Faculty for Informatics

Technical
University
of Munich

Natural Language Processing
IN2361

PD Dr. Georg Groh

Social Computing
Research Group

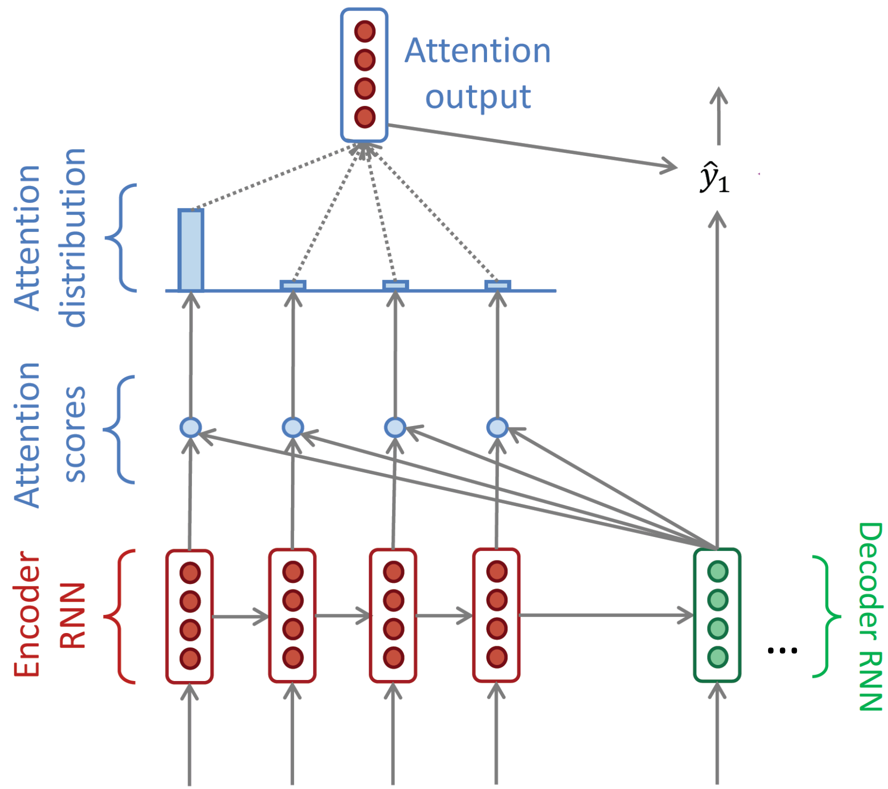
Deep NLP
Part D: Advanced Attention

- content is based on [2] (lecture 11)
- certain elements (e.g. figures, equations or tables) were taken over or taken over in a modified form from [2]
- citations of [2] are omitted for legibility
- errors on these slides are fully in the responsibility of Georg Groh
- BIG thanks to Richard Socher and his colleagues at Stanford for publishing materials [2] of a great Deep NLP lecture

General Attention

- More **general definition of Attention**:
  - given: set of vector values and a vector query
  - Attention: technique to compute a weighted sum of the values, dependent on the query.
- "the query attends to the values"

- seq2seq + attention from before:
  - values: encoder hidden states
  - query: decoder hidden state



General Attention

- More **general definition of Attention**:
  - given: set of vector values and a vector query
  - Attention: technique to compute a weighted sum of the values, dependent on the query.

- **intuition**:
  - weighted sum: selective summary of the information contained in the values, where the query determines which values to focus on.
  - Attention: a way to obtain a fixed-size representation of an arbitrary set of representations (the values), dependent on some other representation (the query).

General Attention

- values: $\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and query $\boldsymbol{s} \in \mathbb{R}^{d_2}$
- attention scores: $\boldsymbol{e} \in \mathbb{R}^N$
- attention distribution: $\boldsymbol{\alpha} = \mathrm{softmax}\,(\boldsymbol{e}) \in \mathbb{R}^N$
- attention output: $\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i \, \boldsymbol{h}_i \in \mathbb{R}^{d_1}$
  (also often denoted as $\boldsymbol{c}_t$ (context vector))

Attention Variants

- values: $\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and query $\boldsymbol{s} \in \mathbb{R}^{d_2}$
- attention scores: $\boldsymbol{e} \in \mathbb{R}^N$
  - basic dot-product attention (as before): $e_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$

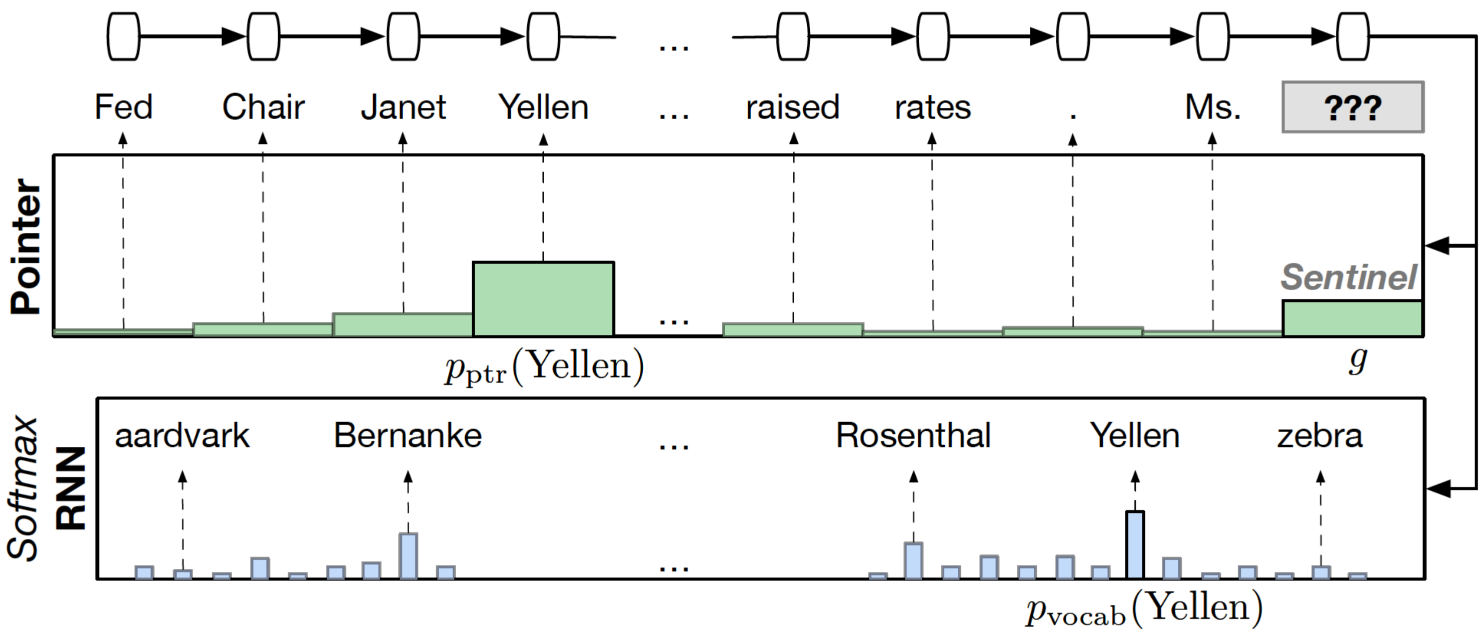
    (assumes $d_1 = d_2$)
  - Multiplicative (bilinear) attention: $e_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$

    where $\boldsymbol{W} \in \mathbb{R}^{d_2 \times d_1}$
  - Additive attention: $e_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$

    where $\boldsymbol{W}_1 \in \mathbb{R}^{d_3 \times d_1}$, $\boldsymbol{W}_2 \in \mathbb{R}^{d_3 \times d_2}$, $\boldsymbol{v} \in \mathbb{R}^{d_3}$
- attention distribution: $\boldsymbol{\alpha} = \mathrm{softmax}(\boldsymbol{e}) \in \mathbb{R}^N$
- attention output: $\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i \boldsymbol{h}_i \in \mathbb{R}^{d_1}$
  (also often denoted as $\boldsymbol{c}_t$ (context vector))

Application: Pointer Sentinel Networks for Language Modelling [7]

- LM with vanilla RNN + V-dim. softmax: bottleneck problem (affects performance) but can predict words in V but not in input (unseen words)
- pointer networks (*predict member of input sequence with highest attention as next word*): better performance (due to attention) on rare words and out of vocabulary words (seen in input) but unable to predict unseen words
- → combine both: use a mixture model: (see [7]):

