this vector

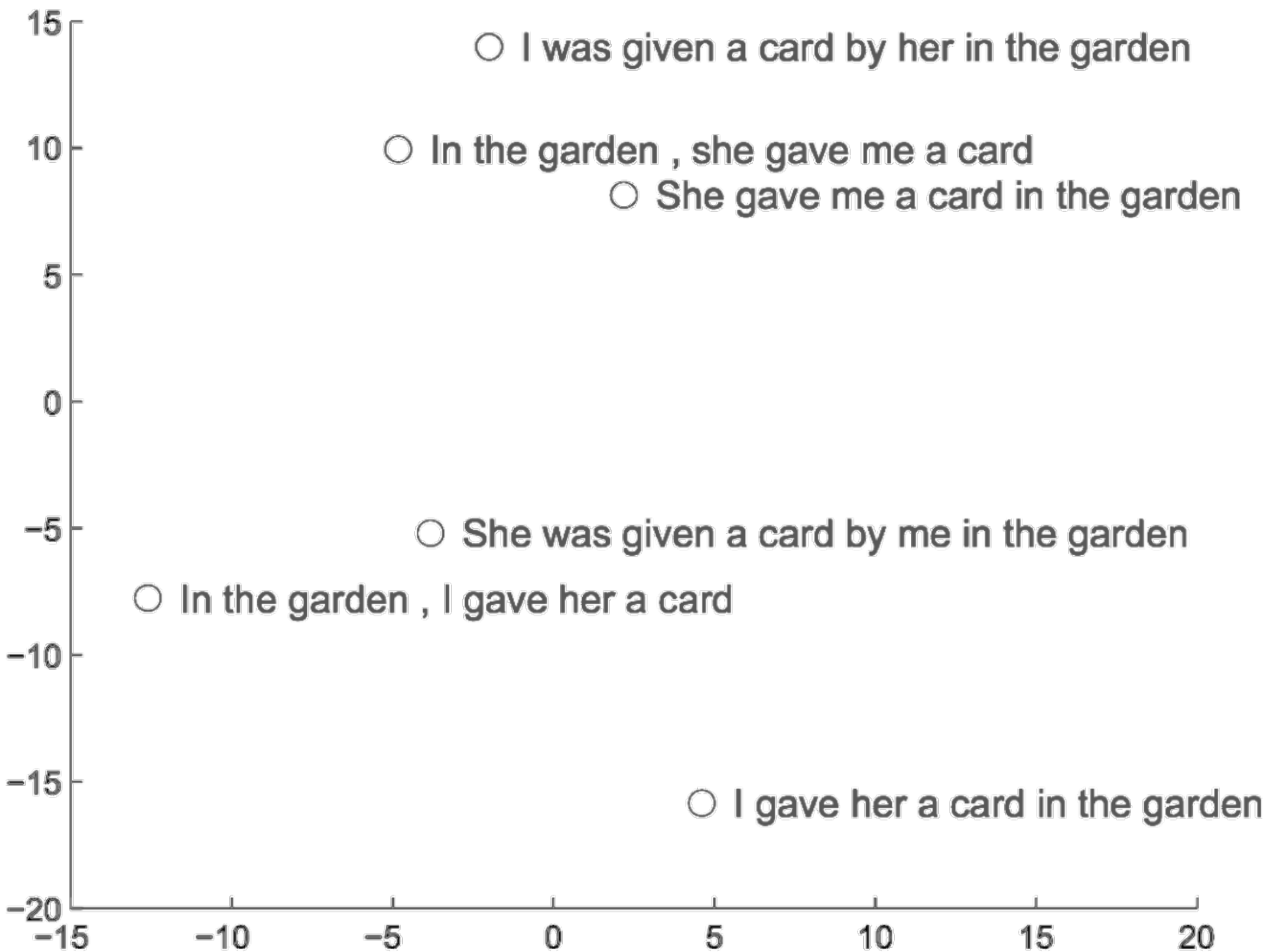
RNN: $s_j = g(s_{j-1}, [y_{j-1}, v])$

- **Prediction** (the simplest way):

$$p(y_j | v, y_1, \dots y_{j-1}) = softmax\,(U s_j + b)$$

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

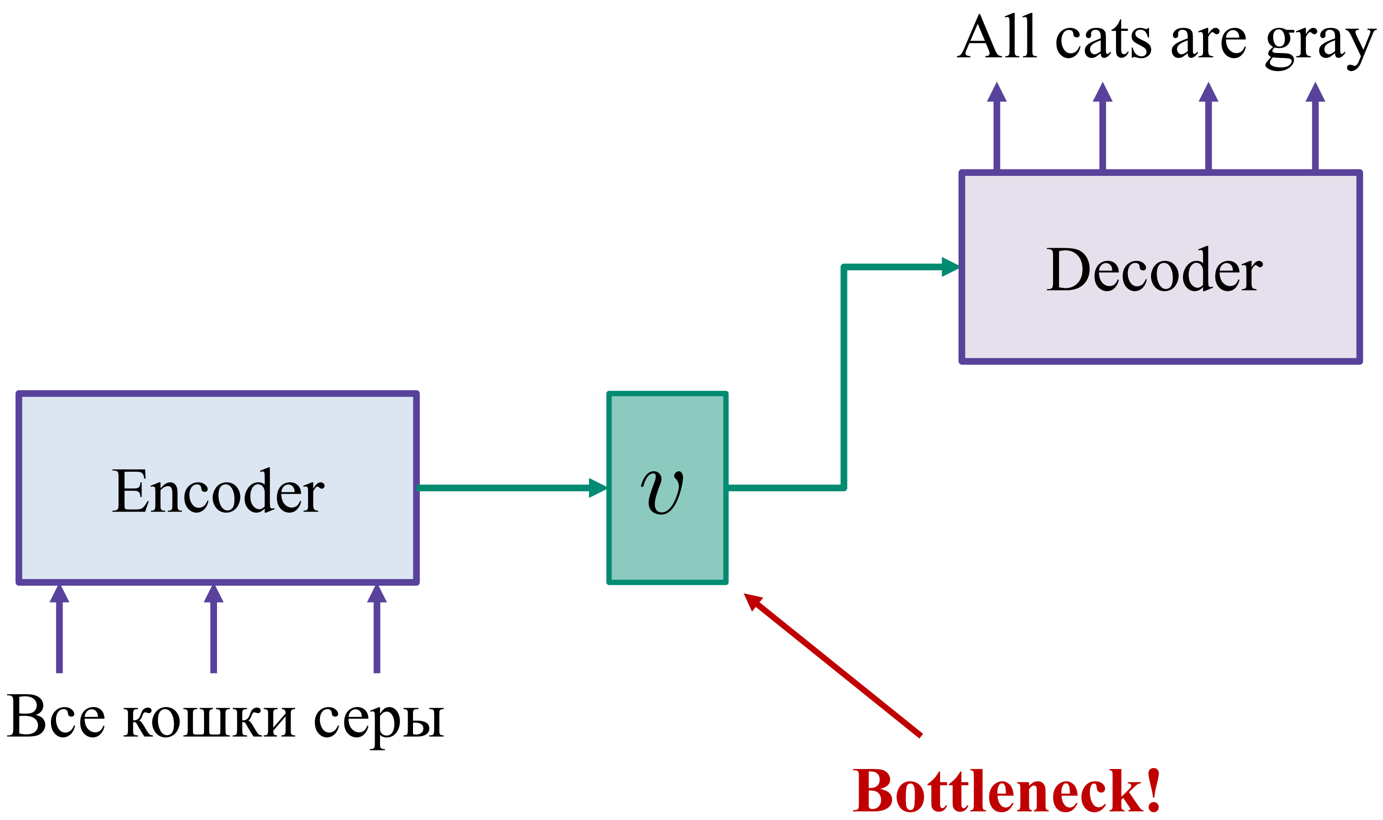
Hidden representations are good…



Ilya Sutskever, Oriol Vinyals, Quoc V. Le. Sequence to Sequence Learning with Neural Network, 2014.

