Natural Language Processing with Deep Learning

CS224N/Ling284

Lecture 11:
Paying attention to attention and

Tips and Tricks for large MT

Richard Socher

Thanks for your Feedback!

What do you most want to learn about in the remaining lectures?

- *"More cutting edge research topics in NLP & DL."*
- *"Solutions/approaches to core NLP problems, variations in the the techniques used"*
- *"More state of the art neural networks"*
- *"More neural tricks"*
- *"More state of the art techniques."*
- *"State-of-the-art neural network architectures"*

See Piazza for feedback form – continued feedback welcome!

Congrats

- The basic/theoretical/mathematical part of the class is over
- Now, we can have fun with recent, state-of-the-art ideas

- You can start hacking up real stuff now

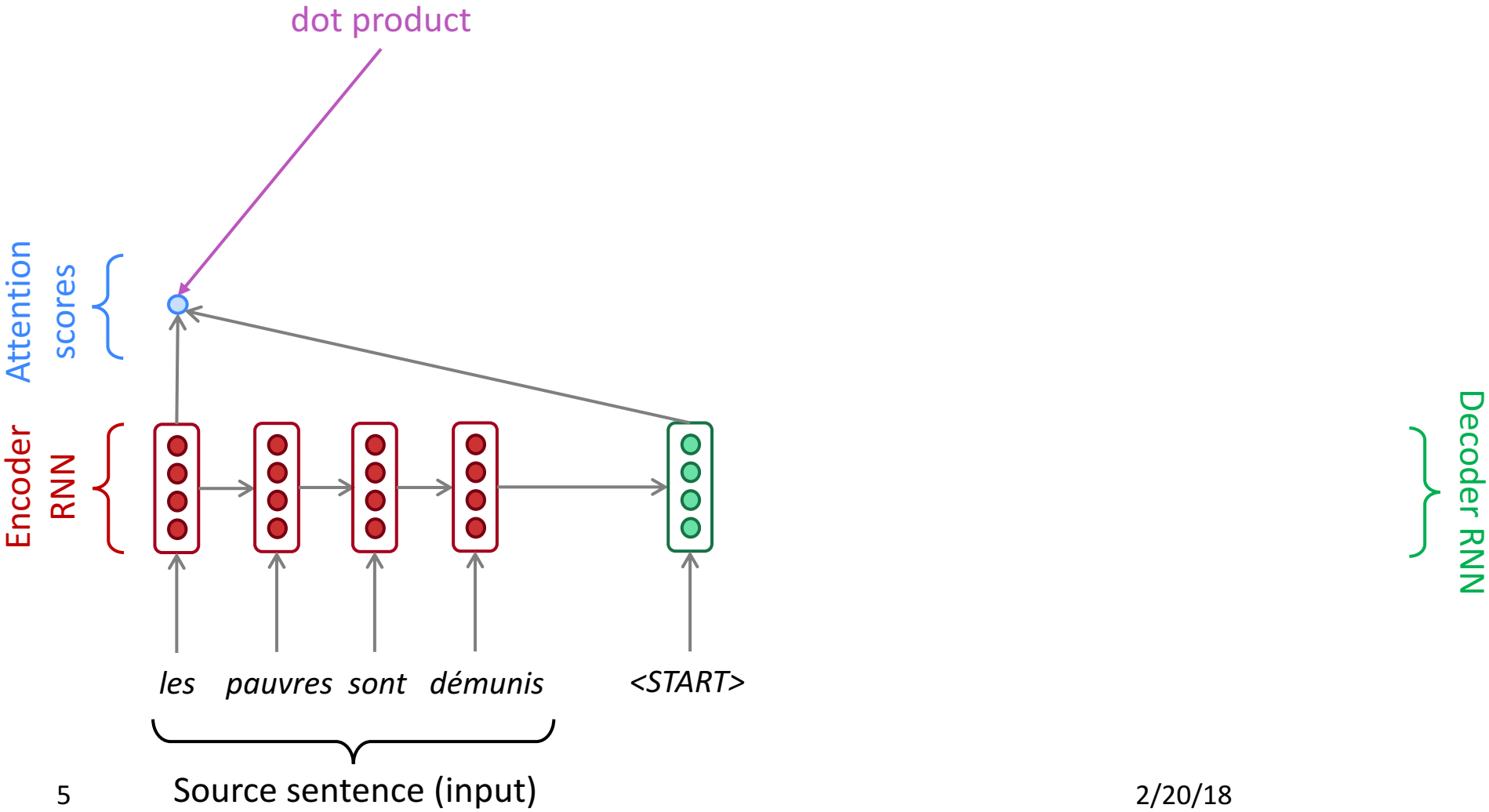
- Today, we will cover more attention details and tips and tricks to make bigger models and systems work

Overview

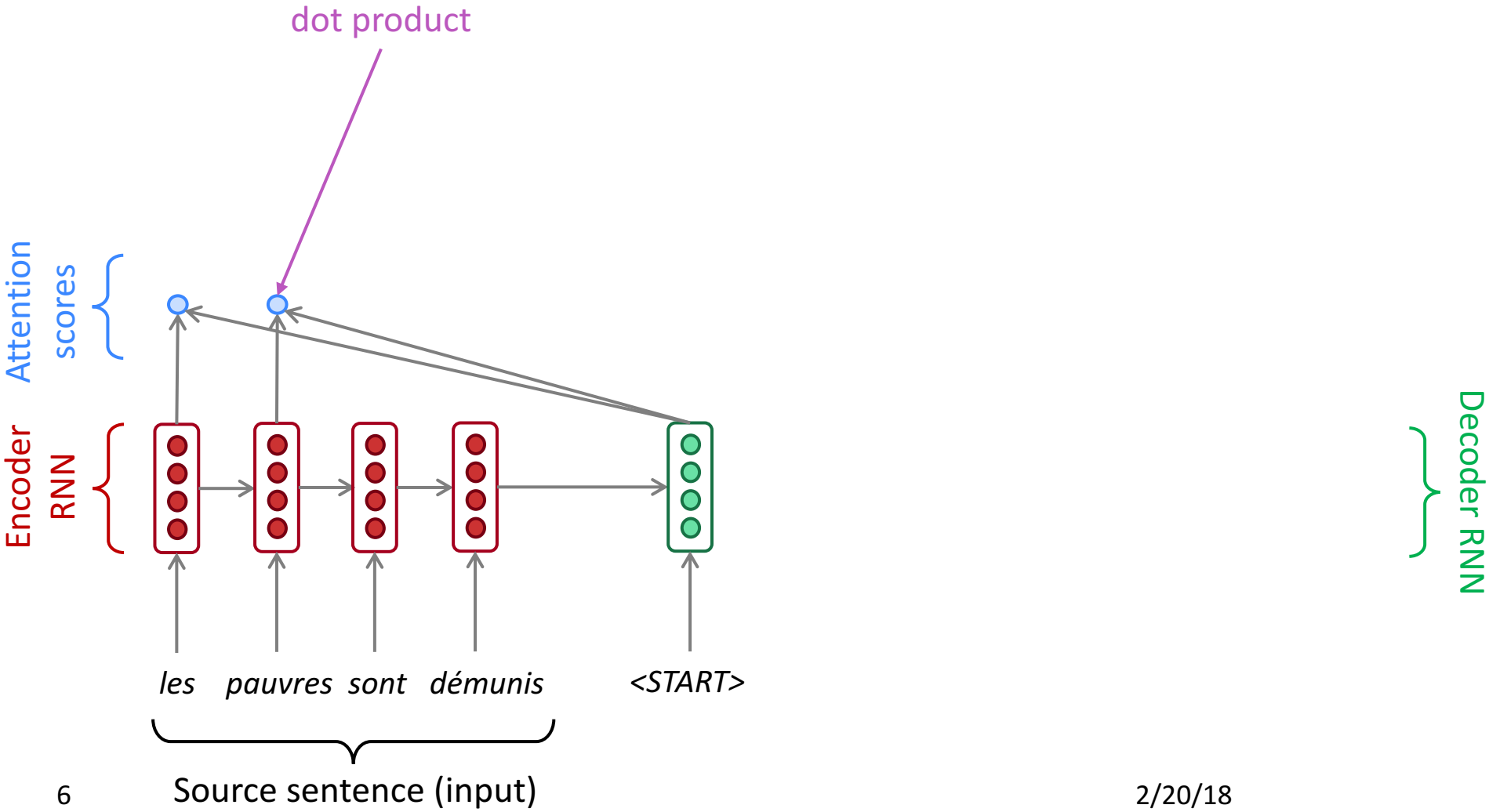
- More types and uses of attention
  - Refresh and math
  - Pointer-sentinel model
  - Self-attention/intra-detention for summarization