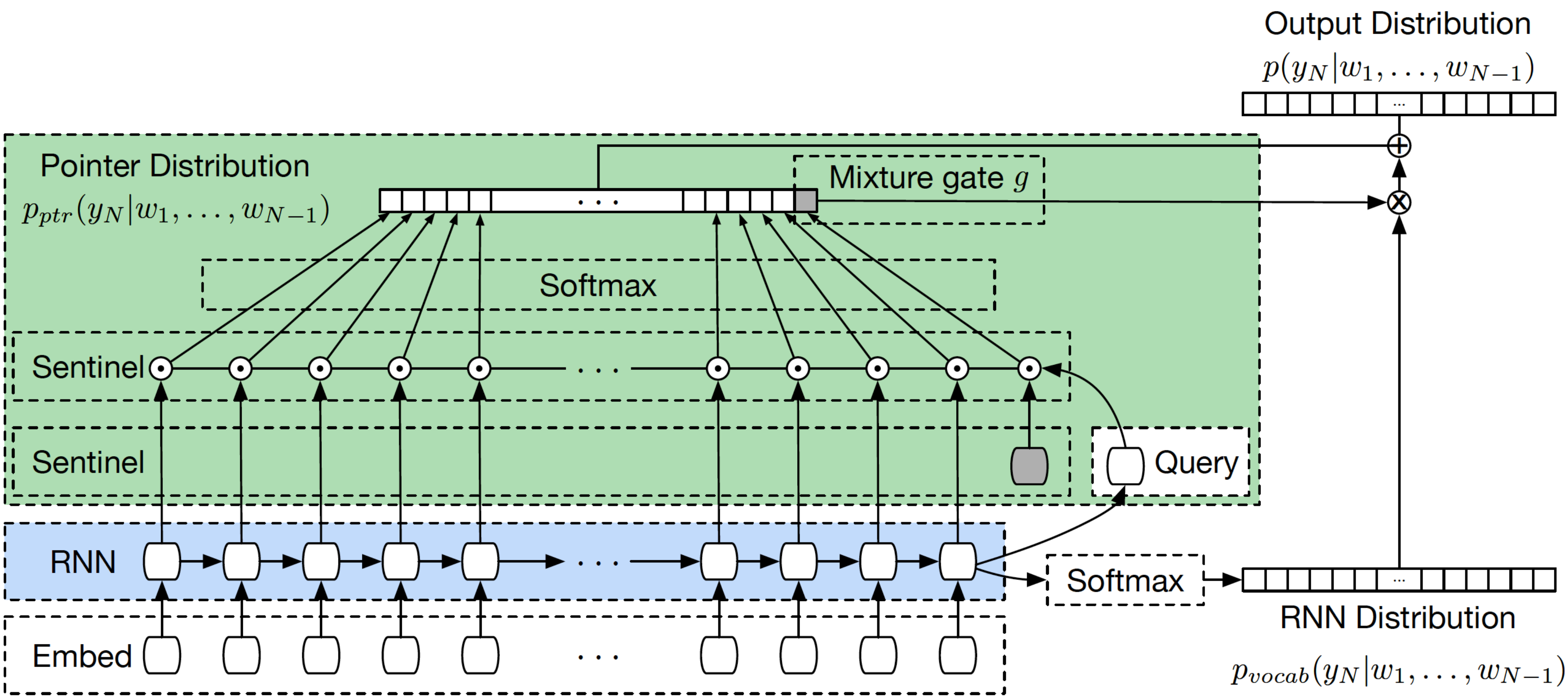


[7]

$$p(\mathrm{Yellen}) = g\, p_{\mathrm{vocab}}(\mathrm{Yellen}) + (1-g)\, p_{\mathrm{ptr}}(\mathrm{Yellen})$$

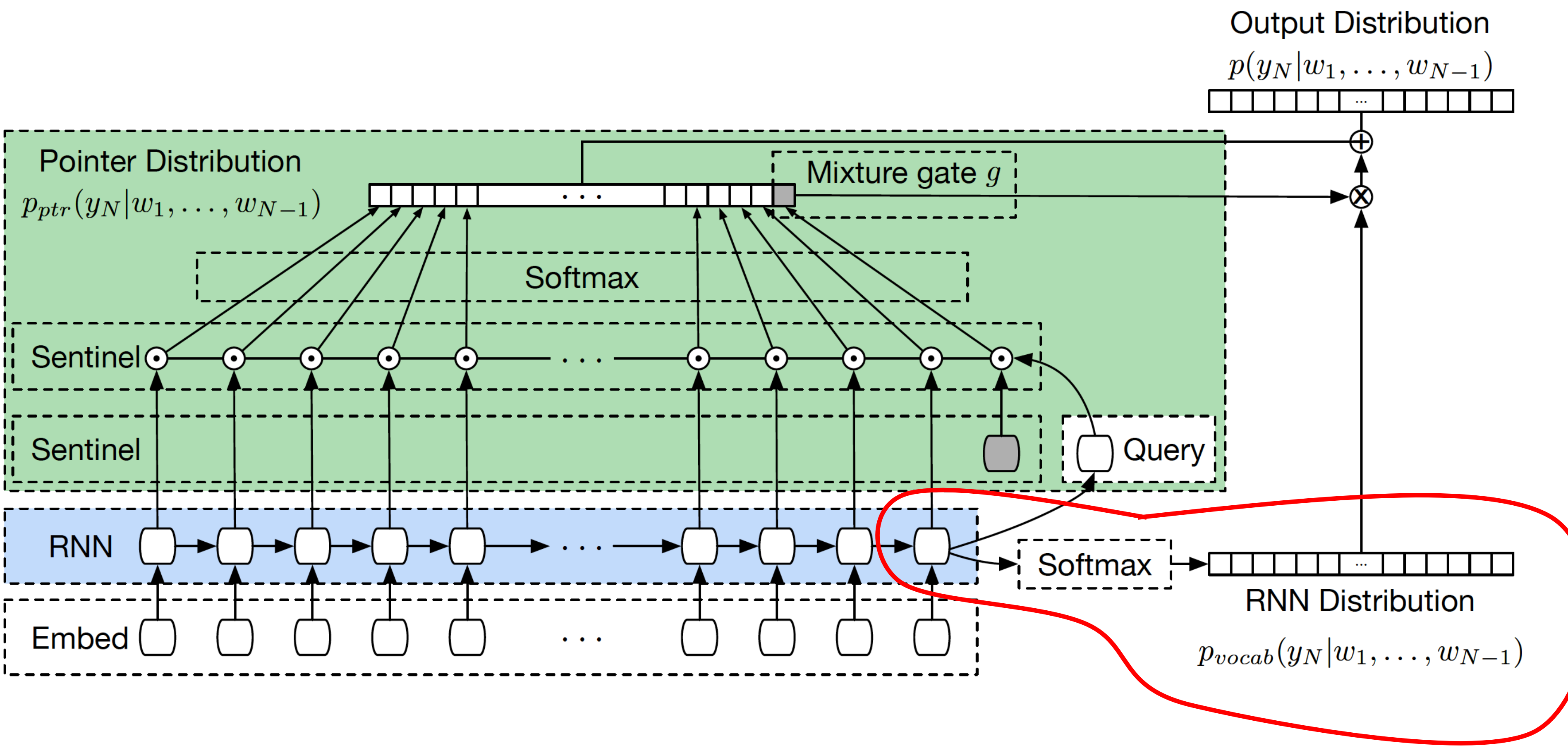
[7]: Merity, Xiong, Bradbury, Socher. Pointer sentinel mixture models.(2016)

Application: Pointer Sentinel Networks for Language Modelling [7]



[7]

Application: Pointer Sentinel Networks for Language Modelling [7]

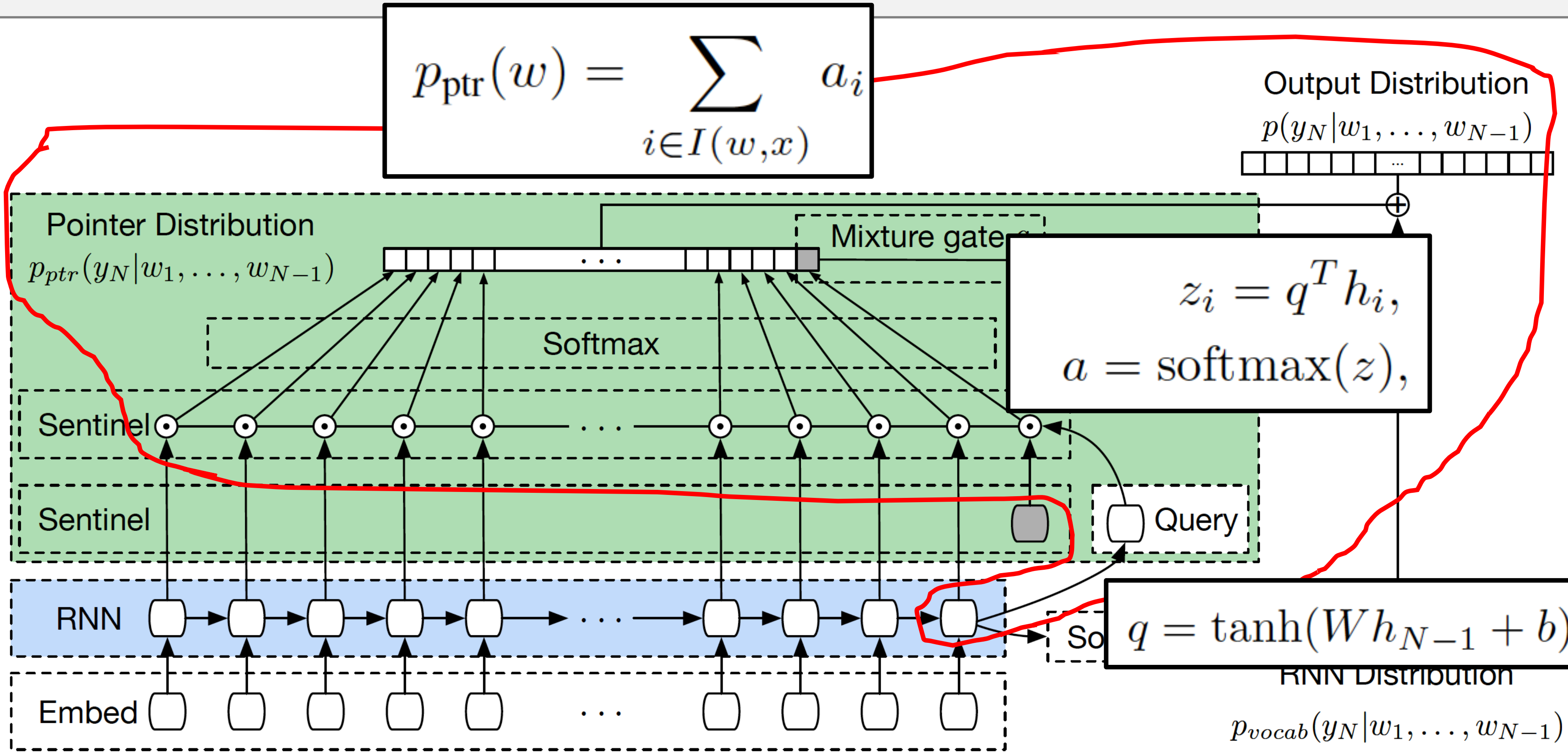


$$p_{\text{vocab}}(w) = \text{softmax}(Uh_{N-1})$$

$$p_{\text{vocab}} \in \mathbb{R}^V, U \in \mathbb{R}^{V \times H}$$

[7]

Application: Pointer Sentinel Networks for Language Modelling [7]



$$p_{\text{ptr}}(w) = \sum_{i \in I(w,x)} a_i$$

$$z_i = q^T h_i,$$

$$a = \text{softmax}(z),$$

$$q = \tanh(W h_{N-1} + b)$$

$$z \in \mathbb{R}^L, a \in \mathbb{R}^L$$

L : number of hidden states (same as window size / no. of steps of of RNN)