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

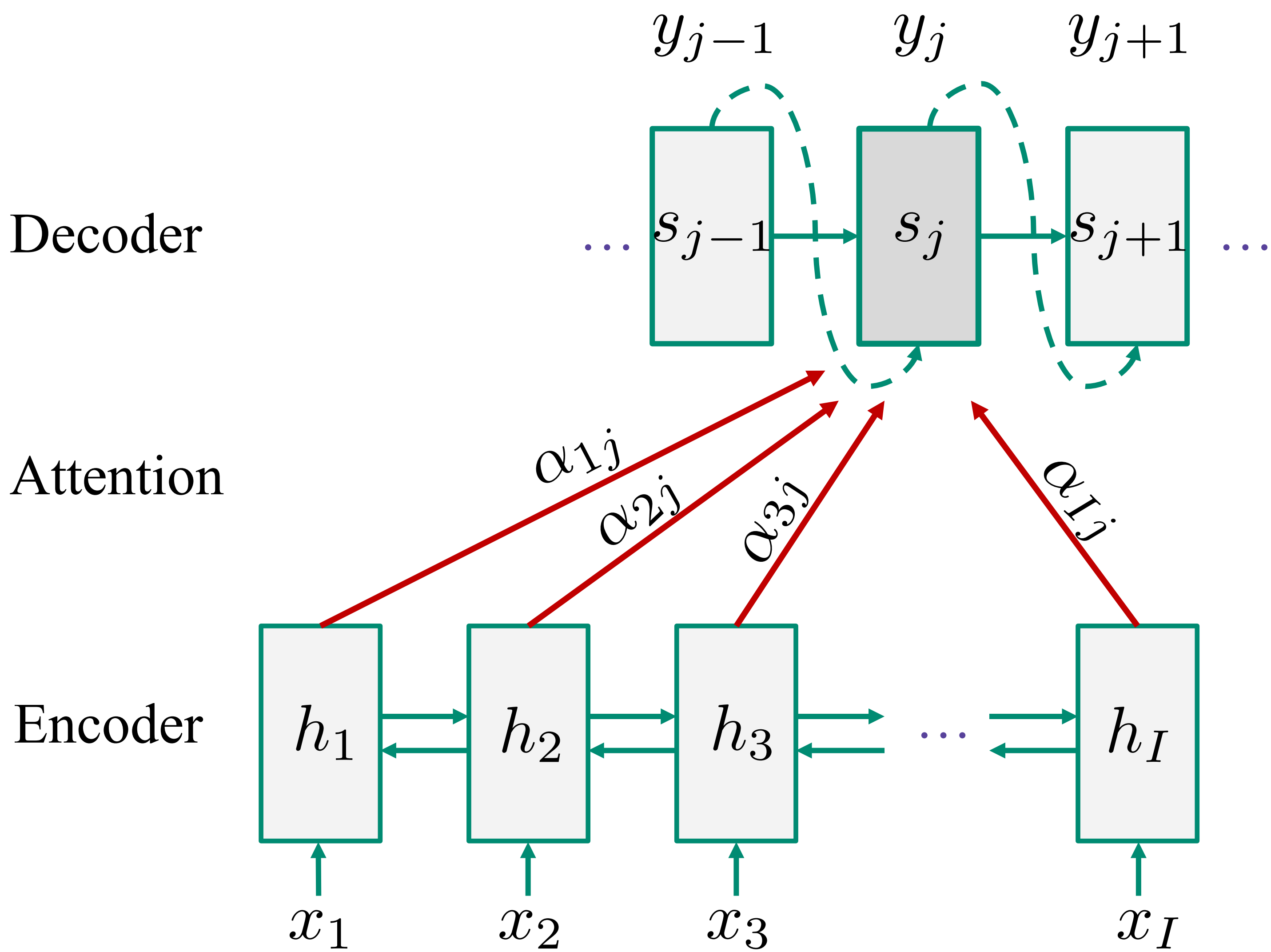
… but still a bottleneck



Attention mechanism

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Attention mechanism



Bahdanau et. al - Neural Machine Translation by jointly learning to align and translate, 2015.

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Attention mechanism

- Encoder states are weighted to obtain the representation relevant to the decoder state:

$$v_j = \sum_{i=1}^{I} \alpha_{ij} h_i$$

- The weights are learnt and should find the most relevant encoder positions:

$$\alpha_{ij} = \frac{\exp(sim(h_i, s_{j-1}))}{\sum_{i'=1}^{I} \exp(sim(h_{i'}, s_{j-1}))}$$

How to compute attention weights?

- **Additive attention:**

$$sim(h_i, s_j) = w^T \tanh(W_h h_i + W_s s_j)$$

- **Multiplicative attention:**

$$sim(h_i, s_j) = h_i^T W s_j$$

- **Dot product also works:**

$$sim(h_i, s_j) = h_i^T s_j$$

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Put all together

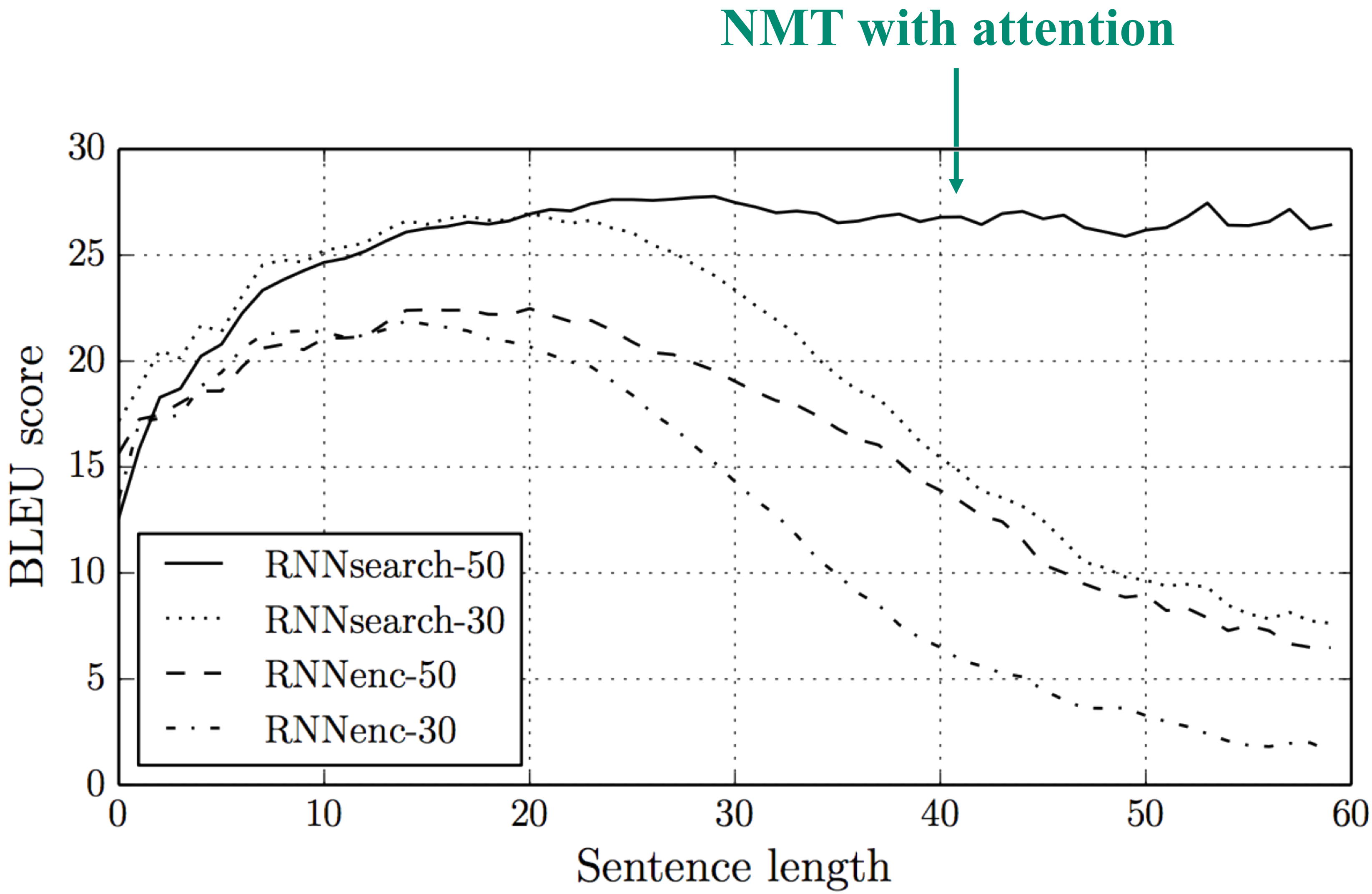
$$p(y_1, \dots y_J | x_1, \dots x_I) = \prod_{j=1}^{J} p(y_j | v_j, y_1, \dots y_{j-1})$$

- Still encoder-decoder architecture with RNNs:

$$h_i = f(h_{i-1}, x_i) \qquad\qquad s_j = g(s_{j-1}, [y_{j-1}, v_j])$$

- But the source representations differ for each position j of the decoder.