- Practical tricks on how to deal with large output vocabularies in neural machine translation (and other tasks)
  - UNKs
  - Vocab reduction
  - Subword Units

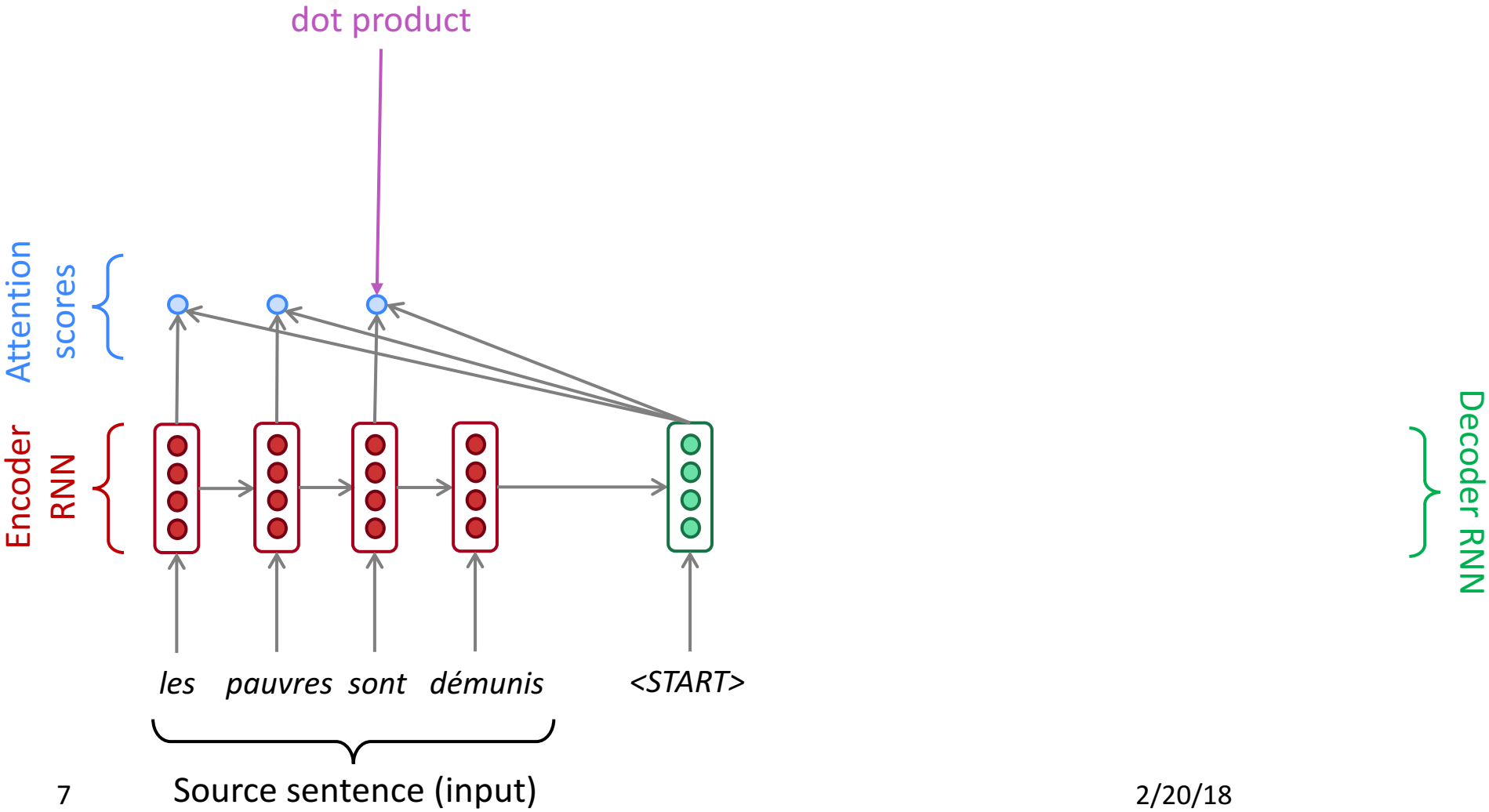
Recap: Sequence-to-sequence with attention