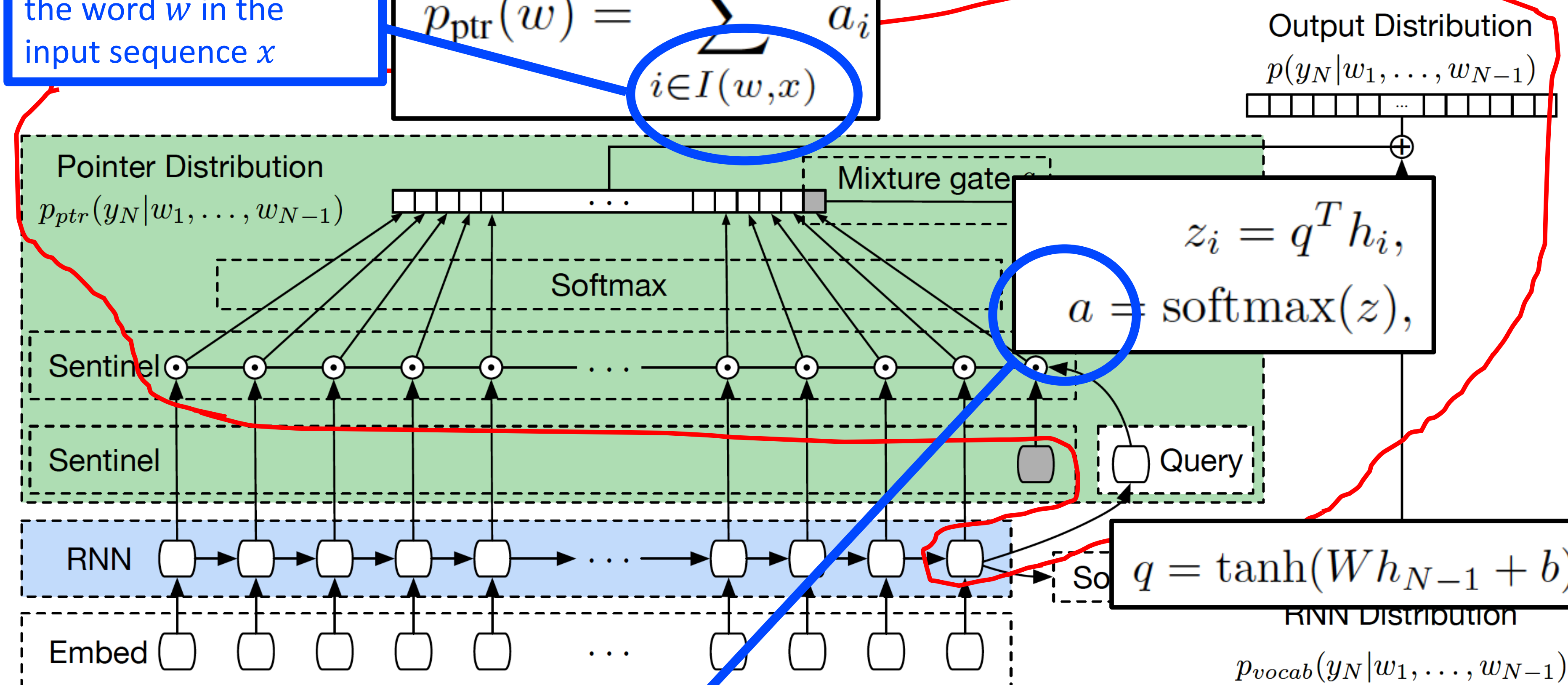
$I(w, x)$: all previous positions of the word w in input x

[7]

Application: Pointer Sentinel Networks for Language Modelling [7]

$I(w, x)$: all positions of the word w in the input sequence x

$$p_{\text{ptr}}(w) = \sum_{i \in I(w,x)} a_i$$



$$z_i = q^T h_i,$$
$$a = \text{softmax}(z),$$

$$q = \tanh(W h_{N-1} + b)$$

$$z \in \mathbb{R}^L, a \in \mathbb{R}^L$$

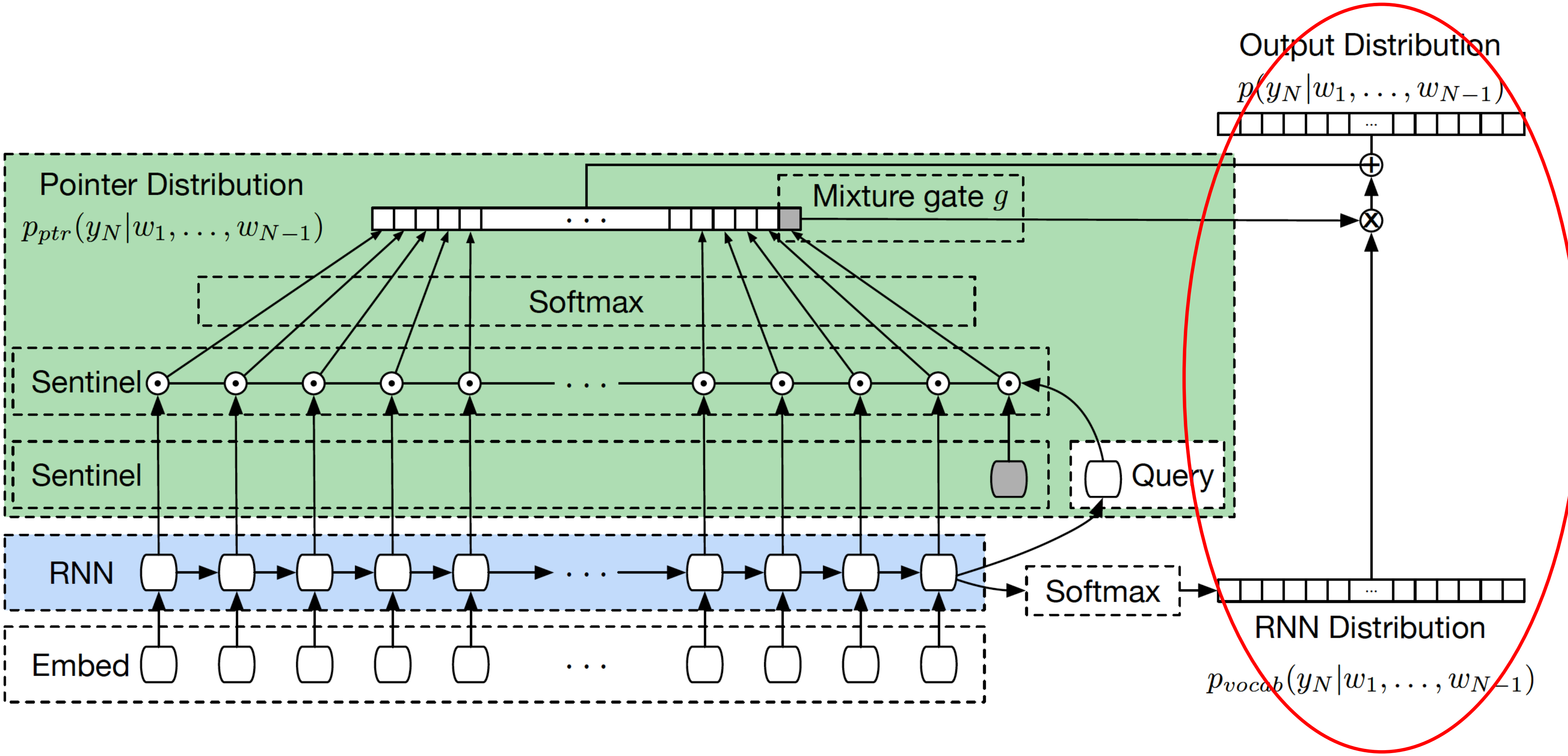
in previous slides' notation this should be denoted as α . Here we use the notation of the paper [7]

n states (same as window size / no. of steps of of RNN)

[7]

$I(w, x)$: positions of w in input x

Application: Pointer Sentinel Networks for Language Modelling [7]

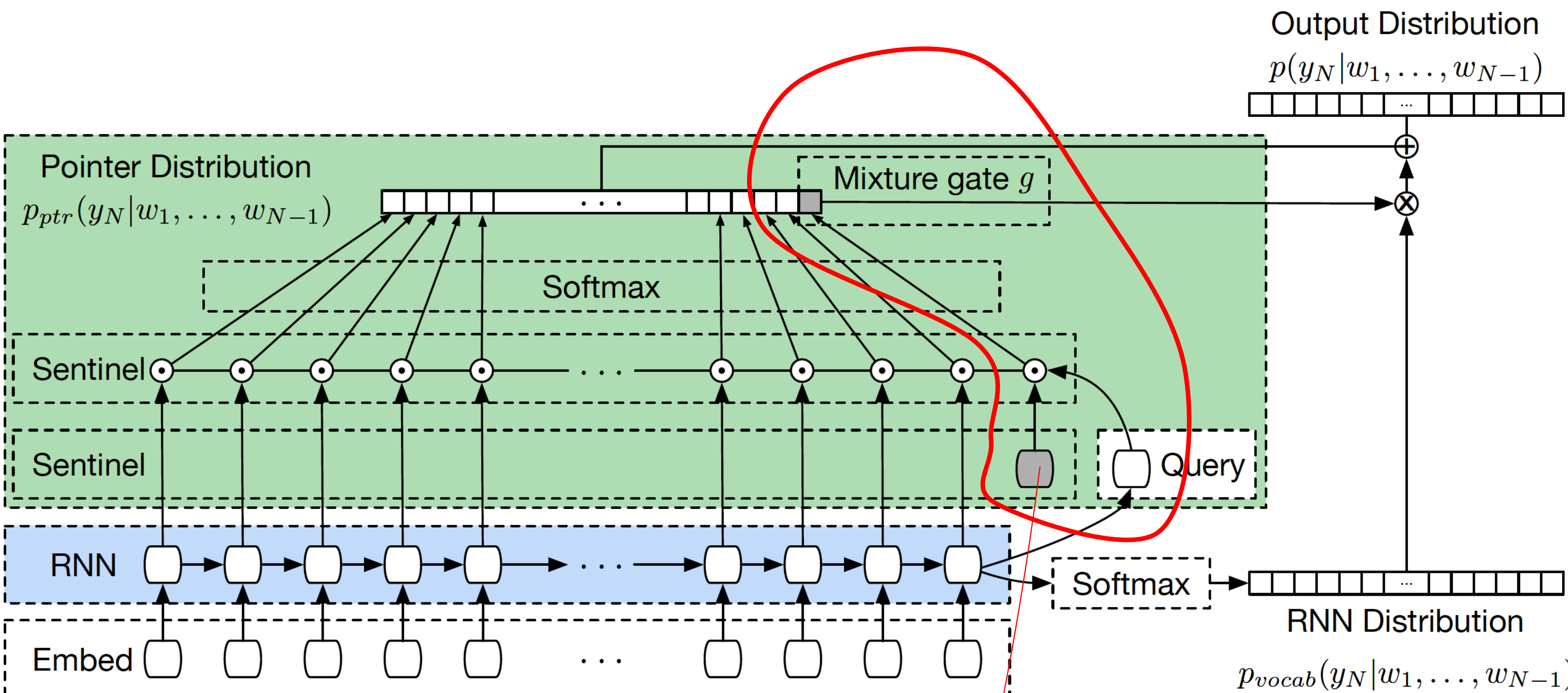


$$p(y_i|x_i) = g\, p_{\text{vocab}}(y_i|x_i) + (1-g)\, p_{\text{ptr}}(y_i|x_i).$$