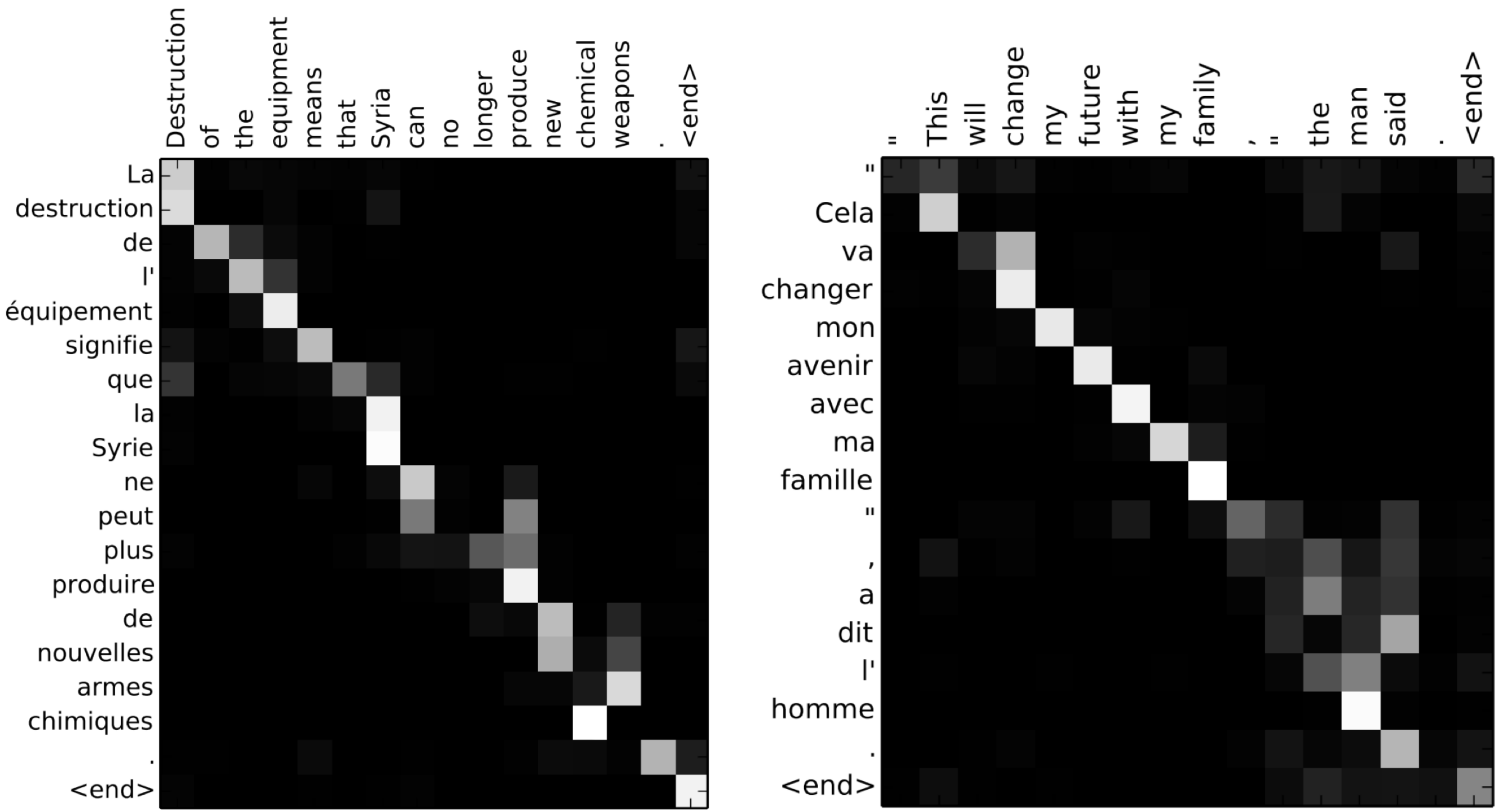
Helps for long sentences



Bahdanau et. al. Neural Machine Translation by jointly learning to align and translate, 2015.

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Example: attention (alignments)



Bahdanau et. al. Neural Machine Translation by jointly learning to align and translate, 2015.

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Is the attention similar to what humans do?

- *For humans:* **saves time**

Attention saves time when reading (i.e. we look only to the relevant parts of the sentence).

- *For machines:* **wastes time**

To compute the attention weights, the model carefully examines ALL the positions, thus wastes even more time.

Local attention

1. **Find the most relevant position $\mathbf{a_j}$ in the source**
   - Monotonic alignments: $a_j = j$
   - Predictive alignments: $a_j = I \cdot \sigma(b^T \tanh(W s_j))$

2. **Attend only positions within a window $[\mathbf{a_j - h; a_j + h}]$**
   - Compute scores as usual
   - Probably multiply by a Gaussian centered in a_j

Luong et. al. Effective Approaches to Attention-based Neural Machine Translation, 2015.

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Global vs local attention

System	Perplexity	BLEU
global (location)	6.4	19.3
global (dot)	6.1	20.5
global (mult)	6.1	19.5
local-m (dot)	>7.0	x
local-m (mult)	6.2	20.4
local-p (dot)	6.6	19.6
local-p (mult)	**5.9**	**20.9**

Luong et. al. Effective Approaches to Attention-based Neural Machine Translation, 2015.

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Global vs local attention

	System	Perplexity	BLEU
$W s_j$	global (location)	6.4	19.3
$h_i^T s_j$	global (dot)	6.1	20.5
$h_i^T W s_j$	global (mult)	6.1	19.5
	local-m (dot)	>7.0	x
	local-m (mult)	6.2	20.4
	local-p (dot)	6.6	19.6
	local-p (mult)	**5.9**	**20.9**

Luong et. al. Effective Approaches to Attention-based Neural Machine Translation, 2015.

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

How to deal with a vocabulary?

Outline

- Computing *softmax* for a large vocabulary is slow!
    - Hierarchical softmax
- Even a large vocabulary has *OOV words*:
    - Copy mechanism
    - Sub-word modeling
        - Word-character hybrid models
        - Byte-pair encoding

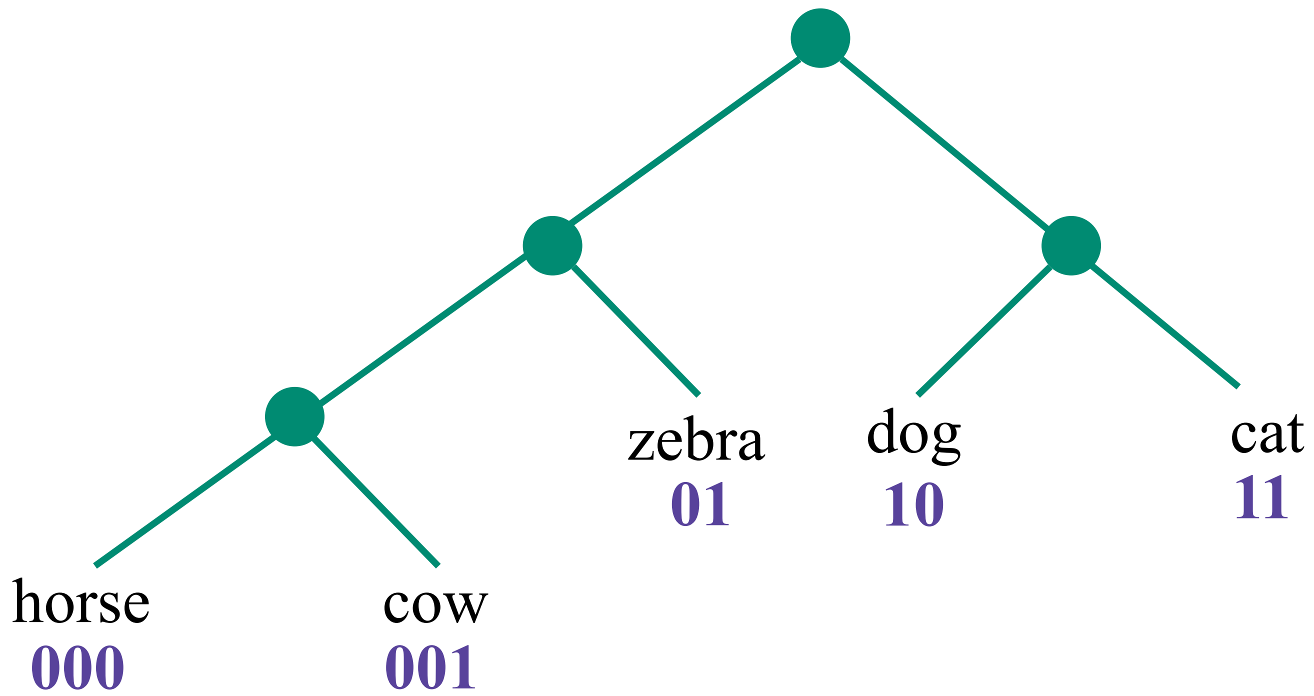
Outline

- Computing *softmax* for a large vocabulary is slow!
  - **Hierarchical softmax**
- Even a large vocabulary has *OOV words*:
  - Copy mechanism
  - Sub-word modeling
    - Word-character hybrid models
    - Byte-pair encoding

Hierarchical softmax

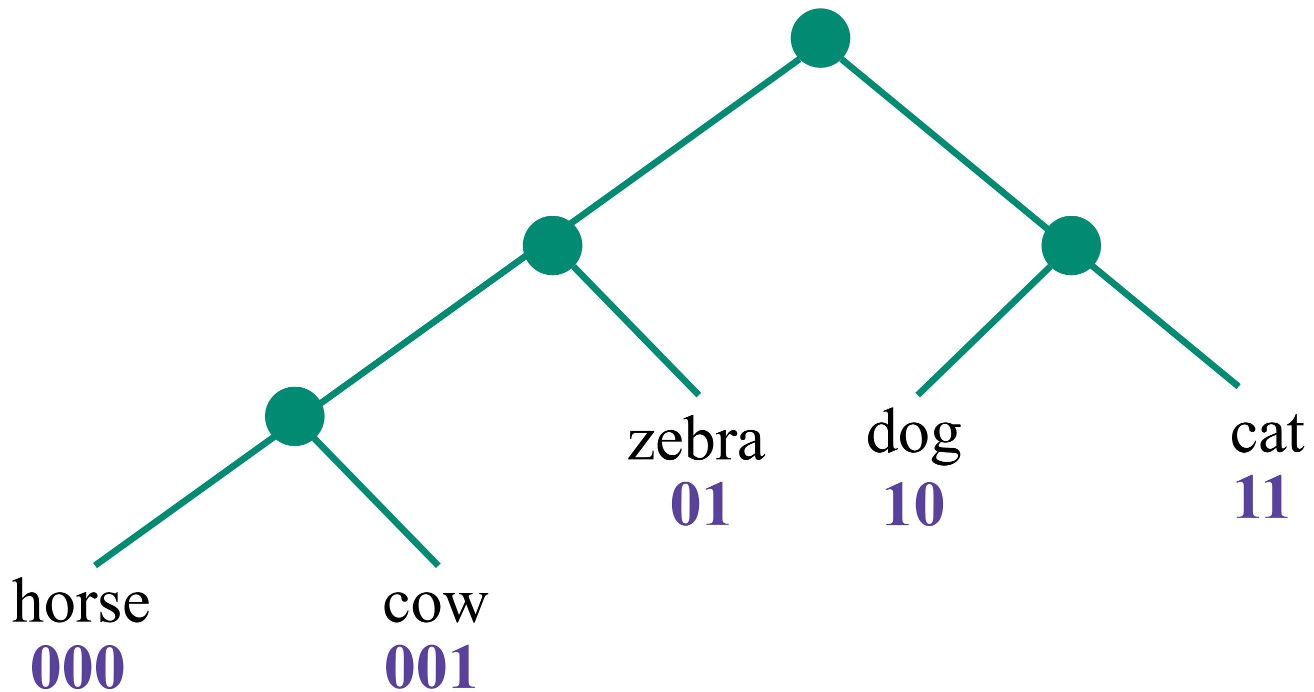
Each word is uniquely represented by a binary code:

- 0 means “go left”, 1 means “go right”



Hierarchical softmax

E.g. for **zebra** the code is $d = (\underset{d_1}{0}, \underset{d_2}{1})$



Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Scaling softmax

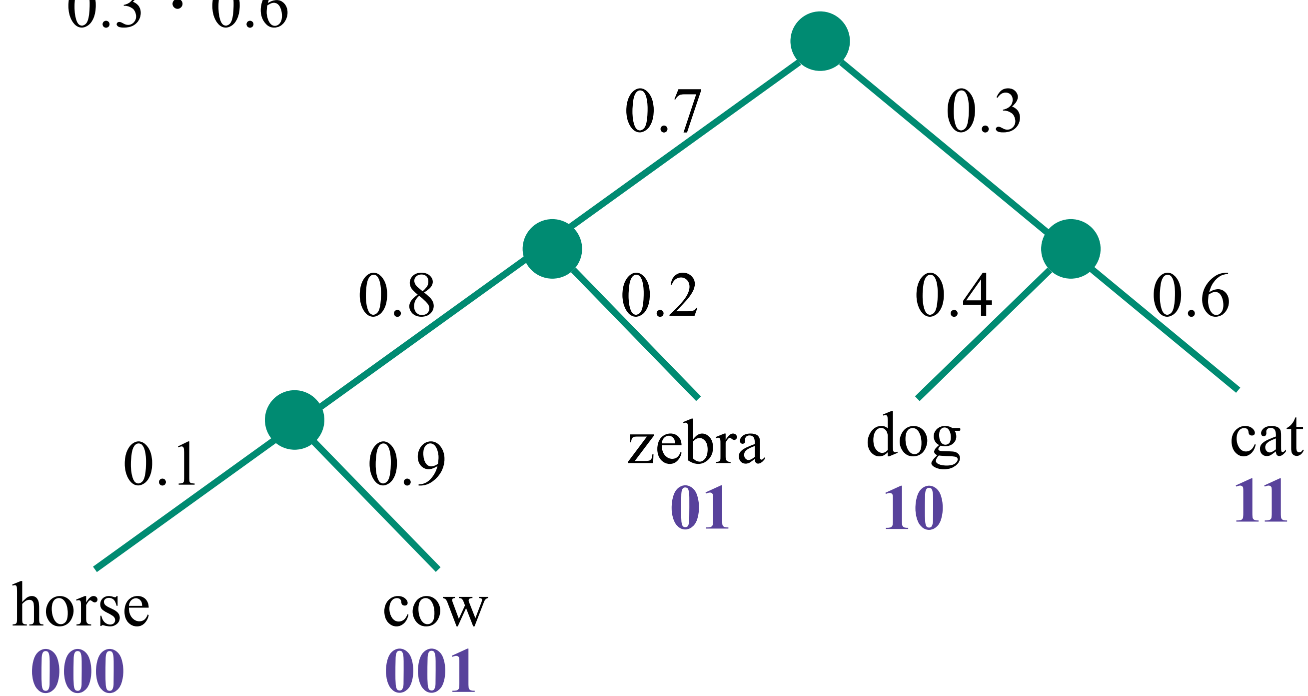
Express the probability of a word (zebra) as a product of probabilities of the binary decisions along the path (d_1, d_2).

$$p(w_n = w | w_1^{n-1}) = \prod_i p(d_i | w_1^{n-1})$$

Do you believe that it sums to 1?

Hierarchical softmax

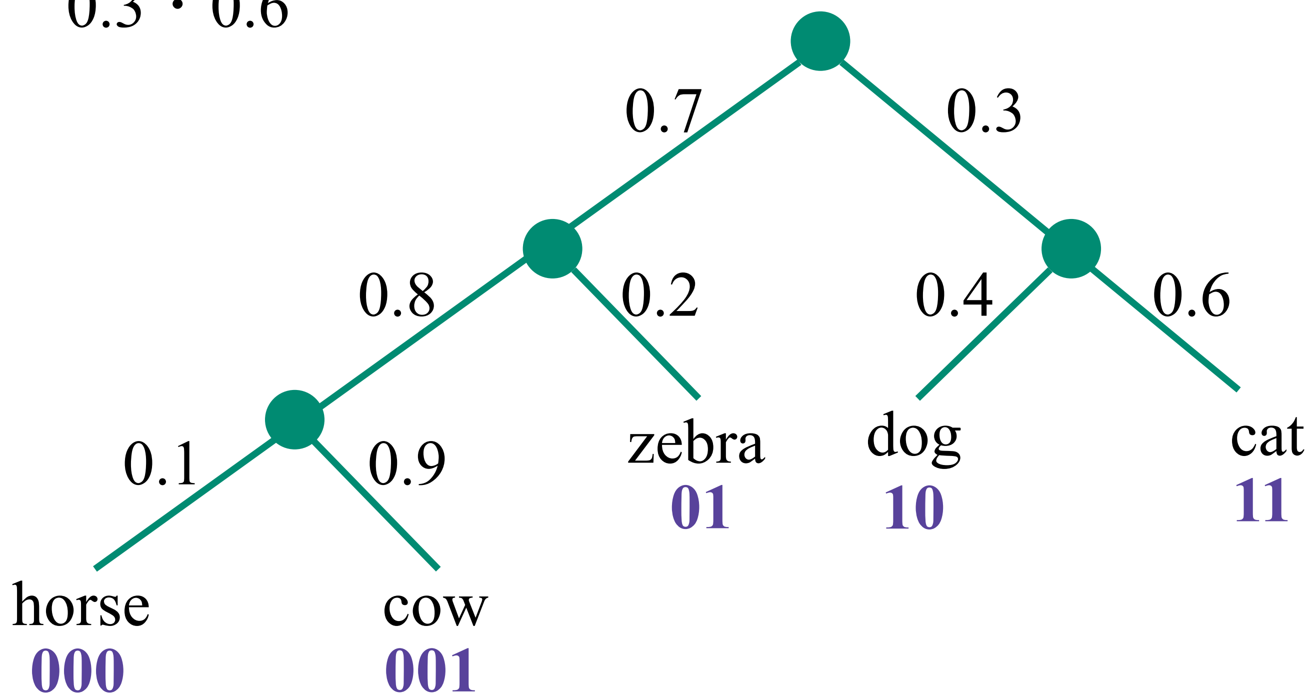
$$
\begin{array}{ll}
 & 0.7 \cdot 0.8 \cdot 0.1 \\
 & 0.7 \cdot 0.8 \cdot 0.9 \\
+ & 0.7 \cdot 0.2 \\
 & 0.3 \cdot 0.4 \\
 & 0.3 \cdot 0.6
\end{array}
$$



Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Hierarchical softmax

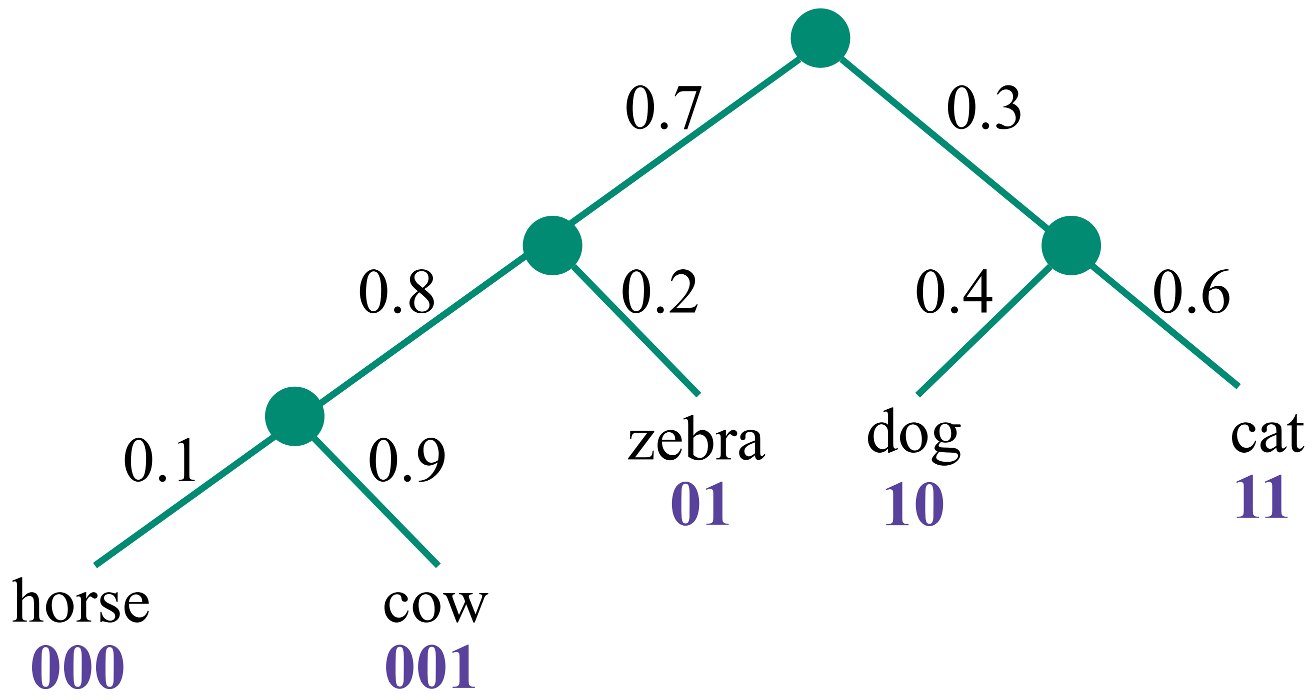
$$
\begin{array}{ll}
 & 0.7 \cdot 0.8 \cdot 0.1 \\
 & 0.7 \cdot 0.8 \cdot 0.9 \\
+ & 0.7 \cdot 0.2 \\
 & 0.3 \cdot 0.4 \\
 & 0.3 \cdot 0.6
\end{array}
$$



Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Hierarchical softmax

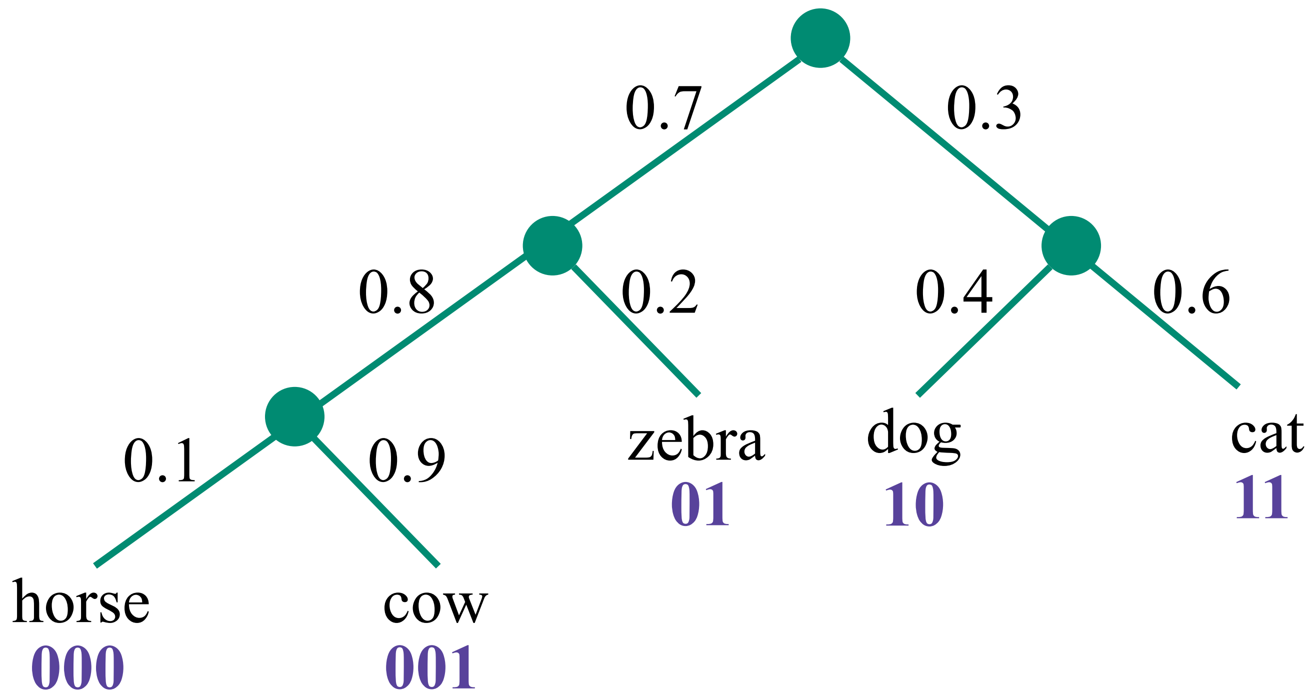
$$
\begin{array}{ll}
 & \color{red}{0.7 \cdot 0.8} \\
 & 0.7 \cdot 0.2 \\
+ & 0.3 \cdot 0.4 \\
 & 0.3 \cdot 0.6
\end{array}
$$



Anna Potapenko (HSE), Anastasia Ianina (MIPT)

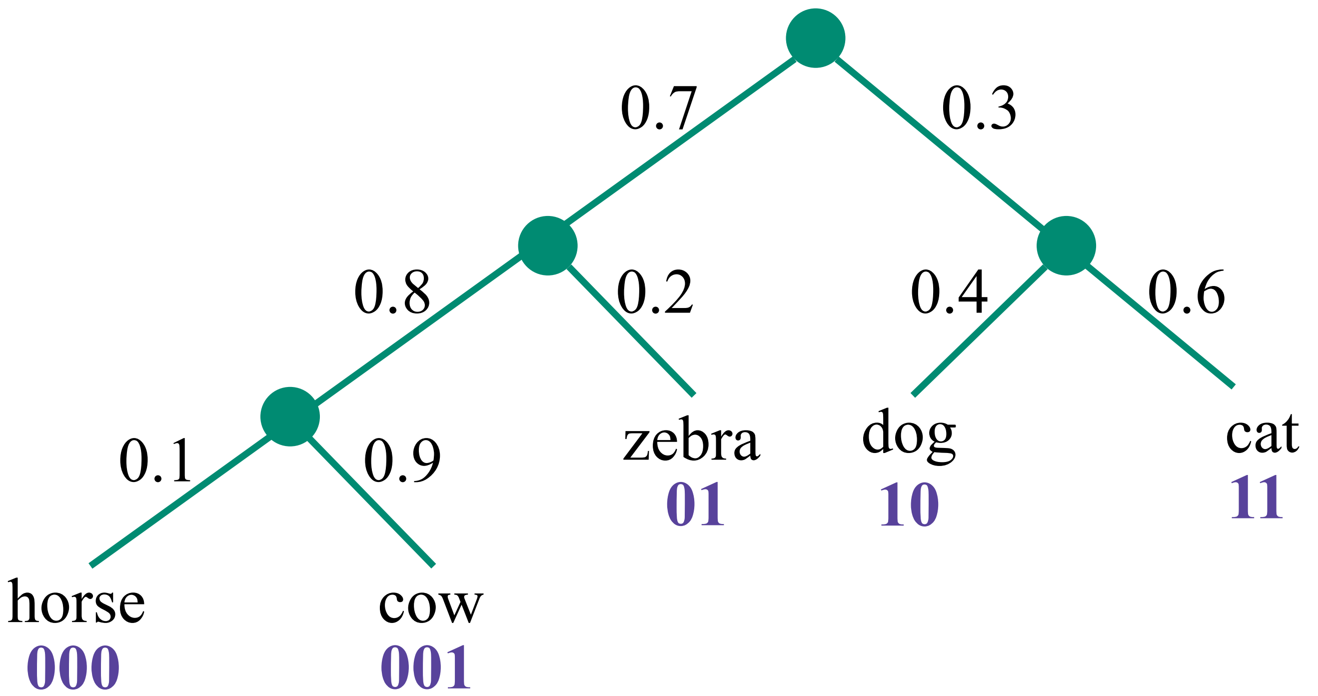
Hierarchical softmax

$$
\begin{array}{l}
\ 0.7 \cdot 0.8 \\
\ 0.7 \cdot 0.2 \\
+\ 0.3 \cdot 0.4 \\
\ 0.3 \cdot 0.6
\end{array}
$$



Hierarchical softmax

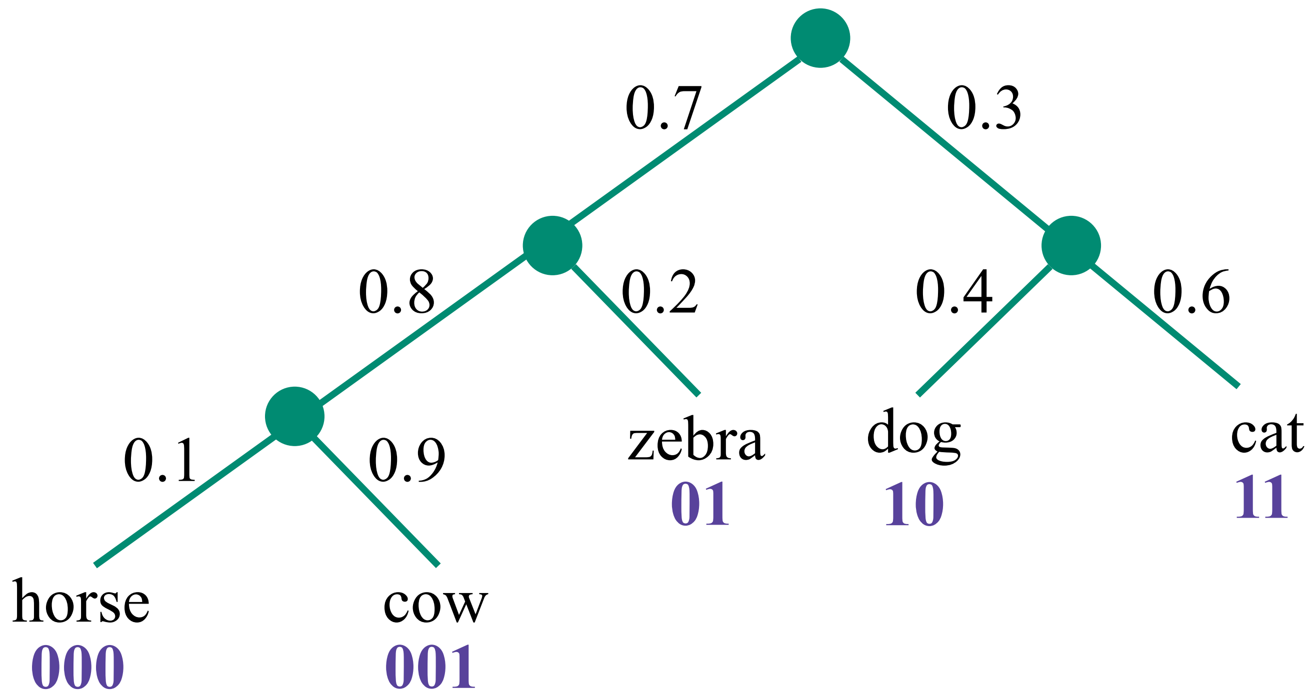
$$
\begin{array}{rl}
 & 0.7 \\
 & 0.3 \cdot 0.4 \\
+ & 0.3 \cdot 0.6
\end{array}
$$



Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Hierarchical softmax

$$
\begin{array}{rl}
 & 0.7 \\
 & 0.3 \cdot 0.4 \\
+ & 0.3 \cdot 0.6
\end{array}
$$



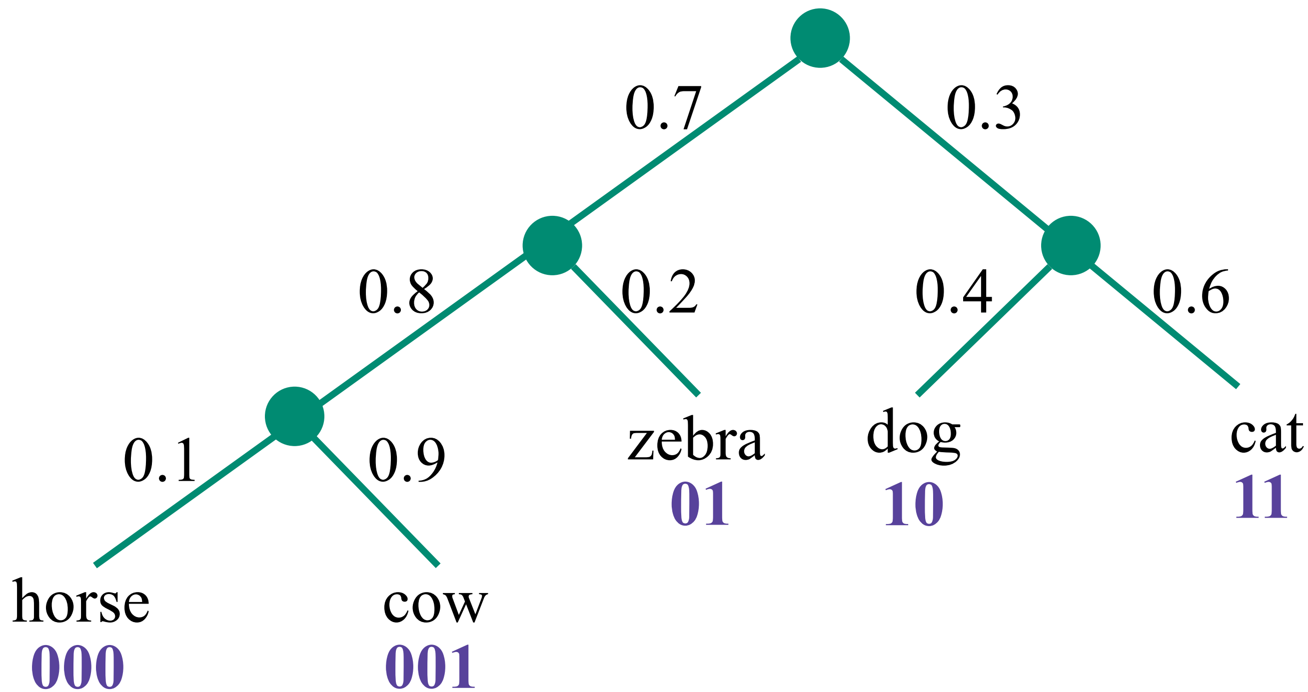
Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Hierarchical softmax

0.7

0.3

+

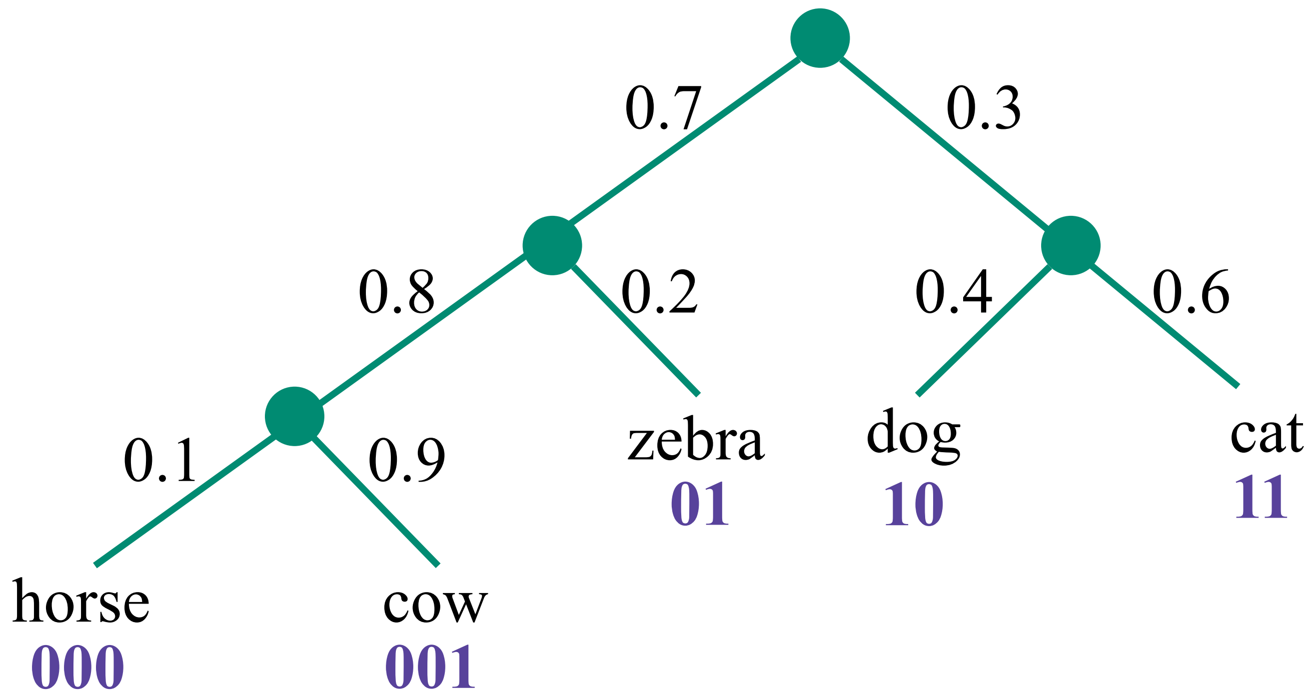


Hierarchical softmax

0.7

0.3

+



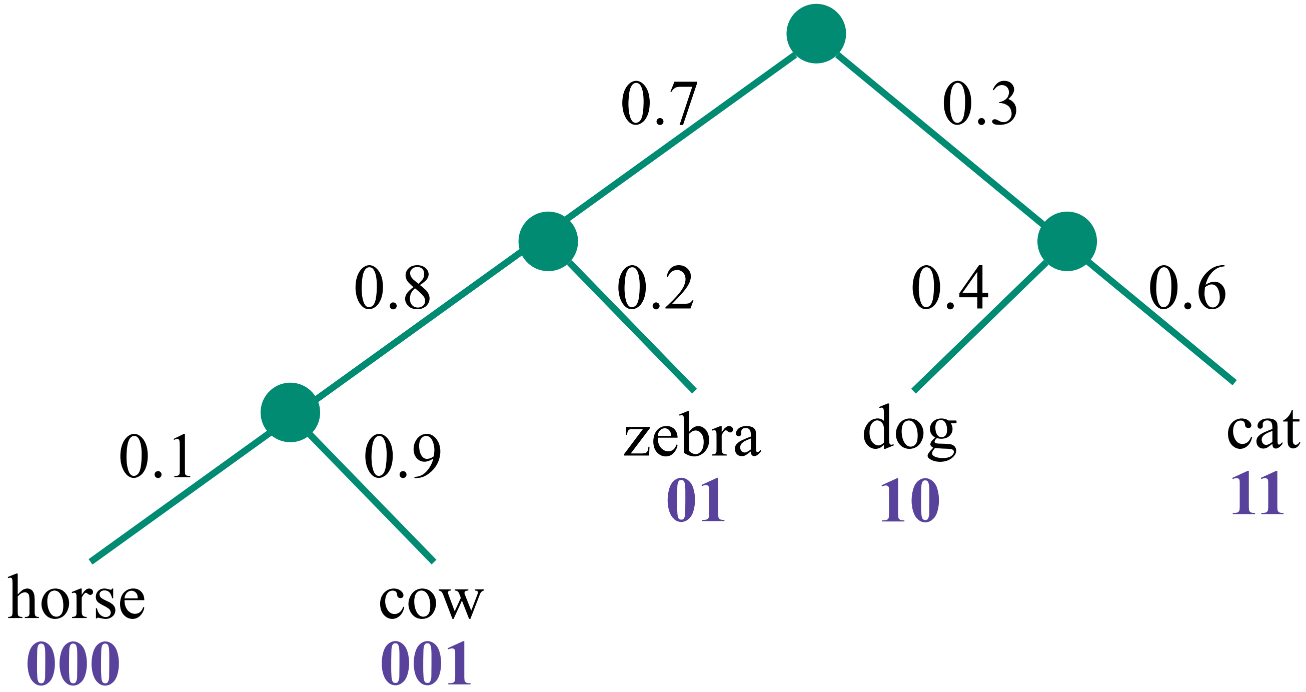
Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Hierarchical softmax

1.0

+

Congratulations!