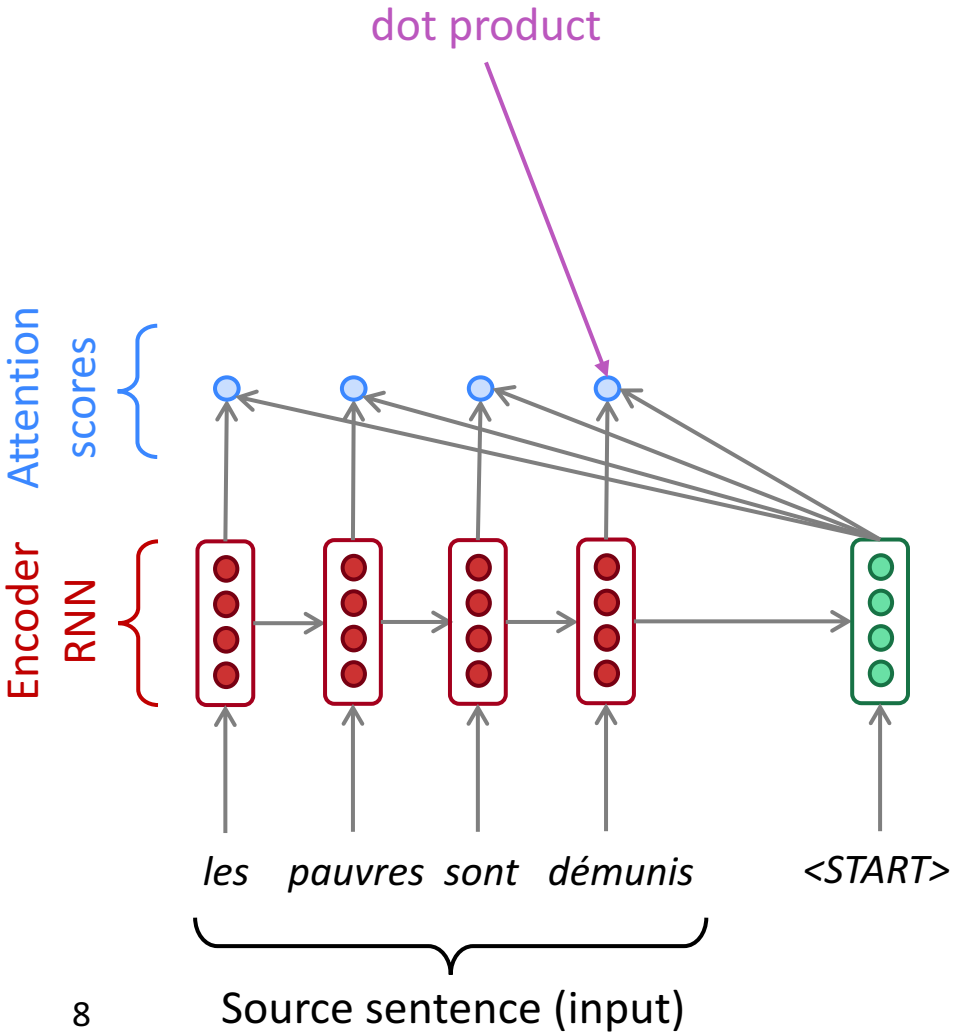
Recap: Sequence-to-sequence with attention



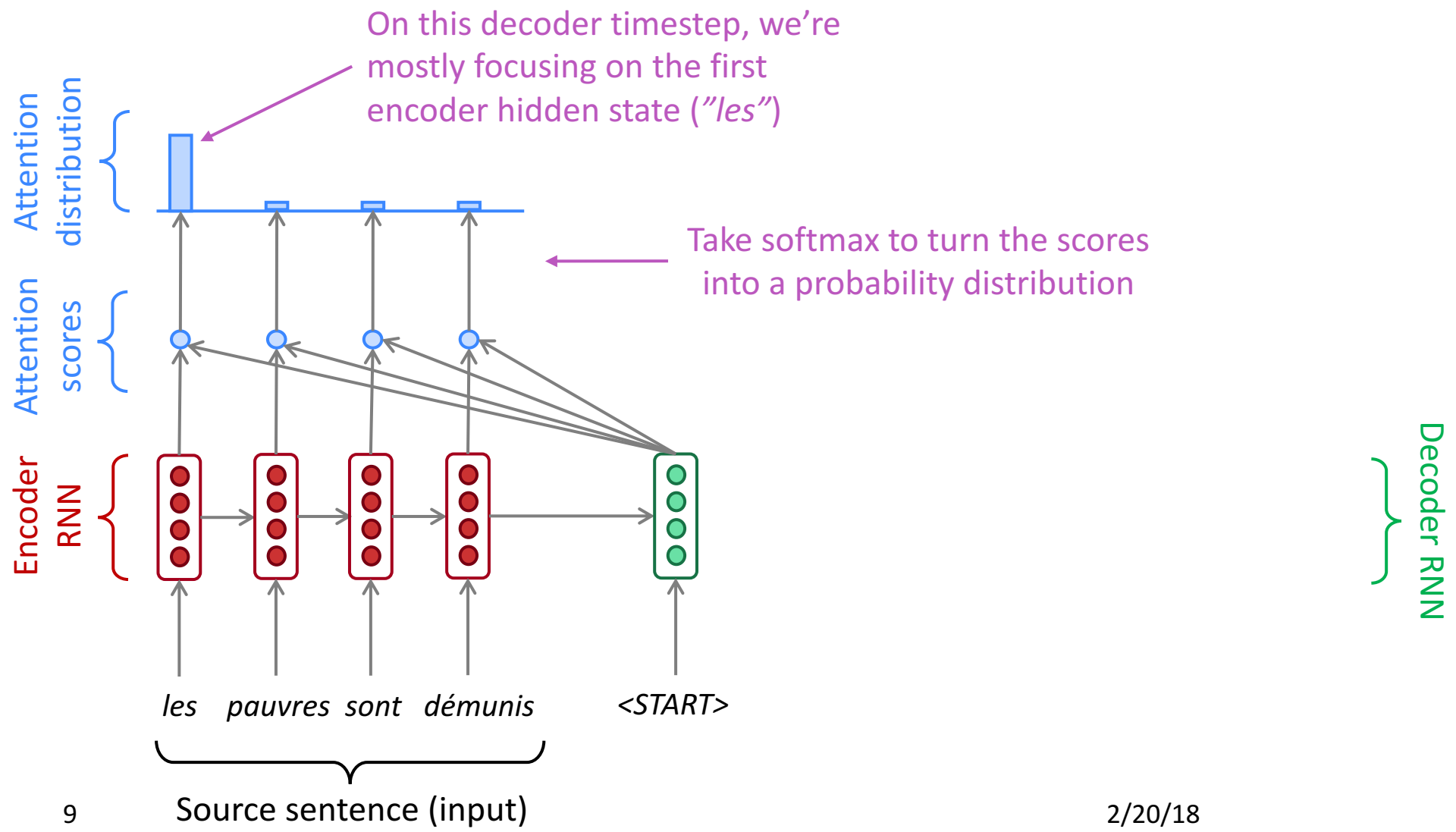
Recap: Sequence-to-sequence with attention