[7]

use pointers whenever possible and back-off to standard softmax otherwise

Application: Pointer Sentinel Networks for Language Modelling [7]



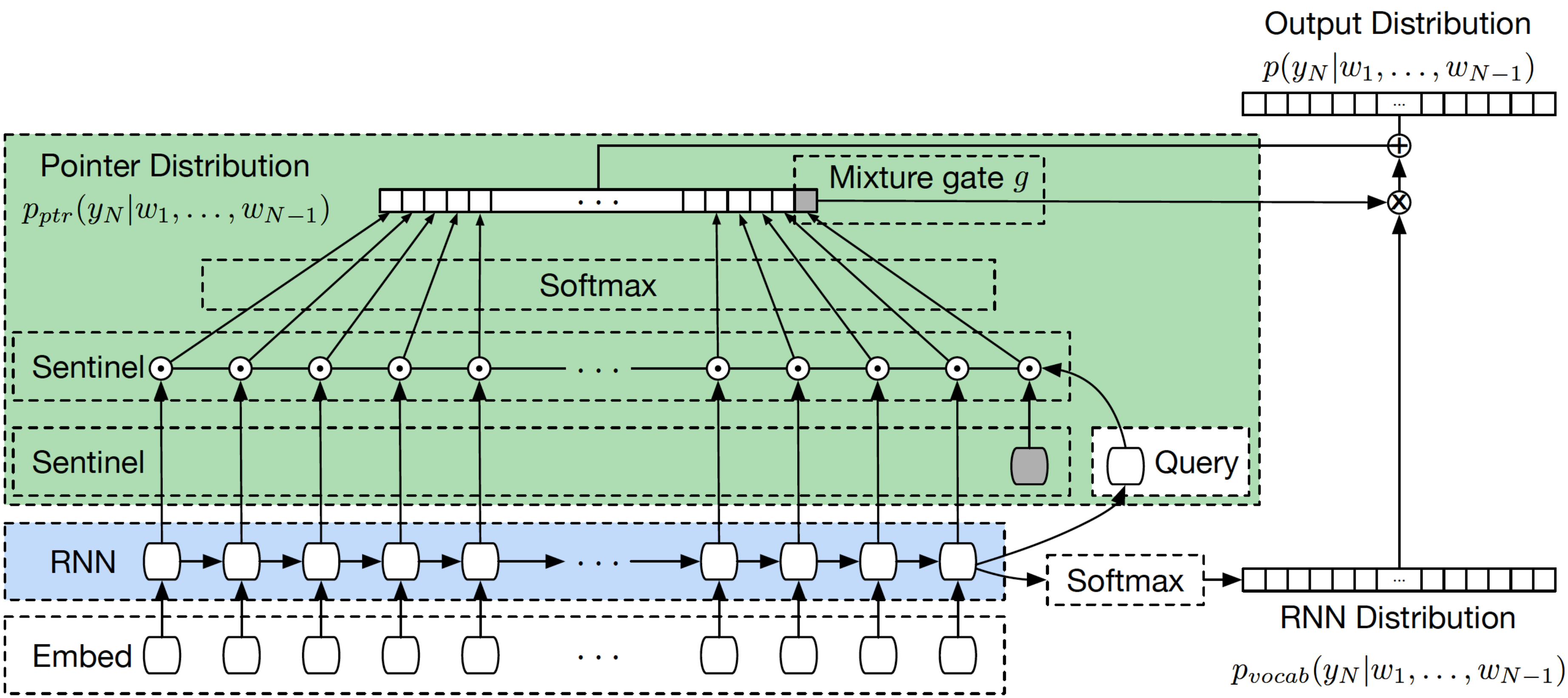
$$a = \mathrm{softmax}\left(\left[\boldsymbol{z}; q^T s\right]\right) \qquad g = a[V+1]$$

sentinel vector $s \in \mathbb{R}^H$: trainable parameter

so actually, we concatenate with dot product $q^T s$, before computing the softmax and take it out again for the pointer distribution: $p_{\text{ptr}}(y_i|x_i) = \frac{1}{1-g}\, a[1:V]$

→ back-off: distribute the rest probability mass not given to pointers to the RNN part
== g is large if if pointer distribution is not peaked = "not sure" (←→ softmax)

Application: Pointer Sentinel Networks for Language Modelling [7]



$$a = \mathrm{softmax}\left(\left[\mathbf{z}; q^T s\right]\right) \qquad g = a[V+1]$$

Loss: standard Cross Entropy for RNN (vocab) and $-\log\left(g + \sum_{i\in I(y,x)} a_i\right)$ for the pointer component

$I(y,x)$: all positions of the ground truth y in the input sequence x; if $g \approx 1$: all emphasis on RNN → no loss for pointer network

Application: Pointer Sentinel Networks for LM: Performance [7]

Model	Parameters	Validation	Test
Mikolov & Zweig (2012) - KN-5	2M‡	–	141.2
Mikolov & Zweig (2012) - KN5 + cache	2M‡	–	125.7
Mikolov & Zweig (2012) - RNN	6M‡	–	124.7
Mikolov & Zweig (2012) - RNN-LDA	7M‡	–	113.7
Mikolov & Zweig (2012) - RNN-LDA + KN-5 + cache	9M‡	–	92.0
Pascanu et al. (2013a) - Deep RNN	6M	–	107.5
Cheng et al. (2014) - Sum-Prod Net	5M‡	–	100.0
Zaremba et al. (2014) - LSTM (medium)	20M	86.2	82.7
Zaremba et al. (2014) - LSTM (large)	66M	82.2	78.4
Gal (2015) - Variational LSTM (medium, untied)	20M	81.9 ± 0.2	79.7 ± 0.1
Gal (2015) - Variational LSTM (medium, untied, MC)	20M	–	78.6 ± 0.1
Gal (2015) - Variational LSTM (large, untied)	66M	77.9 ± 0.3	75.2 ± 0.2
Gal (2015) - Variational LSTM (large, untied, MC)	66M	–	73.4 ± 0.0
Kim et al. (2016) - CharCNN	19M	–	78.9
Zilly et al. (2016) - Variational RHN	32M	72.8	71.3
Zoneout + Variational LSTM (medium)	20M	84.4	80.6
Pointer Sentinel-LSTM (medium)	21M	72.4	**70.9**

Table 2. Single model perplexity on validation and test sets for the Penn Treebank language modeling task. For our models and the models of Zaremba et al. (2014) and Gal (2015), medium and large refer to a 650 and 1500 units two layer LSTM respectively. The medium pointer sentinel-LSTM model achieves lower perplexity than the large LSTM model of Gal (2015) while using a third of the parameters and without using the computationally expensive Monte Carlo (MC) dropout averaging at test time. Parameter numbers with ‡ are estimates based upon our understanding of the model and with reference to Kim et al. (2016).

[7]

Application 2: Intra-Decoder Attention for Summarization [8]

Tony Blair has said he does not want to retire until he is 91 – as he unveiled plans to set up a ‘cadre’ of ex-leaders to advise governments around the world. The defiant 61-year-old former Prime Minister said he had ‘decades’ still in him and joked that he would ‘turn to drink’ if he ever stepped down from his multitude of global roles. He told Newsweek magazine that his latest ambition was to recruit former heads of government to go round the world to advise presidents and prime ministers on how to run their countries. In an interview with the magazine Newsweek Mr Blair said he did not want to retire until he was 91 years old Mr Blair said his latest ambition is to recruit former heads of government to advise presidents and prime ministers on how to run their countries Mr Blair said he himself had been ‘mentored’ by US president Bill Clinton when he took office in 1997. And he said he wanted to build up his organisations, such as his Faith Foundation, so they are ‘capable of changing global policy’. Last night, Tory MPs expressed horror at the prospect of Mr Blair remaining in public life for another 30 years. Andrew Bridgen said: ‘We all know weak Ed Miliband’s called on Tony to give his flailing campaign a boost, but the attention’s clearly gone to his head.’ (...)