Hierarchical softmax

Model binary decisions along the path in the tree:

$$p(w_n = w | w_1^{n-1}) = \prod_i p(d_i | w_1^{n-1})$$

How to construct a tree (balanced vs. semantic):

- Based on some pre-built ontology
- Based on semantic clustering from data
- Huffman tree
- Random

Outline

- Computing *softmax* for a large vocabulary is slow!
    - Hierarchical softmax
- Even a large vocabulary has *OOV words*:
    - **Copy mechanism**
    - Sub-word modeling
        - Word-character hybrid models
        - Byte-pair encoding

Copy mechanism

- Scaling *softmax* is insufficient!
- What do we do with OOV words?
    - Names, numbers, rare words…

ecotax *Pont-de-Buis*

The UNK portico in UNK

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Copy mechanism

- Scaling *softmax* is insufficient!
- What do we do with OOV words?
    - Names, numbers, rare words…

ecotax *Pont-de-Buis*

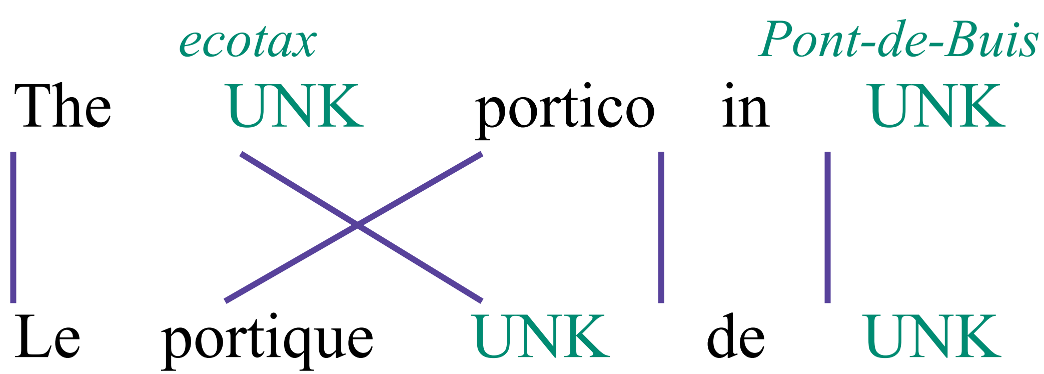
The UNK portico in UNK

Le portique UNK de UNK

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Copy mechanism

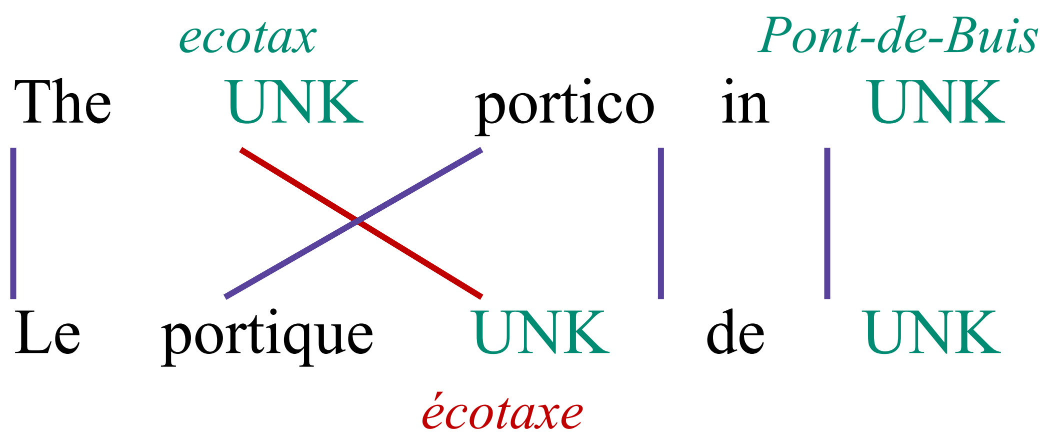
- Scaling *softmax* is insufficient!
- What do we do with OOV words?
    - Names, numbers, rare words…



Copy mechanism

- Scaling *softmax* is insufficient!
- What do we do with OOV words?
    - Names, numbers, rare words…

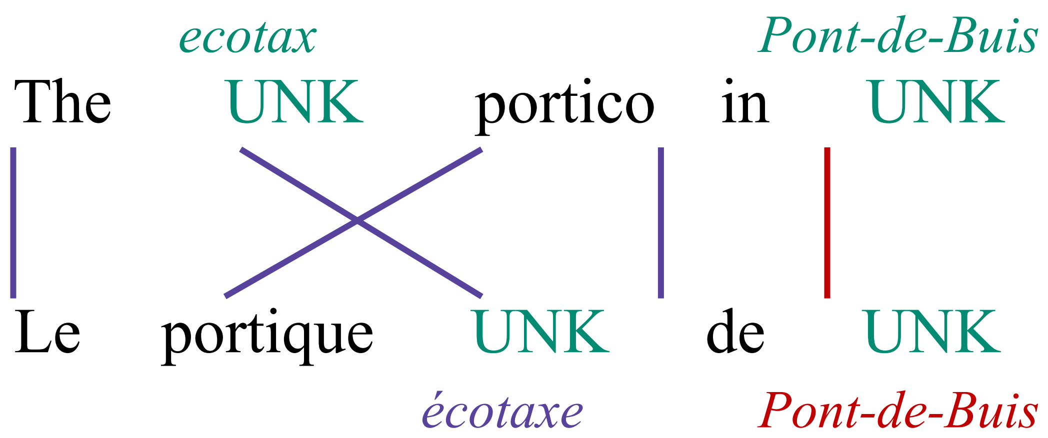
Look-up in a dictionary



Copy mechanism

- Scaling *softmax* is insufficient!
- What do we do with OOV words?
    - Names, numbers, rare words…

Look-up in a dictionary **Copy name**



Copy mechanism

Algorithm:

- Provide word alignments in train time
- Learn relative positions for UNK tokens with NMT
- Post-process the translation:
  - Copy the source word
  - Look up in a dictionary

Simple, but super useful technique!

Towards open vocabulary

Still problems:

- Transliteration: Christopher ↦ Kryštof
- Multi-word alignment: Solar system ↦ Sonnensystem
- Rich morphology: nejneobhospodařovávatelnějšímu
- Informal spelling: goooooood morning !!!!!

Outline

- Computing *softmax* for a large vocabulary is slow!
  - Hierarchical softmax
- Even a large vocabulary has *OOV words*:
  - Copy mechanism
  - **Sub-word modeling**
    - Word-character hybrid models
    - Byte-pair encoding

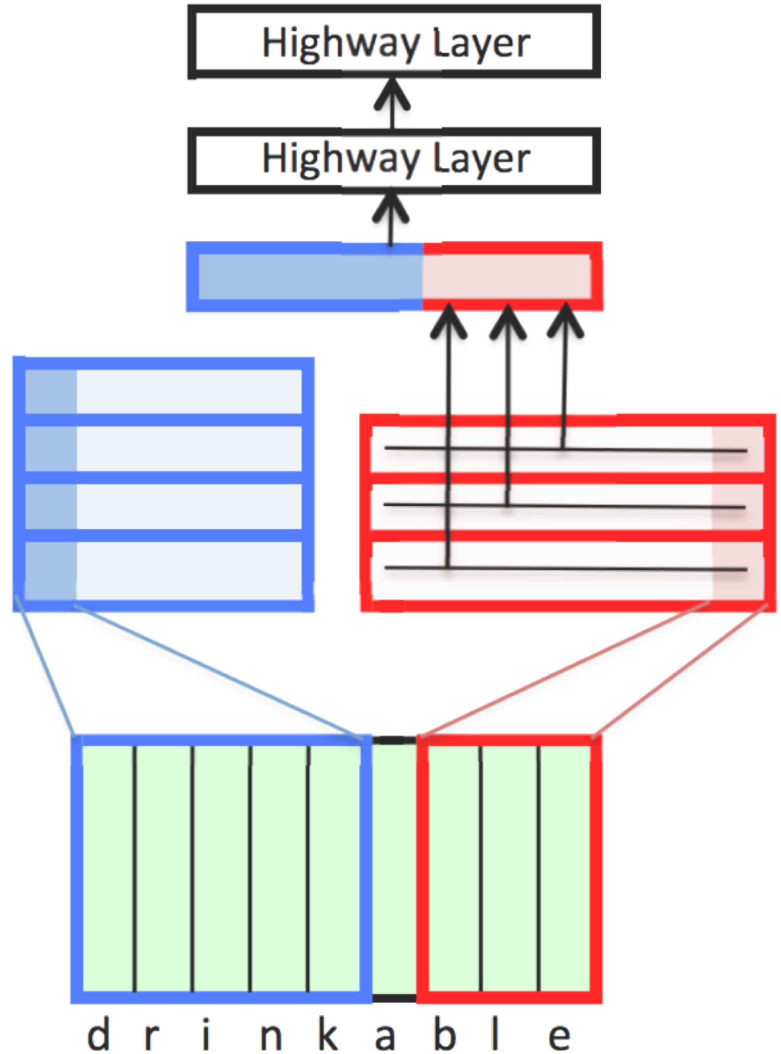
Outline

- Computing *softmax* for a large vocabulary is slow!
  - Hierarchical softmax
- Even a large vocabulary has *OOV words*:
  - Copy mechanism
  - Sub-word modeling
    - **Word-character hybrid models**
    - Byte-pair encoding

Character-based models

Character-based encoder is good for source languages with rich morphology!