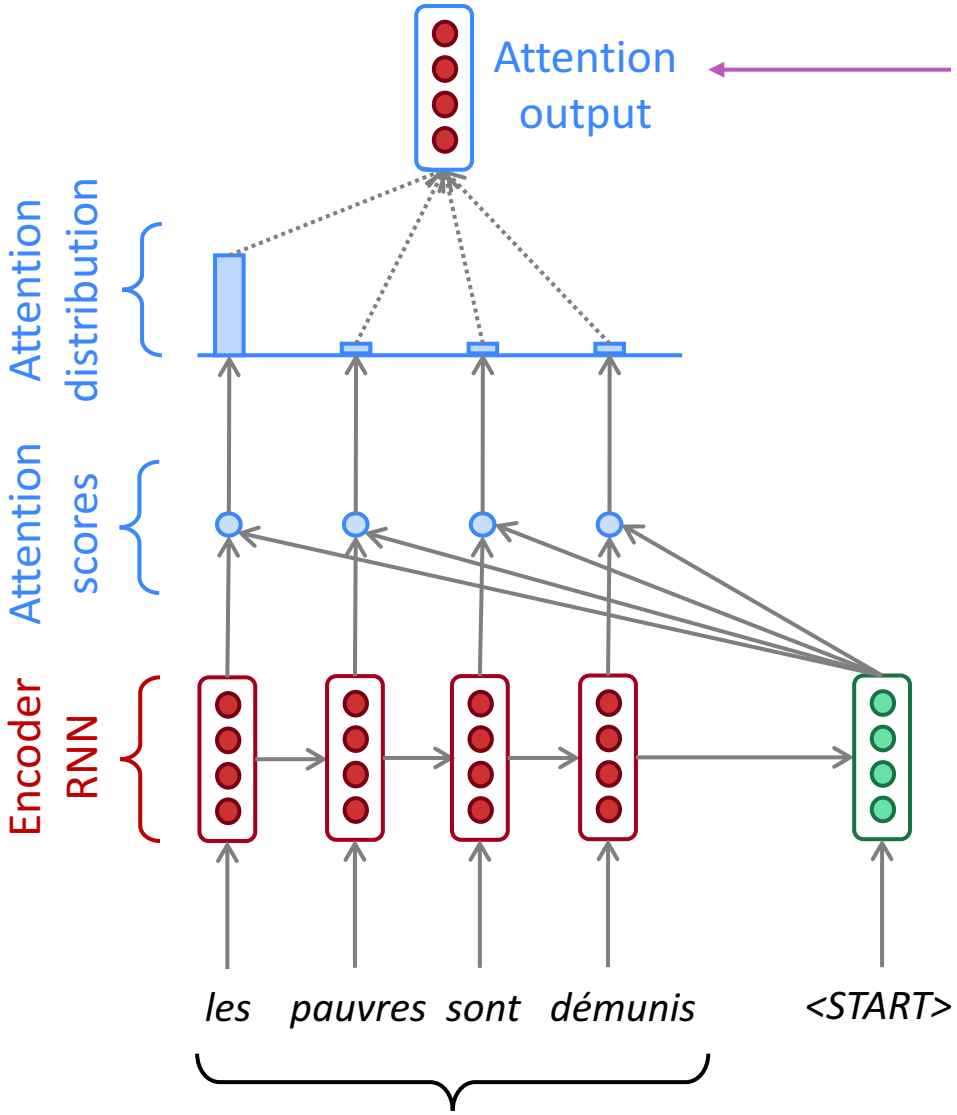
Recap: Sequence-to-sequence with attention



Recap: Sequence-to-sequence with attention



Recap: Sequence-to-sequence with attention

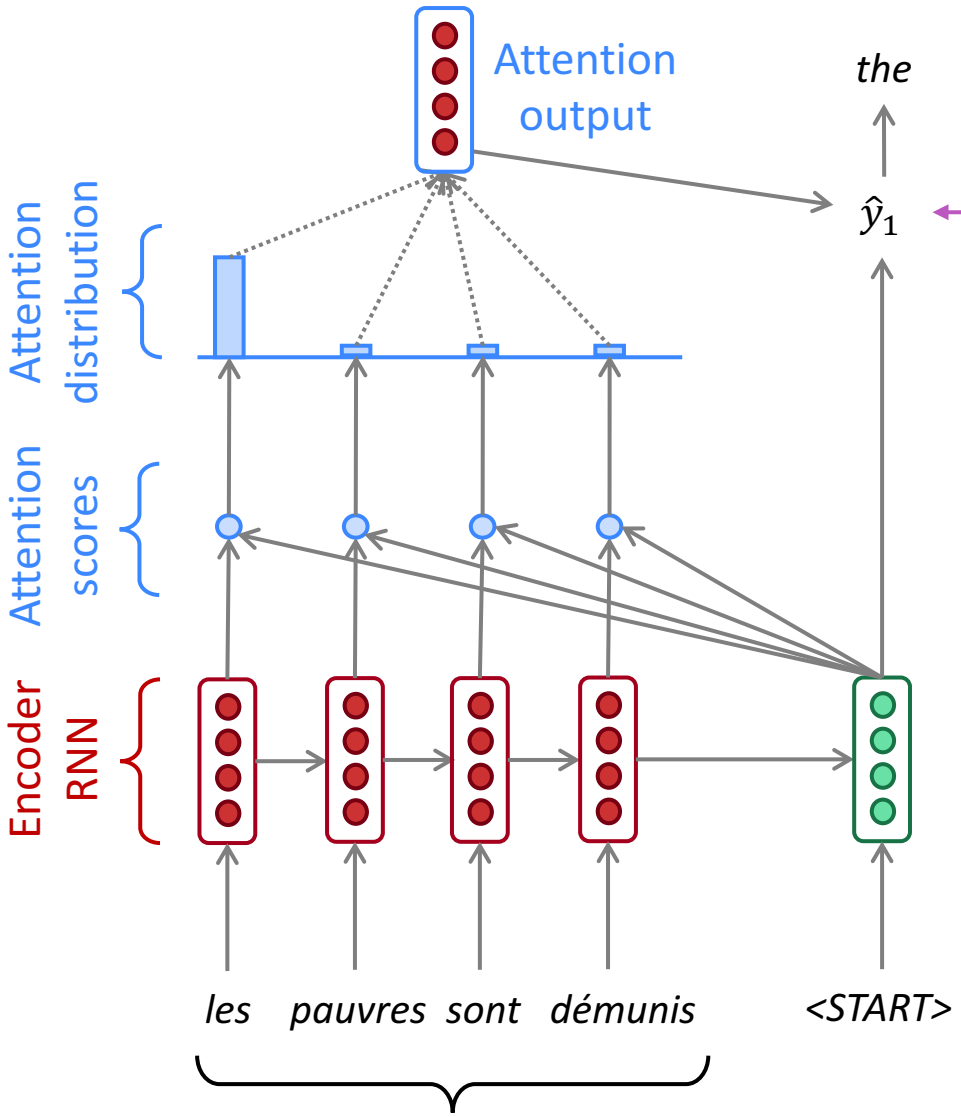


Use the attention distribution to take a **weighted sum** of the encoder hidden states.

The attention output mostly contains information the hidden states that received high attention.