Abstractive Summarization
(possibly rephrase words / phrases)

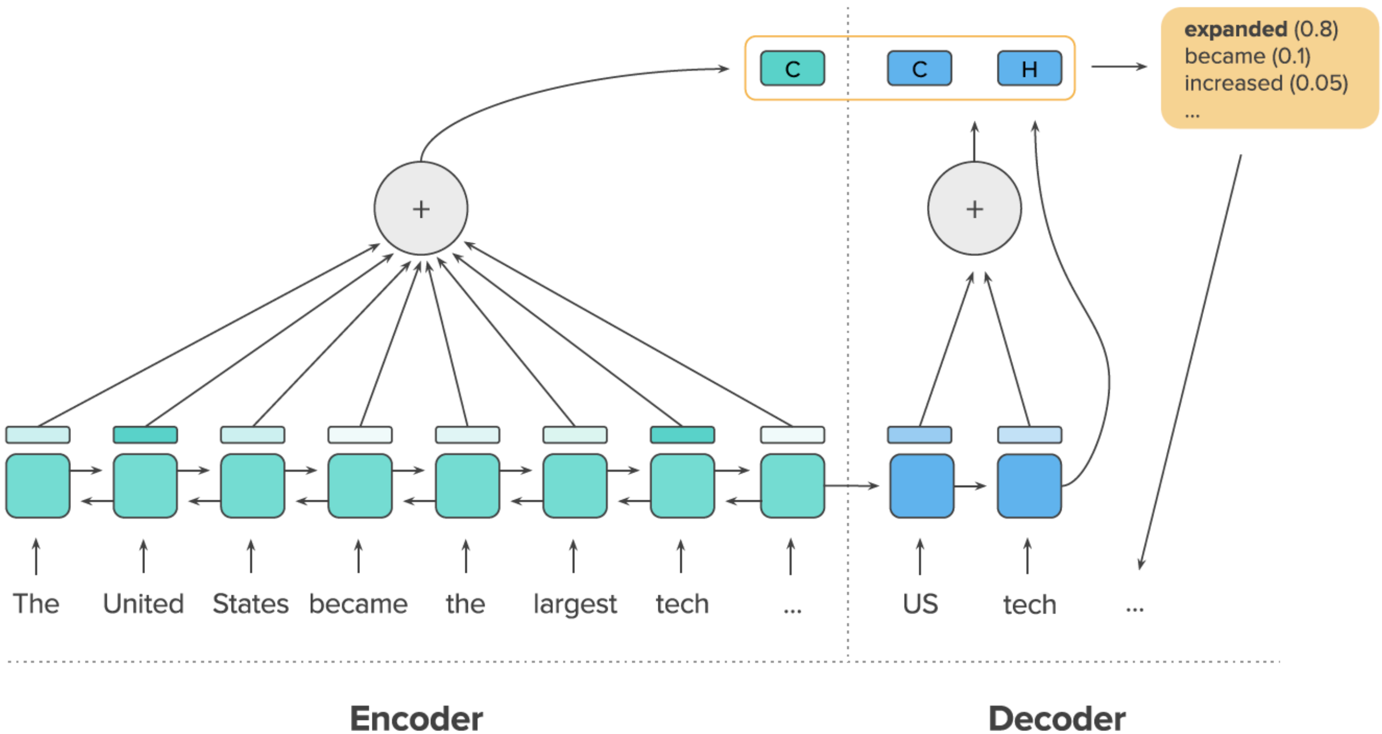
(in contrast to extractive summarization
(select and copy certain input parts))

The former Prime Minister claimed he has 'decades' of work left in him. Joked he would 'turn to drink' if he ever stepped down from global roles. Wants to recruit former government heads to advise current leaders. He was 'mentored' by US president Bill Clinton when he started in 1997.

[8]: Paulus, R., Xiong, C. and Socher, R., (2017). A deep reinforced model for abstractive summarization

Application 2: Intra-Decoder Attention for Summarization [8]

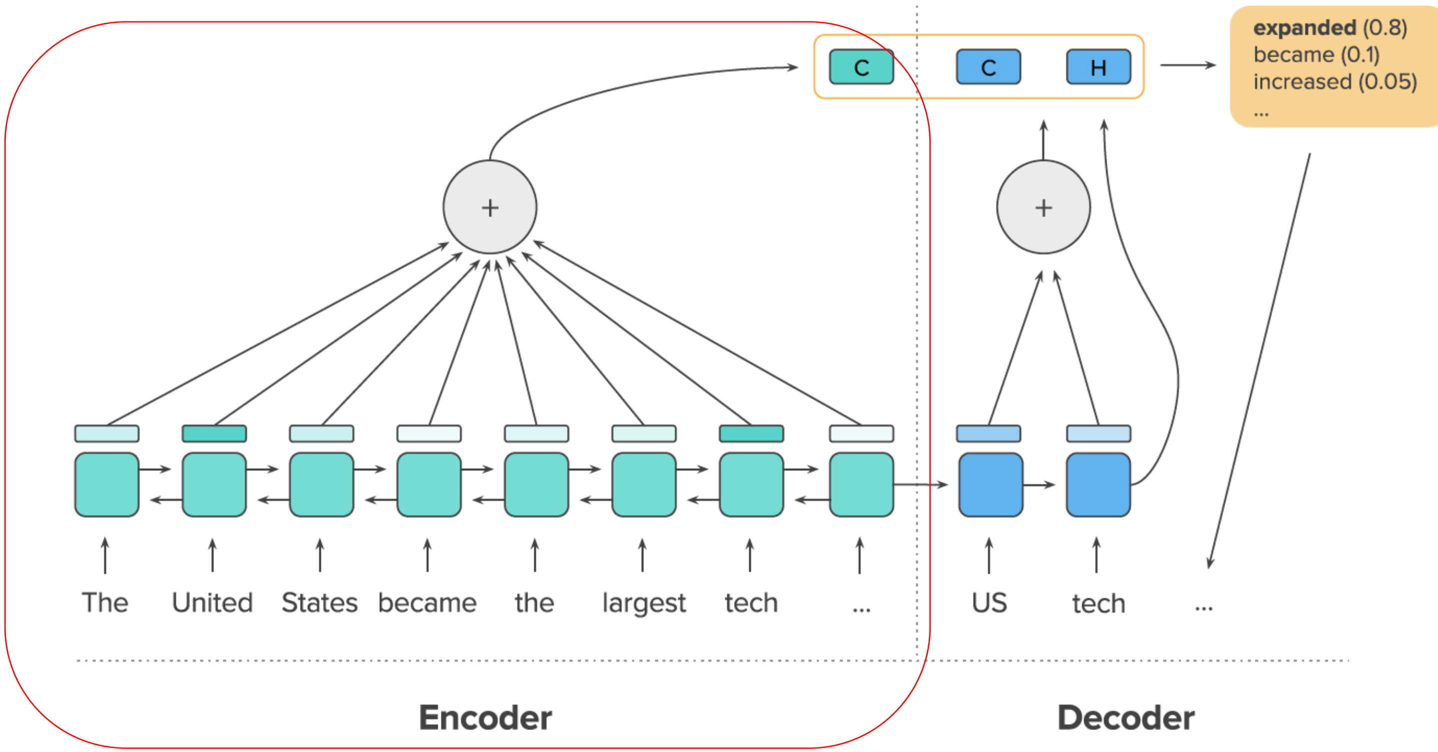
- problem: summarization → longer inputs + outputs → decoder starts to repeat itself → more advanced attention beneficial
  - dot-product encoder attention → multiplicative intra temporal encoder attention on input sequence
  - Self-attention (intra decoder attention)



[8]

Figure 1: Illustration of the encoder and decoder attention functions combined. The two context vectors (marked "C") are computed from attending over the encoder hidden states and decoder hidden states. Using these two contexts and the current decoder hidden state ("H"), a new word is generated and added to the output sequence.

Intra Temporal Attention on Input [8]

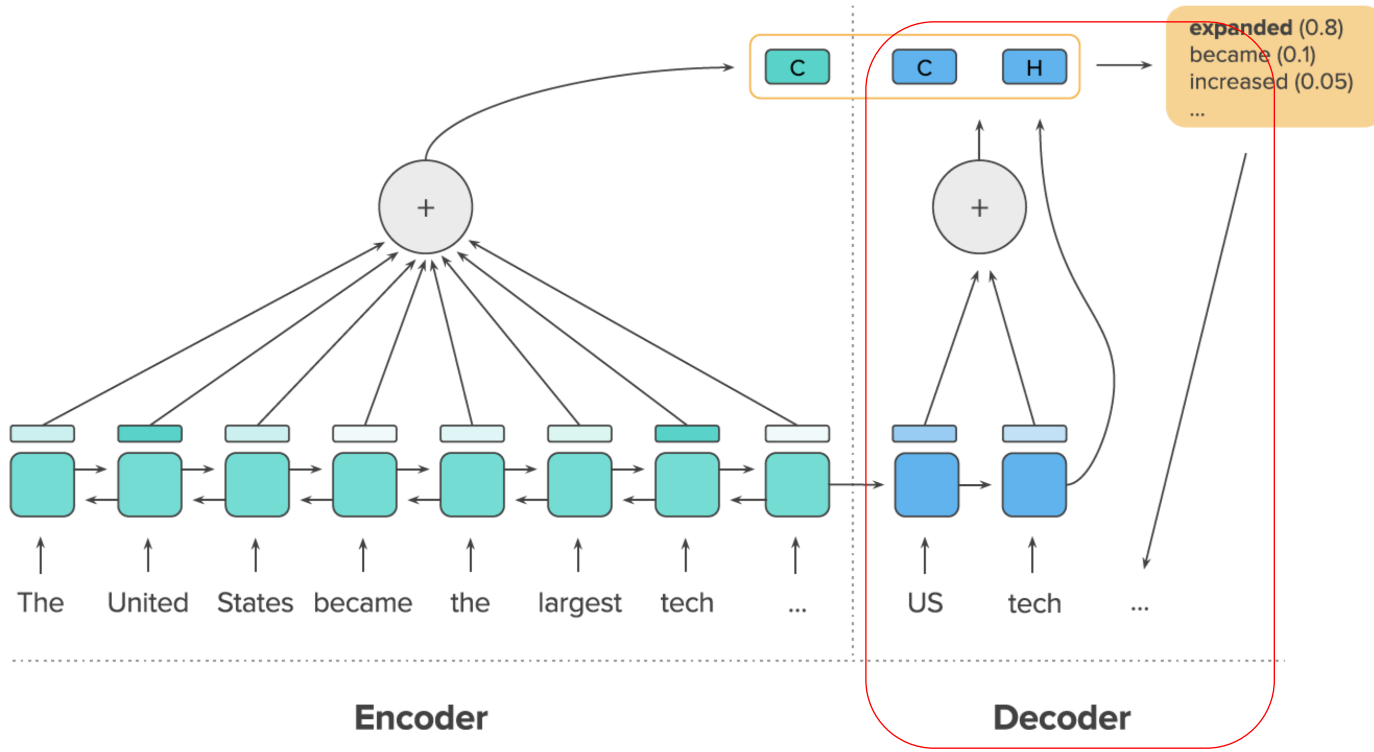


- more advanced similarity (multiplicative): $e_{ti} = {h_t^d}^T W_{\text{attn}}^e h_i^e$
- encoder intra-temporal attention distribution: $\alpha_{ti}^e = \dfrac{e'_{ti}}{\sum_{j=1}^{n} e'_{tj}}$ where $e'_{ti} = \begin{cases} exp(e_{ti}) & \text{if } t = 1 \\ \dfrac{exp(e_{ti})}{\sum_{j=1}^{t-1} \exp(e_{ji})} & \text{otherwise.} \end{cases}$
- usual encoder attention output ("input context vector") :

$$c_t^e = \sum_{i=1}^{n} \alpha_{ti}^e h_i^e$$

penalize input tokens that have obtained high attention scores in past decoding steps → improve coverage and prevent repeated attention to same input elements