- Bi-LSTMs to build word embeddings from characters
- CNNs on characters



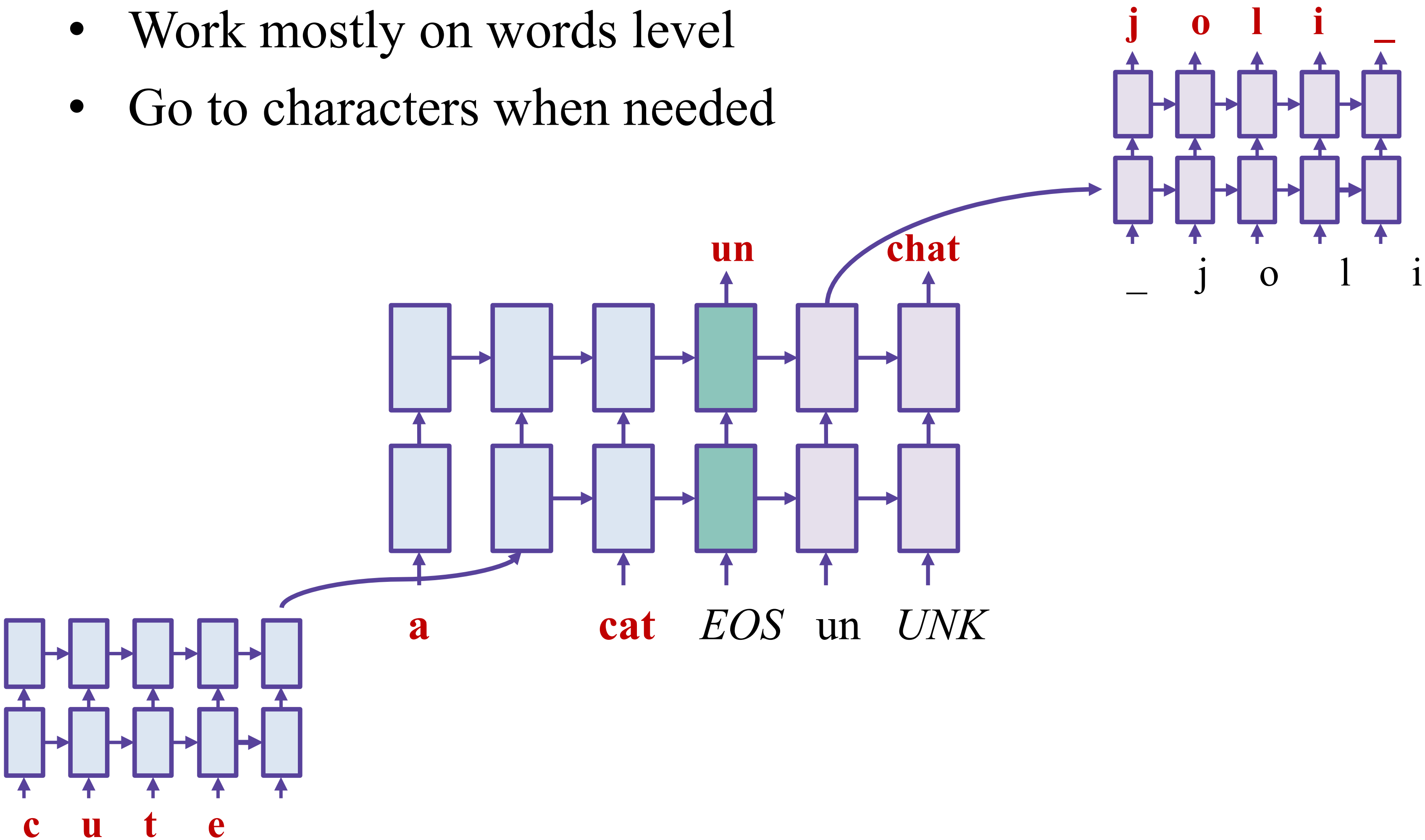
Ling, et. al. Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. EMNLP 2015.

Kim, et. al. Character-Aware Neural Language Models. AAAI 2016.

Marta R. Costa-jussà and José A. R. Fonollosa. Character-based Neural Machine Translation. ACL 2016.

Hybrid models: the best of two worlds

- Work mostly on words level
- Go to characters when needed



Thang Luong and Chris Manning. Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models. ACL 2016.

Outline

- Computing *softmax* for a large vocabulary is slow!
  - Hierarchical softmax
- Even a large vocabulary has *OOV words*:
  - Copy mechanism
  - Sub-word modeling
    - Word-character hybrid models
    - **Byte-pair encoding**

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

She sells seashells by the seashore

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

S h e _ s e l l s _ s e a s h e l l s _ b y _ t h e _ s e a s h o r e _

Compute how many times we see each character bAIgram

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

S h e _ s e l l s _ s e a s h e l l s _ b y _ t h e _ s e a s h o r e _

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Sh e _ s e l l s _ s e a sh e l l s _ b y _ t h e _ s e a sh o r e _

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Sh e _ s e l l s _ s e a sh e l l s _ b y _ t h e _ s e a sh o r e _

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Sh e _ se l l s _ se a sh e l l s _ b y _ t h e _ se a sh o r e _

Anna Potapenko (HSE), Anastasia Ianina (MIPT)

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Sh e _ se l l s _ se a sh e l l s _ b y _ t h e _ se a sh o r e _

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Sh e _ se ll s _ se a sh e ll s _ b y _ t h e _ se a sh o r e _

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Sh e _ se ll s _ se a sh e ll s _ b y _ t h e _ se a sh o r e _

Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Sh e _ se ll s _ sea sh e ll s _ b y _ t h e _ sea sh o r e _

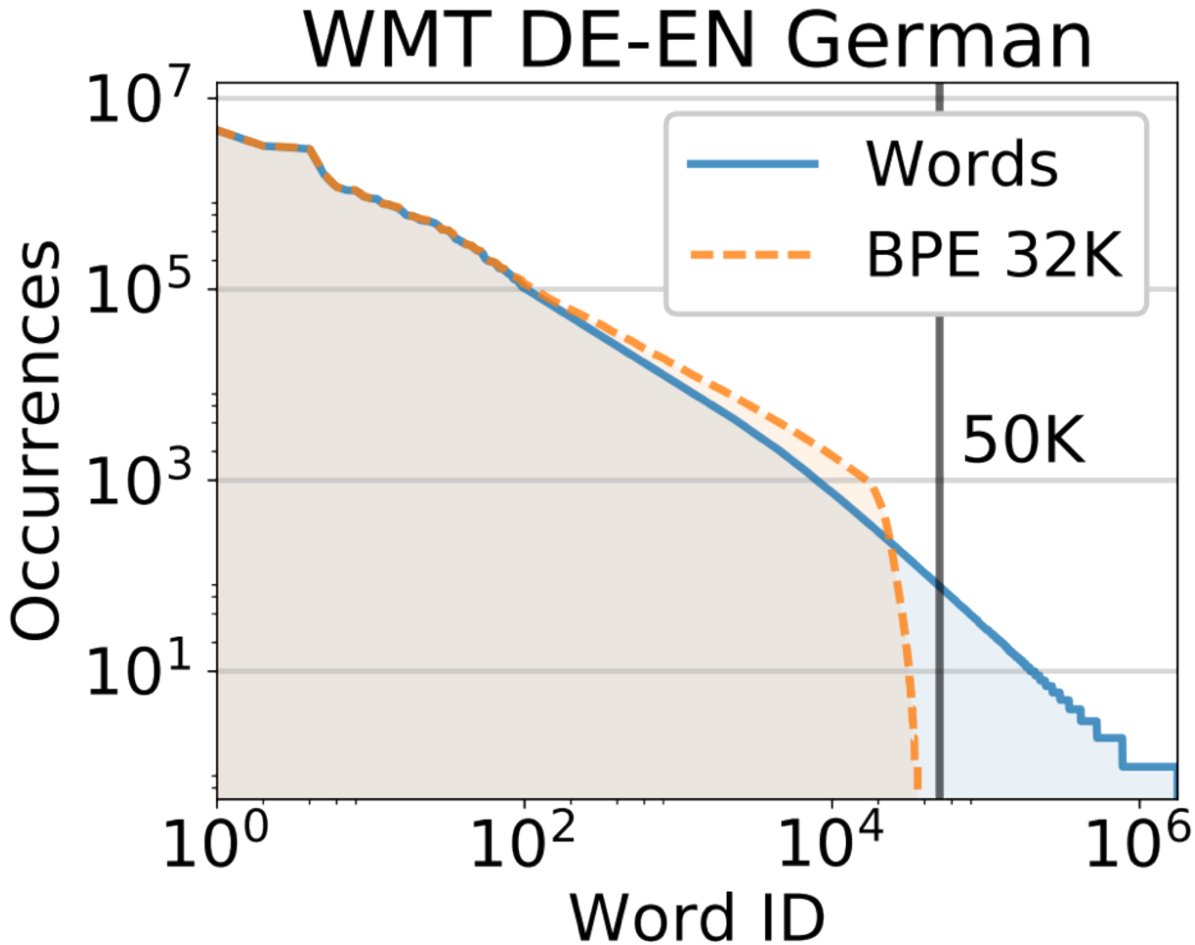
Byte-pair encoding

- Simple way to handle open vocabulary:
  - Start with characters
  - Iteratively replace the most frequent pair with one unit

Sh e _ se ll s _ sea sh e ll s _ b y _ t h e _ sea sh o r e _

- End whenever you reach the vocabulary size limit
- Stick to that vocabulary of sub-word units
- Apply the same algorithm to test sentences

Why is it so useful?



Denkowski, Neubig. Stronger Baselines for Trustable Results in Neural Machine Translation, 2017.

BLEU score comparison

	WMT			IWSLT	
	DE-EN	EN-FI	RO-EN	EN-FR	CS-EN
Words 50K	31.6	12.6	27.1	33.6	21.0
BPE 32K	**33.5**	**14.7**	**27.8**	34.5	22.6
BPE 16K	33.1	**14.7**	**27.8**	**34.8**	**23.0**

- Byte-pair encoding improves BLEU score
- It is a nice and simple way to handle the vocabulary
- Very common trick in modern NMT

Denkowski, Neubig. Stronger Baselines for Trustable Results in Neural Machine Translation, 2017.