Decoder RNN

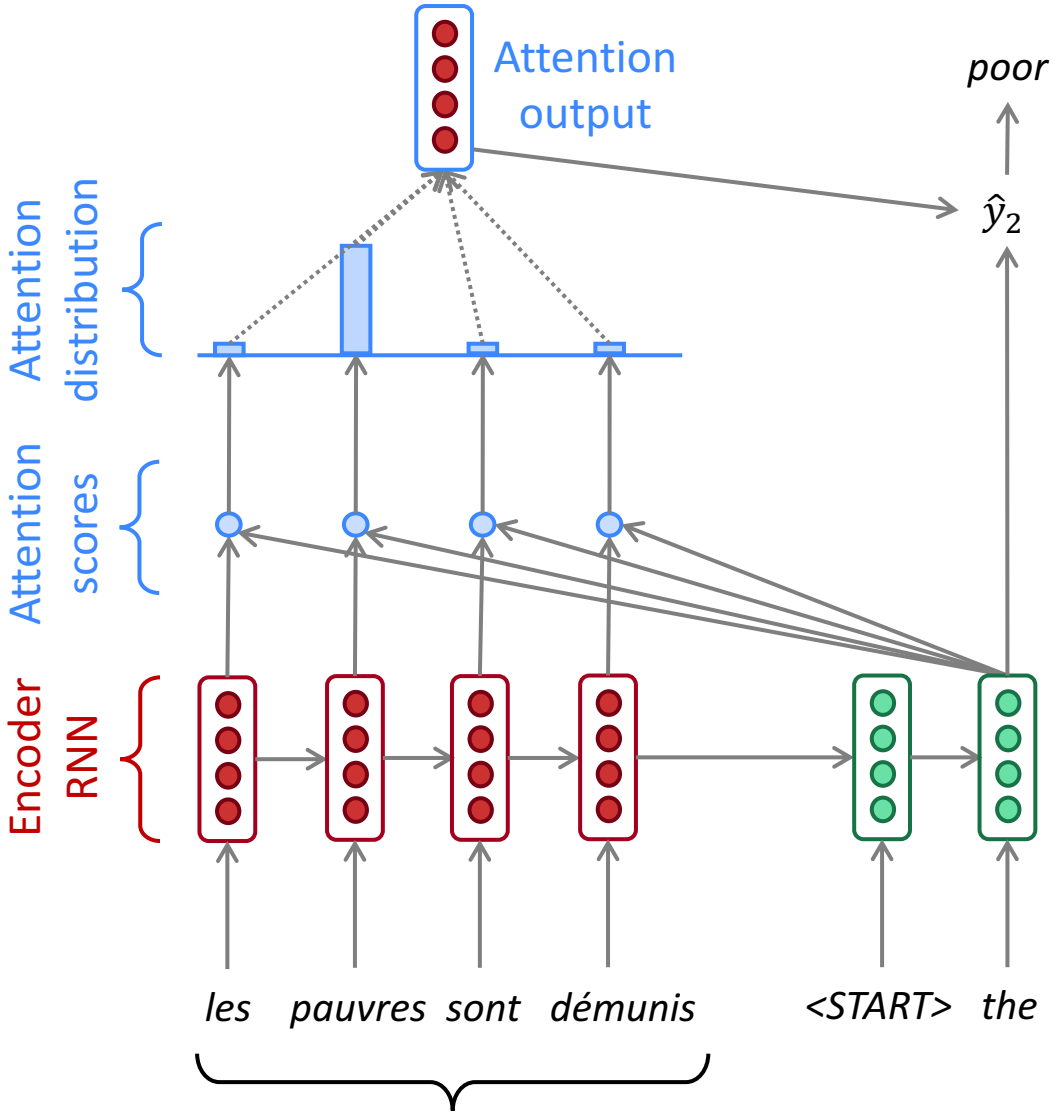
Recap: Sequence-to-sequence with attention



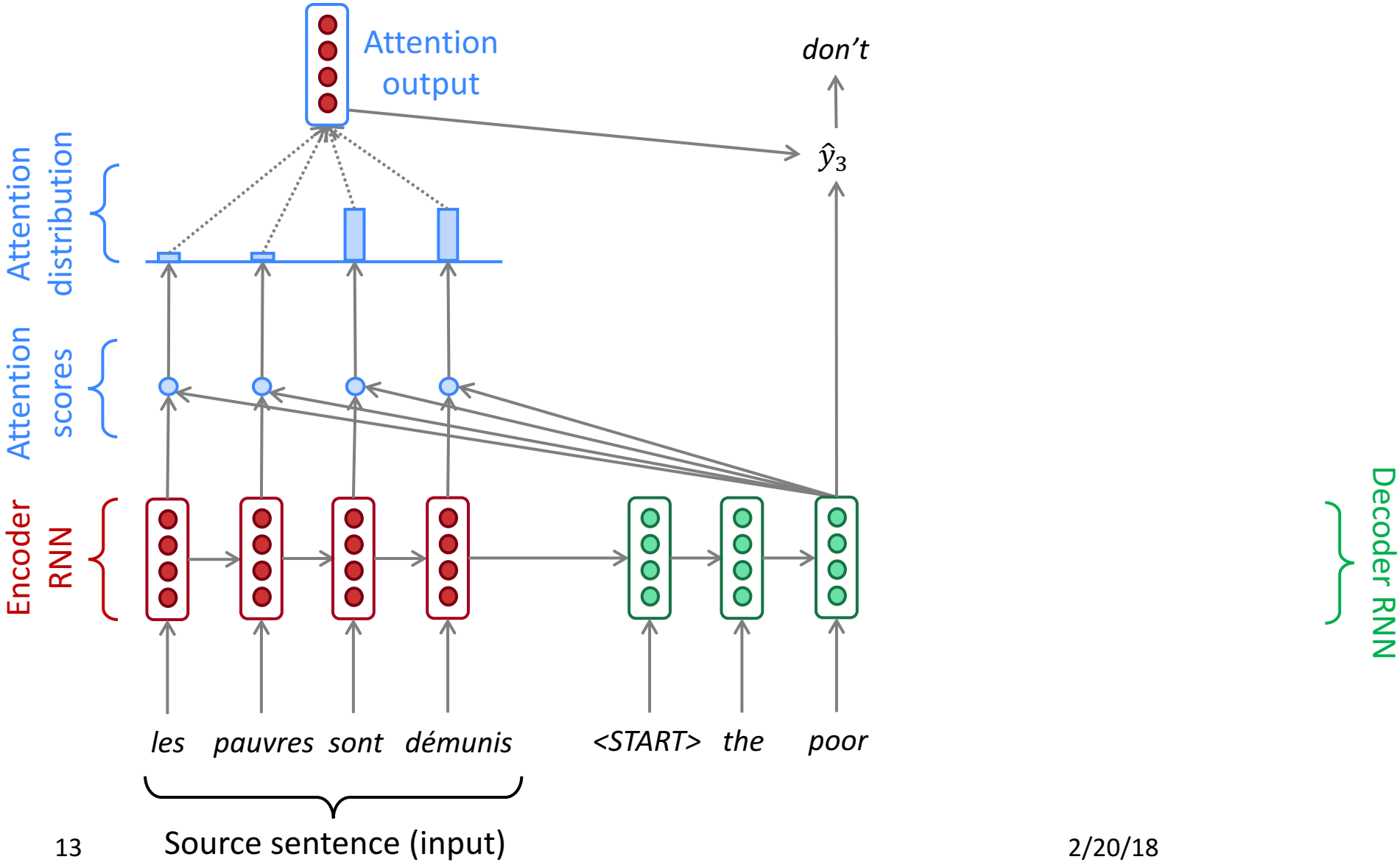
Concatenate attention output with decoder hidden state, then use to compute $\hat{y}_1$ as before

Decoder RNN

Recap: Sequence-to-sequence with attention



Recap: Sequence-to-sequence with attention



Recap: Basic Attention equations

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores e^t for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

- We take softmax to get the attention distribution α^t for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \text{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

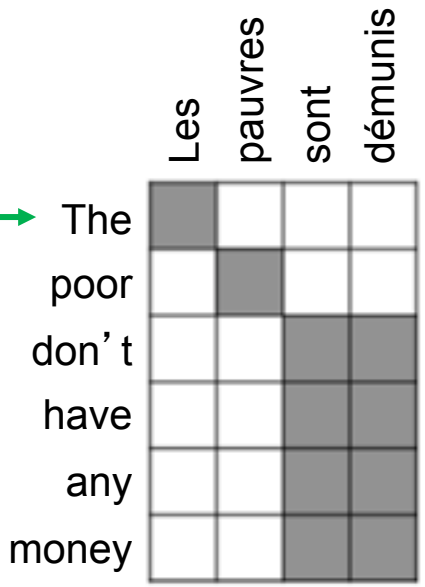
$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