Self Attention on Decoder [8]



- decoder self attention: hidden decoder states attend to themselves
→ make more structured predictions and avoid repeating the same information, even if that information was generated many steps away:

$$\alpha^d_{tt'} = \frac{exp(e^d_{tt'})}{\sum_{j=1}^{t-1} exp(e^d_{tj})}$$

$$\boxed{e^d_{tt'} = {h^d_t}^T W^d_{\text{attn}} h^d_{t'}}$$

- decoder attention output: $$c^d_t = \sum_{j=1}^{t-1} \alpha^d_{tj} h^d_j$$

Application 2: Generating Output: Mixture of Pointers and Softmax [8]

same idea as in application 1: mixture model of pointer network and normal softmax prediction (latent mixing parameter denoted as u here):

- compute mixing parameter (=probability of copying / pointing from input):

$$p(u_t = 1) = \sigma(W_u[h_t^d \| c_t^e \| c_t^d] + b_u)$$

- if not copying / pointing: use softmax prediction:

$$p(y_t | u_t = 0) = \text{softmax}(W_{\text{out}}[h_t^d \| c_t^e \| c_t^d] + b_{\text{out}})$$

- if copying: use encoder attention distribution weights:

$$p(y_t = x_i | u_t = 1) = \alpha_{ti}^e$$

- overall:

$$p(y_t) = p(u_t = 1)p(y_t | u_t = 1) + p(u_t = 0)p(y_t | u_t = 0)$$

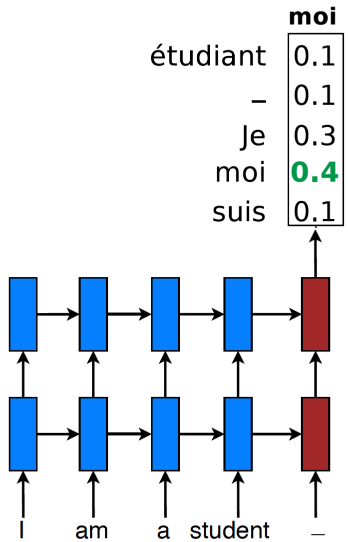
|| denotes the vector concatenation operator;

all probabilities are conditioned on $y_1, \ldots, y_{t-1}, x$ even when not explicitly stated

Large Vocabularies: Word Generation Problem

- e.g. traditional encoder-decoder approach with attention for NMT: involves softmax computation which is expensive, growing V makes this worse:

$$p_j = \frac{e^{u_i}}{\sum_{j=1}^{|V|} e^{u_j}}$$



- → restrict to modest V (e.g. 50k) („shortlist“):
  → lots of unknowns:

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

ground truth translation for full V

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

ground truth if using reduced V (e.g. 50k) → translation quality may suffer from many <unk>s

Word Generation Problem

- solution idea 1 (([Mnih & Teh, ICML'12], [Vaswani et al., EMNLP'13]) in [2]) (also see [9] section 2.2)) : only approximate the target word probability ((Mnih and Kavukcuoglu, 2013; Mikolov et al., 2013) in [9]) (e.g. via noise contrastive estimation ((Gutmann and Hyvarinen, 2010) in [9]) (estimate unnormalized probability AND normalization constant by learning to discriminate data from artificial noise))
  - disadvantage: need to create different noise samples per training example)

- solution idea 2 (([Morin & Bengio, AISTATS'05], [Mnih & Hinton, NIPS'09]) in [2]) (also see [9] section 2.2)) : classify words into hierarchical classes (tree structured vocabulary): factorize target probability of output word y_t $p(y_t|y_{<t}, x)$ into class probability $p(c_t|y_{<t}, x)$ and intra-class word probability $p(y_t|c_t, y_{<t}, x)$. → reduces the number of required dot-products
  - disadvantage: complex, sensitive to class hierarchy

- further disadvantage of these ideas: only faster at training time, often not faster at test time ([9] section 2.2), not GPU friendly

Large Vocabulary NMT [9]

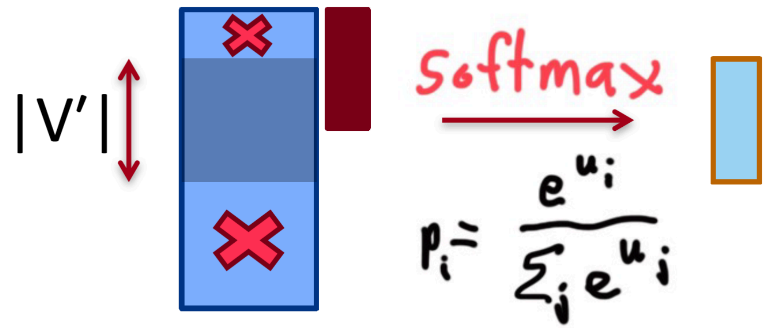
- [9]: idea: (related to idea 1): clever approximation of normalization constant:
  - Training: use a small subset $V' \subset V$ of the vocabulary at a time.
  - Testing: make smart choices on the set of possible translations.

- fast at train and test time
- GPU friendly

[9]: Jean, Cho, Memisevic, Bengio: On using very large target vocabulary for neural machine translation. (2015)

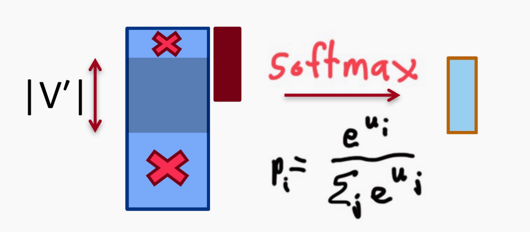
Large Vocabulary NMT [9] : Training

- each time train on small sub-vocabulary $|V'| \ll |V|$



- Partition training data in subsets:
  each subset has τ distinct target words, $|V'| = \tau$.

Large Vocabulary NMT [9] : Training



- source sequence: $x = (x_1, \ldots, x_T)$
- encoder hidden states: $h = (h_1, \cdots, h_T)$: $\quad h_t = f\left(x_t, h_{t-1}\right)$ (encoder RNN)
- target sequence: $y = (y_1, \cdots, y_{T'})$
- decoder hidden states: $z = (z_1, \ldots, z_{T'})$: $\quad z_t = g\left(y_{t-1}, z_{t-1}, c_t\right)$ (decoder RNN)
- encoder - decoder attention : $c_t = r\left(z_{t-1}, h_1, \ldots, h_T\right)$

$$= \sum_k \alpha_k h_k \quad ; \quad \alpha_t = \frac{\exp\left\{a\left(h_t, z_{t-1}\right)\right\}}{\sum_k \exp\left\{a\left(h_k, z_{t-1}\right)\right\}}$$

a : simple bilinear attention (1-layer FFNN)

- decoder output probability: $p(y_t \mid y_{<t}, x) = \frac{1}{Z} \exp\left\{\mathbf{w}_t^\top \phi\left(y_{t-1}, z_t, c_t\right) + b_t\right\}$

$$Z = \sum_{k: y_k \in V} \exp\left\{\mathbf{w}_k^\top \phi\left(y_{t-1}, z_t, c_t\right) + b_k\right\}$$

ϕ : e.g. 1-layer FFNN

- decoder output probability approximation: $p(y_t \mid y_{<t}, x)$

$$= \frac{\exp\left\{\mathbf{w}_t^\top \phi\left(y_{t-1}, z_t, c_t\right) + b_t\right\}}{\sum_{k: y_k \in V'} \exp\left\{\mathbf{w}_k^\top \phi\left(y_{t-1}, z_t, c_t\right) + b_k\right\}}$$

Large Vocabulary NMT [9] : Training

- Segment data: sequentially select examples: |V'| = 5. (choose in target language)

she loves cats
he likes dogs
cats have tails
dogs have tails
dogs chase cats
she loves dogs
cats hate dogs

V' = {she, loves, cats, he, likes}

Large Vocabulary NMT [9] : Training

- Segment data: sequentially select examples: |V’| = 5.

she loves cats
he likes dogs
cats have tails
dogs have tails
dogs chase cats
she loves dogs
cats hate dogs

V’ = {cats, have, tails, dogs, chase}

Large Vocabulary NMT [9] : Training

- Segment data: sequentially select examples: $|V'| \; = \; 5$.

she loves cats
he likes dogs
cats have tails
dogs have tails
dogs chase cats
she loves dogs
cats hate dogs

V' = {she, loves, dogs, cats, hate}

- in practice: $|V| = 500k, |V'| \; = \; 30k$ or $50k$.

Large Vocabulary NMT [9] : Testing: Candidate List

- **testing**:
  -- either use full V **or**
  -- also use limited V' (similar to training) but no correct translation available to guide selection of V' →

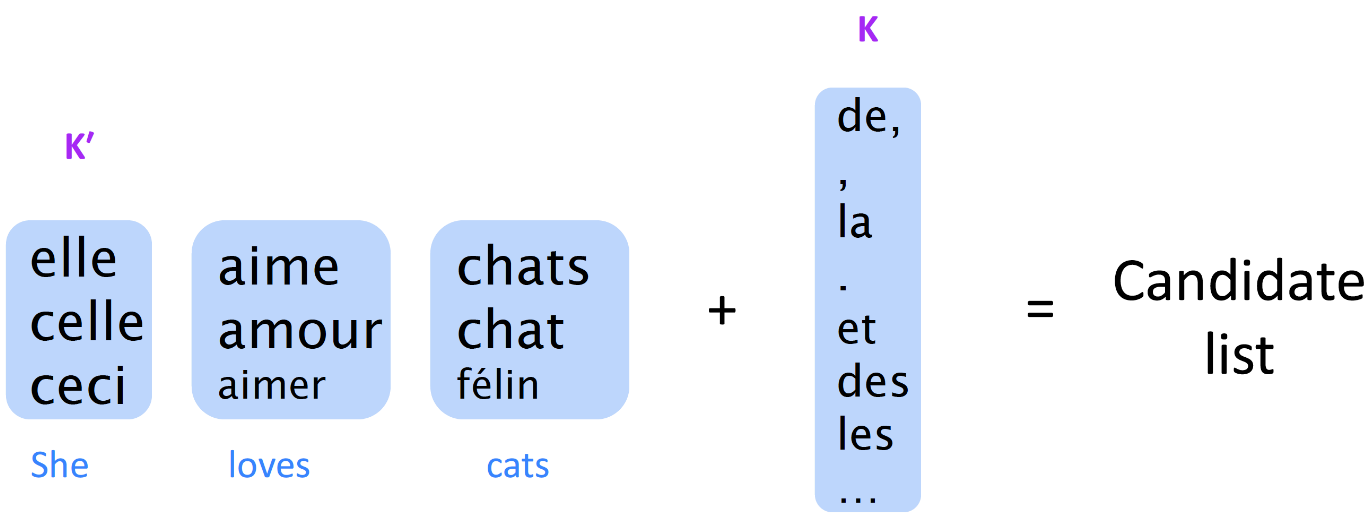
  - select K most frequent words (global unigram probability):

    de,
    ,
    la
    .
    et
    des
    les
    ...