Recap: Attention is great

- Attention significantly improves NMT performance
  - It's very useful to allow decoder to focus on certain parts of the source
- Attention solves the bottleneck problem
  - Attention allows decoder to look directly at source; bypass bottleneck
- Attention helps with vanishing gradient problem
  - Provides shortcut to faraway states
- Attention provides some interpretability
  - By inspecting attention distribution, we can see what the decoder was focusing on
  - We get alignment for free!
  - This is cool because we never explicitly trained an alignment system
  - The network just learned alignment by itself



Attention is a *general* Deep Learning technique

- Last time: We saw that attention is a great way to improve the sequence-to-sequence model for Machine Translation.
- However: Today we'll see attention is applied to many architectures (not just seq2seq) and many tasks (not just MT)

- **More general definition of attention**:
  - Given a set of vector *values*, and a vector *query*, **attention** is a technique to compute a weighted sum of the values, dependent on the query.*
- We sometimes say that the query *attends to* the values.
- For example, in the seq2seq + attention model, each decoder hidden state *attends to* the encoder hidden states.

** Note: This is slightly different to the terminology in the default final project code. See Piazza post for more details.*

Attention is a *general* Deep Learning technique

More general definition of attention:

Given a set of vector *values*, and a vector *query*, **attention** is a technique to compute a weighted sum of the values, dependent on the query.

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

There are *several* attention variants

- We have some *values* $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and a *query* $\boldsymbol{s} \in \mathbb{R}^{d_2}$

- Attention always involves computing the *attention output* $\boldsymbol{a} \in \mathbb{R}^{d_1}$ (sometimes called the *context vector*) from the *attention scores* $\boldsymbol{e} \in \mathbb{R}^N$ (or *attention logits*) like so:

$$\alpha = \text{softmax}(\boldsymbol{e}) \in \mathbb{R}^N \qquad \text{(take softmax)}$$

$$\boldsymbol{a} = \sum_{i=1}^{N} \alpha_i \boldsymbol{h}_i \in \mathbb{R}^{d_1} \qquad \text{(take weighted sum)}$$

- However, there are several ways you can compute $\boldsymbol{e} \in \mathbb{R}^N$

Attention variants

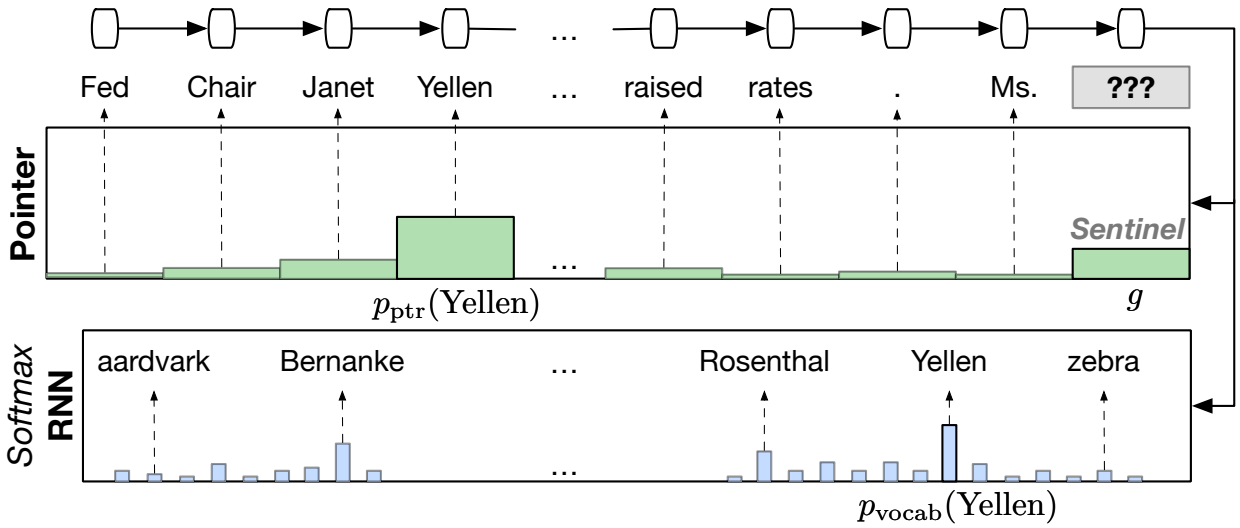
There are several ways you can compute $\boldsymbol{e} \in \mathbb{R}^N$ from $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and $\boldsymbol{s} \in \mathbb{R}^{d_2}$:

- Basic dot-product attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$
  - Note: this assumes $d_1 = d_2$
  - This is the version we saw earlier
- Multiplicative attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$
  - Where $\boldsymbol{W} \in \mathbb{R}^{d_2 \times d_1}$ is a weight matrix
- Additive attention: $\boldsymbol{e}_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$
  - Where $\boldsymbol{W}_1 \in \mathbb{R}^{d_3 \times d_1}, \boldsymbol{W}_2 \in \mathbb{R}^{d_3 \times d_2}$ are weight matrices and $\boldsymbol{v} \in \mathbb{R}^{d_3}$ is a weight vector

More information: http://ruder.io/deep-learning-nlp-best-practices/index.html#attention

Attention application: Pointing to words for language modeling

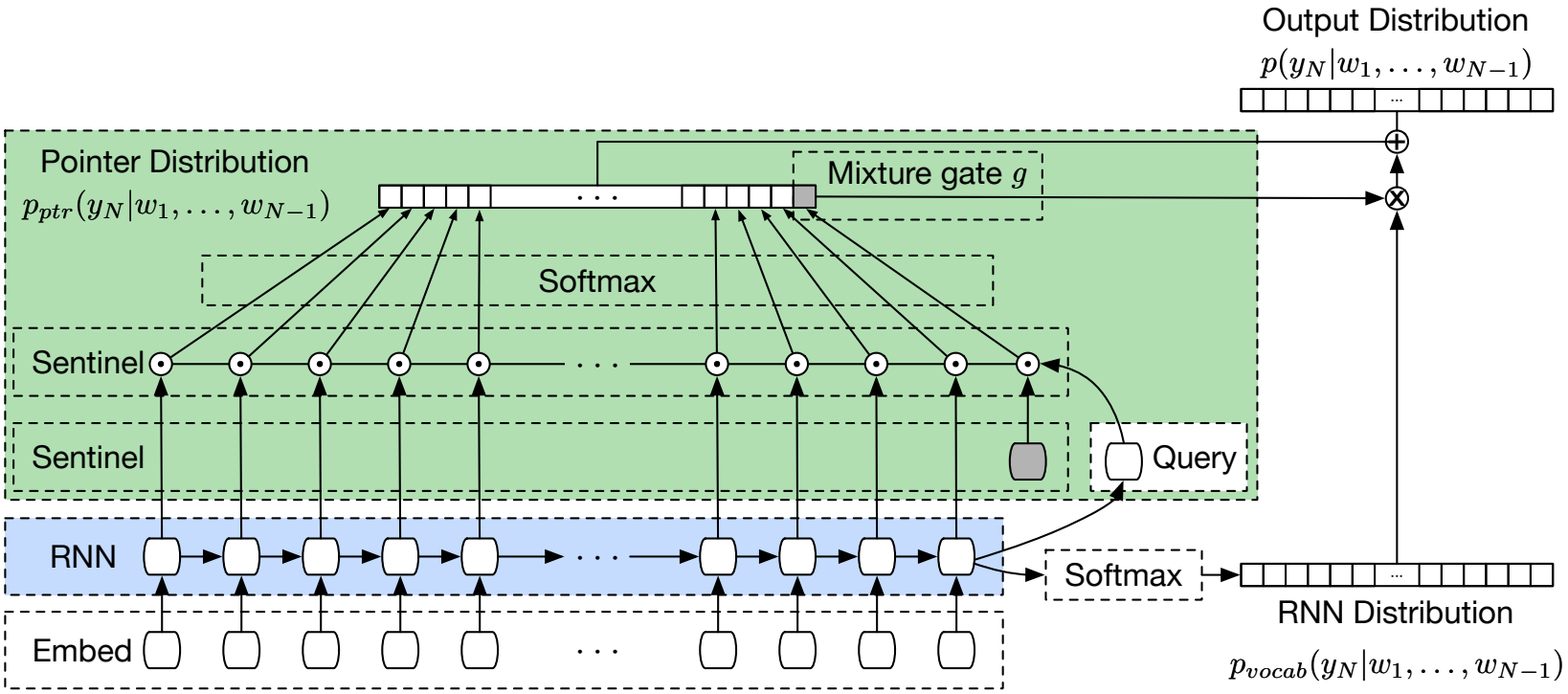
- Idea: Mixture Model of softmax and pointers:



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

- Pointer Sentinel Mixture Models by Stephen Merity, Caiming Xiong, James Bradbury, Richard Socher

Pointer-Sentinel Model - Details



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

$$z_i = q^T h_i, \qquad p_{\text{ptr}}(w) = \sum_{i \in I(w,x)} a_i,$$

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

Pointer Sentinel for Language Modeling

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**

Attention application: Intra-Decoder attention for Summarization

- Longer document summarization. Example:

- 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.’ (...)

- Summary:

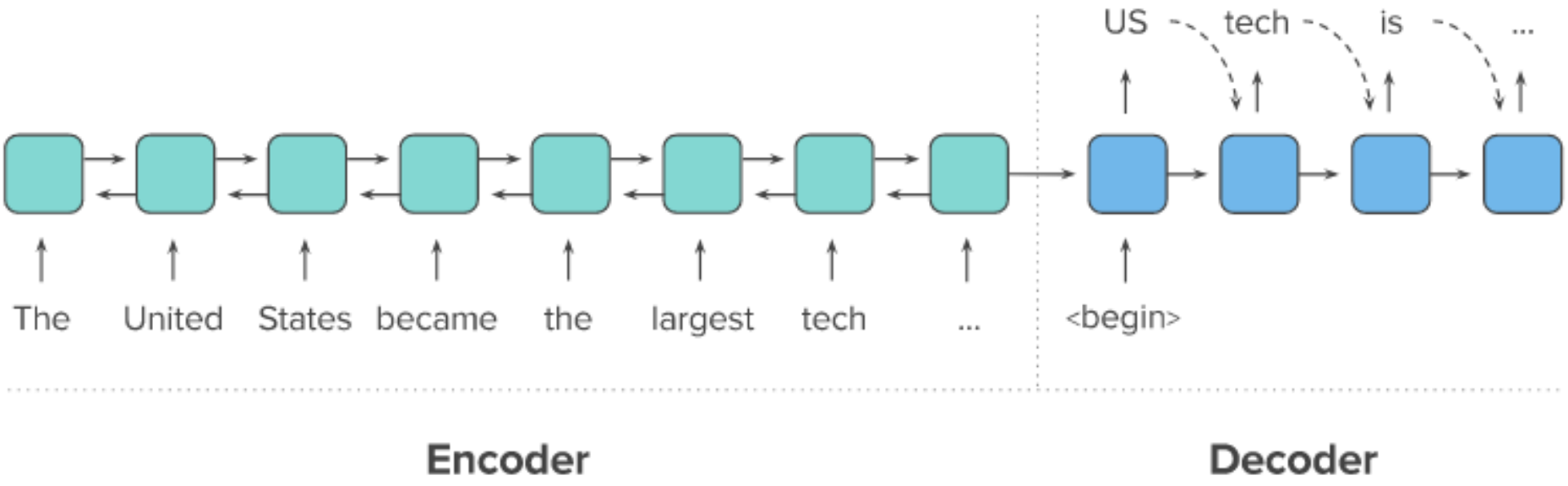
  The former Prime Minister claimed he has 'decades' of work left in him. Joked he would 'turn to drink' if he ever stepped down from global roles. Wants to recruit former government heads to advise current leaders. He was 'mentored' by US president Bill Clinton when he started in 1997.

Attention application: Intra-Decoder attention for Summarization

- Based on paper:
- Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A Deep Reinforced Model for Abstractive Summarization
- But similar ideas appear elsewhere also

- Two necessary, new ingredients
  - Attention during generation → Today
  - Reinforcement learning → Not in this class

Attention application:
Similar Seq2Seq Idea as in Translation



- Problems: For longer outputs (MT was just single sentences), the decoder starts to repeat itself