  - AND with the help of some alignment model (e.g. via attentions) : select K' candidate target words per choice word from training data:

She	loves	cats
elle	aime	chats
celle	amour	chat
ceci	aimer	félin

Large Vocabulary NMT [9] : Testing: Candidate List



- use K and K' to approximate involved probabilities (←→ normalization constants)
- in practice: $\mathrm{K}' = 10 \text{ or } 20, K = 15k, 30k, \text{or } 50k.$

NMT for More "Complicated" Languages

- "Copy" mechanisms (*copy source words into target sentence, if no translation is possible*) are not sufficient (see [12]):
  - Transliteration (mapping characters): Christopher → Kryštof
  - Multi-word alignment: Solar system → Sonnensystem

- Need to handle large, open vocabulary
  - rich morphology: nejneobhospodařovávatelnějšímu == Czech: "to the worst farmable one"
  - compounds: Donaudampfschiffahrtsgesellschaftskapitänsmütze == German: "Danube steamship company captain's hat"

- Informal spelling: gooooood morning !!!!!

→ Need to be able to **operate at sub-word levels**!

Sub-Word NMT [10]

- need to work on sub-word level → use smaller units (characters, bytes etc.)
  OR smaller units AND words in models

- → some more papers! ☺

e.g. argument for smaller units:
cases translatable by a
competent translator, even if she doesn't know the word:

Transcription / Transliteration (e.g. names):

Barack Obama (English; German)
Барак Обама (Russian)
バラク・オバマ (ba-ra-ku o-ba-ma) (Japanese)

cognates and loanwords

claustrophobia (English)
Klaustrophobie (German)
Клаустрофобия (Klaustrofobiâ) (Russian)

morphologically complex words

solar system (English)
Sonnensystem (Sonne + System) (German)
Naprendszer (Nap + Rendszer) (Hungarian)

analogous formations:

sweetish: sweet + ish → süßlich
funtoish: funto (??) + ish → funtolich

[10]

Byte Pair Encoding [10]

- split text into smaller units than words → better compression (→ longer text-segments can be efficiently encoded), smaller "vocabulary" size)

- [10]: use idea of Byte Pair Encoding (encode most frequent byte pair with a new unused byte) on characters:
  - start with vocabulary of characters + end-of-word-character.
  - for most frequent character n-gram pair in words: create new n-gram.
  - iterate. → word segmentation into character n-grams

tf	word
5	low
2	lower
6	newest
3	widest
5	despite

l, o, w, e, r, n, w, s, t, i, d, p

all characters in vocab

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. ACL 2016.

Byte Pair Encoding [10]

- split text into smaller units than words → better compression (→ longer text-segments can be efficiently encoded), smaller "vocabulary" size)

- [10]: use idea of Byte Pair Encoding (encode most frequent byte pair with a new unused byte) on characters:
  - start with vocabulary of characters + end-of-word-character.
  - for most frequent character n-gram pair in words: create new n-gram.
  - iterate. → word segmentation into character n-grams

tf	word
5	low
2	lower
6	new**es**t
3	wid**es**t
5	d**es**pite

l, o, w, e, r, n, w, s, t, i, d, p, **es**

all 2-gram **es** (with frequency 14)

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. ACL 2016.

Byte Pair Encoding [10]

- split text into smaller units than words → better compression (→ longer text-segments can be efficiently encoded), smaller "vocabulary" size)

- [10]: use idea of Byte Pair Encoding (encode most frequent byte pair with a new unused byte) on characters:
  - start with vocabulary of characters + end-of-word-character.
  - for most frequent character n-gram pair in words: create new n-gram.
  - iterate. → word segmentation into character n-grams

tf	word
5	low
2	lower
6	new**est**
3	wid**est**
5	despite

l, o, w, e, r, n, w, s, t, i, d, p, es, **est**

all 3-gram **est** (with frequency 9)

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. ACL 2016.

Byte Pair Encoding [10]

- split text into smaller units than words → better compression (→ longer text-segments can be efficiently encoded), smaller "vocabulary" size)

- [10]: use idea of Byte Pair Encoding (encode most frequent byte pair with a new unused byte) on characters:
  - start with vocabulary of characters + end-of-word-character.
  - for most frequent character n-gram pair in words: create new n-gram.
  - iterate. → word segmentation into character n-grams

tf	word
5	**lo**w
2	**lo**wer
6	newest
3	widest
5	despite

l, o, w, e, r, n, w, s, t, i, d, p, es, est, **lo**

all 2-gram **lo** (with frequency 7)

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. ACL 2016.

Byte Pair Encoding [10]

- split text into smaller units than words → better compression (→ longer text-segments can be efficiently encoded), smaller "vocabulary" size)

- [10]: use idea of Byte Pair Encoding (encode most frequent byte pair with a new unused byte) on characters:
  - start with vocabulary of characters + end-of-word-character.
  - for most frequent character n-gram pair in words: create new n-gram.
  - iterate. → word segmentation into character n-grams

tf	word
5	low
2	lower
6	newest
3	widest
5	despite

l, o, w, e, r, n, w, s, t, i, d, p, es, est, lo

→ lower = lo-w-e-r
→ newest = n-e-w-est

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. ACL 2016.

Byte Pair Encoding [10]

- split text into smaller units than words → better compression (→ longer text-segments can be efficiently encoded), smaller "vocabulary" size)

- [10]: use idea of Byte Pair Encoding (encode most frequent byte pair with a

using this approach on a "vanilla" encoder-decoder NMT with attention (as described before) (Bahdanau et al., 2015 in [10]): achieved top scores in WMT 2016 conference

2	lower
6	newest
3	widest
5	despite

→ lower = lo-w-e-r
→ newest = n-e-w-est

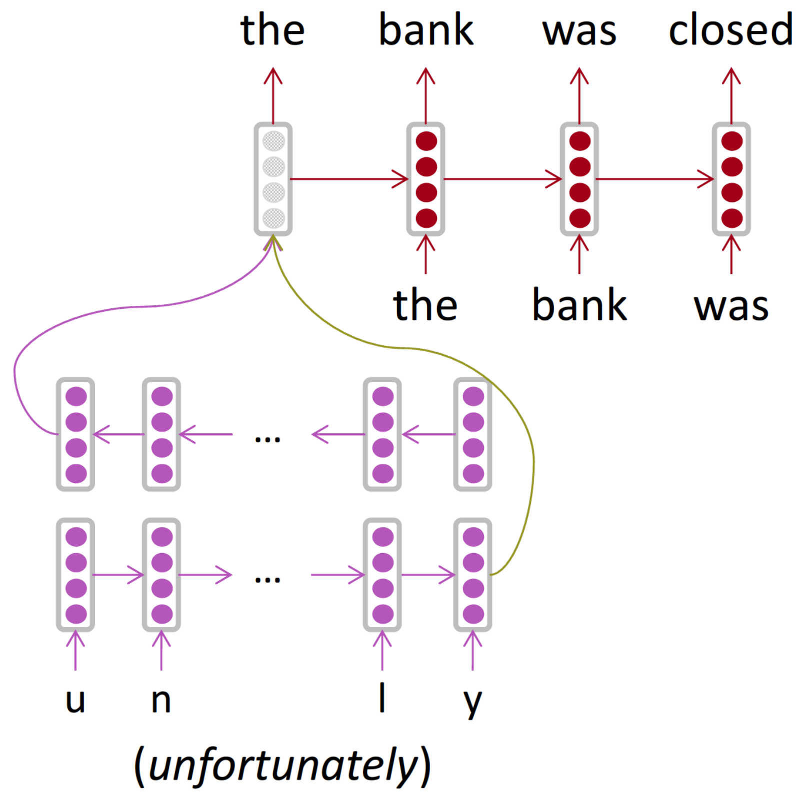
[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. ACL 2016.

Character-Based Bi-LSTM [11]

[11]: idea: use character based bi-LSTM model to compute word-embeddings (e.g. for a recurrent language model) that can e.g.

- account for orthographic ←→ functional relatedness (*king, kings; queen queens*)
- handle compounds (*Frenchification, French, –ification*)
- etc.

  - useful: dealing with morphemes such as -ly, -ing, -ed, pre-, -s
  - less useful cases: batter ←→ butter, coarse ←→ course



	Fusional			Agglutinative	
Perplexity	EN	PT	CA	DE	TR
5-gram KN	70.72	58.73	39.83	59.07	52.87
Word	59.38	46.17	35.34	43.02	44.01
C2W	**57.39**	**40.92**	**34.92**	**41.94**	**32.88**
#Parameters					
Word	4.3M	4.2M	4.3M	6.3M	5.7M
C2W	**180K**	**178K**	**182K**	**183K**	**174K**

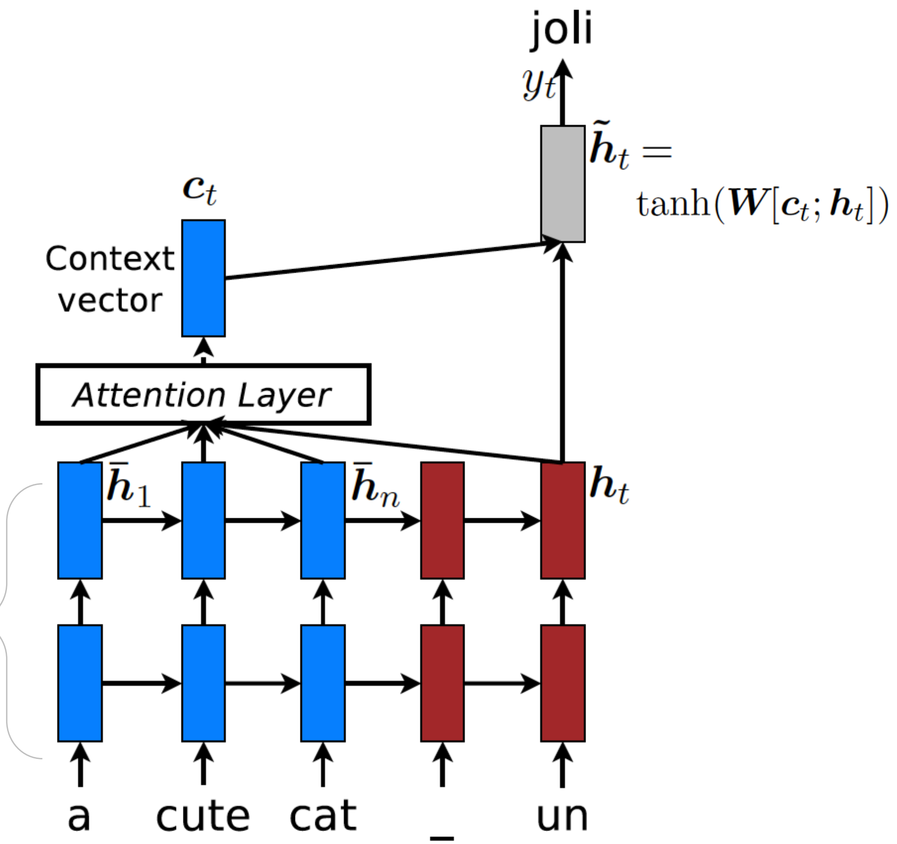
Language modeling on various languages of two types

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

Hybrid Word-Character NMT [12]

- [12]: idea: best-of-both-worlds architecture:
  - translate mostly at the word level
  - only go to the character level when needed

- **basic model**:
  - two LSTM layers
  - attention used on top layer hidden states ($\bar{h}_1, \ldots, \bar{h}_N$ and h_t)
  - similarity between $\bar{h}_N$ and h_t: multiplicative (bilinear)

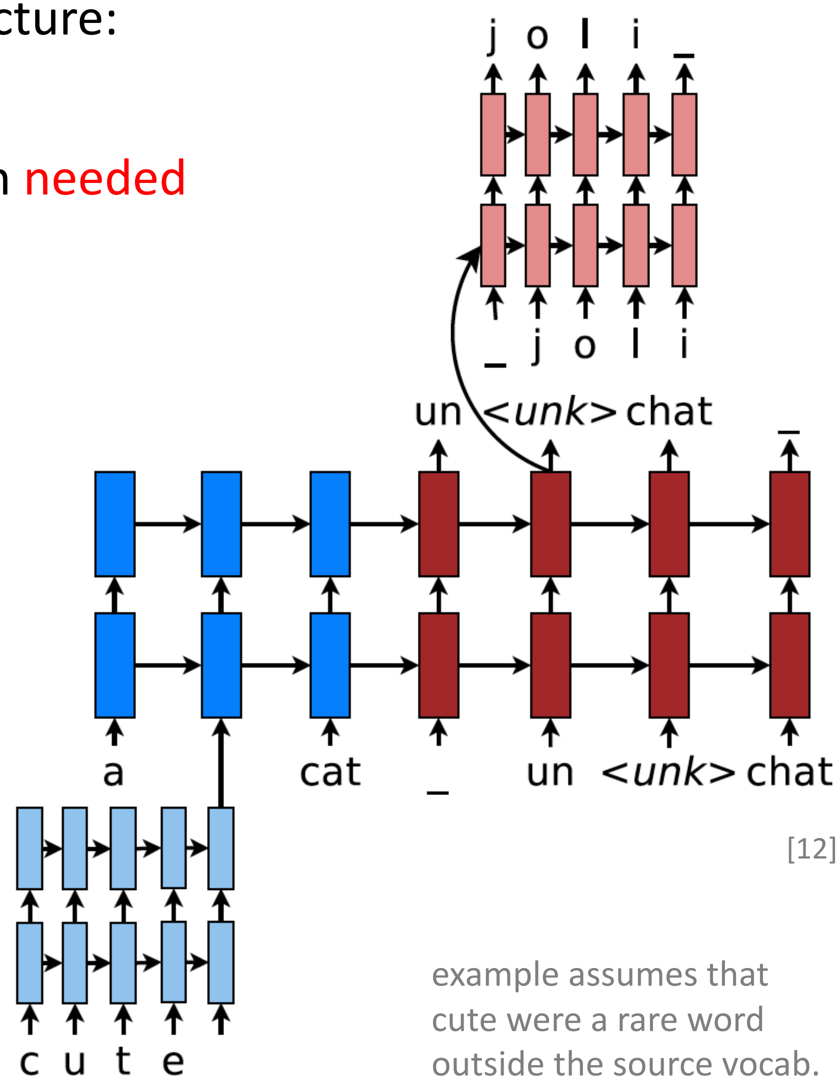


[12]

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

Hybrid Word-Character NMT [12]

- [12]: idea: best-of-both-worlds architecture:
  - translate mostly at the word level
  - only go to the character level when needed



[12]

example assumes that cute were a rare word outside the source vocab.

- **hybrid model**:
  - basic word model for V frequent „known“ words
  - varying V: blend word-level and character-level treatment
  - $\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}[\boldsymbol{c}_t; \boldsymbol{h}_t])$ predicts <unk> → resort to char-level sub-model (use $\breve{\boldsymbol{h}}_t = \tanh(\breve{\boldsymbol{W}}[\boldsymbol{c}_t; \boldsymbol{h}_t])$ as h_0
  - test time: use beam search: generate several candidate translations; if contains <unk> invoke char-level model (again beam search)

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

Hybrid Word-Character NMT [12] : Performance

- NMT state of the art in 2016

source	As the Reverend ***Martin Luther King Jr.*** said ***fifty years ago*** :
human	Jak ***před padesáti lety*** řekl reverend ***Martin Luther King Jr*** . :
word	Jak řekl reverend Martin <*unk*> King <*unk*> před padesáti lety :
word	Jak řekl reverend ***Martin Luther King*** **řekl** před padesáti lety :
char	Jako reverend ***Martin Luther*** **král říkal** před padesáti lety :
hybrid	Jak před <*unk*> lety řekl <*unk*> Martin <*unk*> <*unk*> <*unk*> :
hybrid	Jak ***před padesáti lety*** řekl reverend ***Martin Luther King*** Jr. :
source	Her ***11-year-old*** daughter , ***Shani Bart*** , said it felt a " little bit ***weird*** " [
human	Její ***jedenáctiletá*** dcera ***Shani Bartová*** prozradila , že " je to trochu ***zvláš***
word	Její <*unk*> dcera <*unk*> <*unk*> řekla , že je to " trochu divné " , [..] vr
word	Její **11-year-old** dcera ***Shani*** **,** řekla , že je to " trochu divné " , [..] vrací
char	Její ***jedenáctiletá*** dcera , ***Shani Bartová*** , říkala , že cítí trochu divně , [..
hybrid	Její <*unk*> dcera , <*unk*> <*unk*> , řekla , že cítí " trochu <*unk*> " , [..
hybrid	Její ***jedenáctiletá*** dcera , **Graham** Bart , řekla , že cítí " trochu divný " ,

correct, **wrong**, and close [12]

Bibliography

(2) Richard Socher et al: “CS224n: Natural Language Processing with Deep Learning”, Lecture Materials (slides and links to background reading) http://web.stanford.edu/class/cs224n/ (URL, May 2018), 2018

(3) https://youtu.be/K-HfpsHPmvw (URL, Aug 2018) (in [2])

(4) Rico Sennrich (University of Edinburgh): Neural Machine Translation: Breaking the Performance Plateau, Talk July 2016; http://www.meta-net.eu/events/meta-forum-2016/slides/09_sennrich.pdf (URL, Aug 2018) (in [2])

(5) https://hackernoon.com/bias-sexist-or-this-is-the-way-it-should-be-ce1f7c8c683c (URL, Aug 2018) (in [2])

(6) http://languagelog.ldc.upenn.edu/nll/?p=35120#more-35120 (URL, Aug 2018) (in [2])

(7) Merity, Stephen, Caiming Xiong, James Bradbury, and Richard Socher. "Pointer sentinel mixture models." arXiv preprint arXiv:1609.07843 (2016).

Bibliography

(8) Paulus, R., Xiong, C. and Socher, R., (2017). A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

(9) Jean, S., Cho, K., Memisevic, R., & Bengio, Y. (2015). On using very large target vocabulary for neural machine translation. arXiv preprint arXiv:1412.2007. and ACL 2015

(10) Sennrich, Rico, Barry Haddow, and Alexandra Birch (2015). "Neural machine translation of rare words with subword units." arXiv preprint arXiv:1508.07909.

(11) Ling, Luís, Marujo, Astudillo, Amir, Dyer, Black, Trancoso (2015). Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. EMNLP’15 and arXiv preprint arXiv:1508.02096

(12) Thang Luong and Chris Manning (2016). Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models. ACL 2016 and arXiv preprint arXiv:1604.00788.

Recommendations for Studying

- minimal approach:

  work with the slides and understand their contents! Think beyond instead of merely memorizing the contents

- standard approach:

  minimal approach + read the corresponding first paragraphs of a choice of the papers corresponding to the discussed example systems:

  - [7] sections 1 & 2,
  - [8] sections 1 & 2 & 3
  - [9] sections 2 & 3
  - [10] sections 1 & 2 & 3
  - [11] sections 1 & 2
  - [12]* sections 1 & 2 & 3 & 4

  * indicates an especially nicely legible paper for our learning purpose

- interested / deeply interested student's approach:

  standard approach + study the few omitted elements of the corresponding lecture slides from [2] + read all of the recommended background reading of [2] for lecture 11