More advanced attention

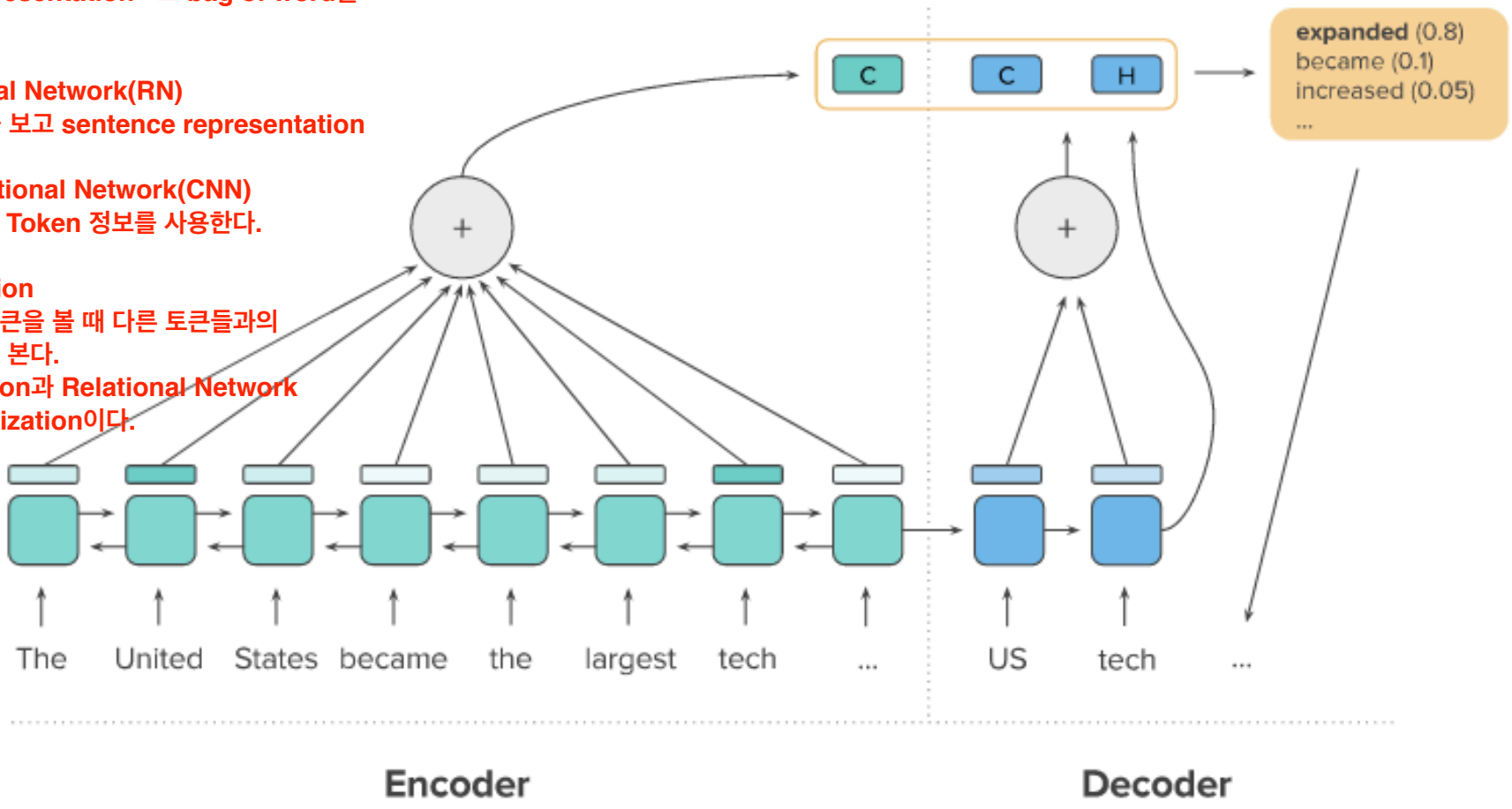
1. More advanced encoder attention
2. Self-attention (= intra-decoder attention)

1. Continuous Bag of Word
- Word representation으로 bag of word를 사용한다.

2. Relational Network(RN)
- 모든 pair을 보고 sentence representation

3. Convolutional Network(CNN)
- 작은 범위의 Token 정보를 사용한다.

Self-Attention
- 각 T번째 토큰을 볼 때 다른 토큰들과의 pair 관계를 본다.
- Convolution과 Relational Network의 generalization이다.



1. Details of this attention mechanism

- More advanced similarity function than simple inner product:

$$e_{ti} = f(h_t^d, h_i^e)$$

$$f(h_t^d, h_i^e) = h_t^{dT} W_{\text{attn}}^e h_i^e$$

- Temporal attention function, penalizing input tokens that have obtained high attention scores in past decoding steps:

$$e'_{ti} = \begin{cases} exp(e_{ti}) & \text{if } t = 1 \\ \frac{exp(e_{ti})}{\sum_{j=1}^{t-1} \exp(e_{ji})} & \text{otherwise} \end{cases}$$

- Improves coverage and prevent repeated attention to same inputs

1. Details of this attention mechanism

- Combine softmax'ed weighted hidden states from encoder:

$$\alpha_{ti}^{e} = \frac{e'_{ti}}{\sum_{j=1}^{n} e'_{tj}}$$

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

- Remember softmax-normalized encoder $\alpha's$ for later!

2. Self-attention on decoder

- Self-attention: A general idea used in many RNN models (e.g. Language Models). The hidden states "attend to themselves", i.e. each hidden state attends to the previous hidden states of the *same* RNN.

- On step *t*, h_t^d attends to *previous* **decoder** hidden states $h_{t'}^d$:

$$e_{tt'}^d = h_t^{dT} W_{\text{attn}}^d h_{t'}^d$$

- Apply softmax to get attention distribution over *previous* hidden states $h_{t'}^d$ for *t' = 1,…,t-1*:

$$\alpha_{tt'}^d = \frac{exp(e_{tt'}^d)}{\sum_{j=1}^{t-1} exp(e_{tj}^d)}$$

- Compute decoder attention output: $c_t^d = \sum_{j=1}^{t-1} \alpha_{tj}^d h_j^d$

2. Combine softmax and pointers using both attention computations

- Compute probability of copying/pointing to word 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 standard softmax:

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

- If pointing, use encoder attention weights (from 2 slides ago)

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

- Combine everything:

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

Summarization Results

cia documents reveal iot-specific televisions can be used to secretly record conversations .
cybercriminals who initiated the attack managed to commandeer a large number of internet-connected devices in current use .
cia documents revealed that microwave ovens can spy on you - maybe if you personally don't suffer the consequences of the sub-par security of the iot .

Internet of Things (IoT) security breaches have been dominating the headlines lately . WikiLeaks's trove of CIA documents revealed that internet-connected televisions can be used to secretly record conversations . Trump's advisor Kellyanne Conway believes that microwave ovens can spy on you - maybe she was referring to microwave cameras which indeed can be used for surveillance . And don't delude yourself that you are immune to IoT attacks , with 96 % of security professionals responding to a new survey expecting an increase in IoT breaches this year . Even if you personally don't suffer the consequences of the sub-par security of the IoT , your connected gadgets may well be unwittingly cooperating with criminals . Last October , Internet service provider Dyn came under an attack that disrupted access to popular websites . The cybercriminals who initiated the attack managed to commandeer a large number of internet-connected devices (mostly DVRs and cameras) to serve as their helpers . As a result , cybersecurity expert Bruce Schneier has called for government regulation of the IoT , concluding that both IoT manufacturers and their customers don't care about the security of the 8.4 billion internet-connected devices in current use . Whether because of government regulation or good old-fashioned self-interest , we can expect increased investment in IoT security technologies . In its recently-released TechRadar report for security and risk professionals , Forrester Research discusses the outlook for the 13 most relevant and important IoT security technologies , warning that " there is no single , magic security bullet that can easily fix all IoT security issues . " Based on Forrester's analysis , here's my list of the 6 hottest technologies for IoT security : IoT network security : Protecting and securing the network connecting IoT devices to back-end systems on the internet . IoT network security is a bit more challenging than traditional network security because there is a wider range of communication protocols , standards , and device capabilities , all of which pose significant issues and increased complexity . Key capabilities include traditional endpoint security features such as antivirus and antimalware as well as other features such as firewalls and intrusion prevention and detection systems . Sample vendors : Bayshore Networks , Cisco , Darktrace , and Senrio . IoT authentication : Providing the ability for users to authenticate an IoT device , including managing multiple users of a single device (such as a connected car) , ranging from simple static password/pins to more robust authentication mechanisms such as two-factor

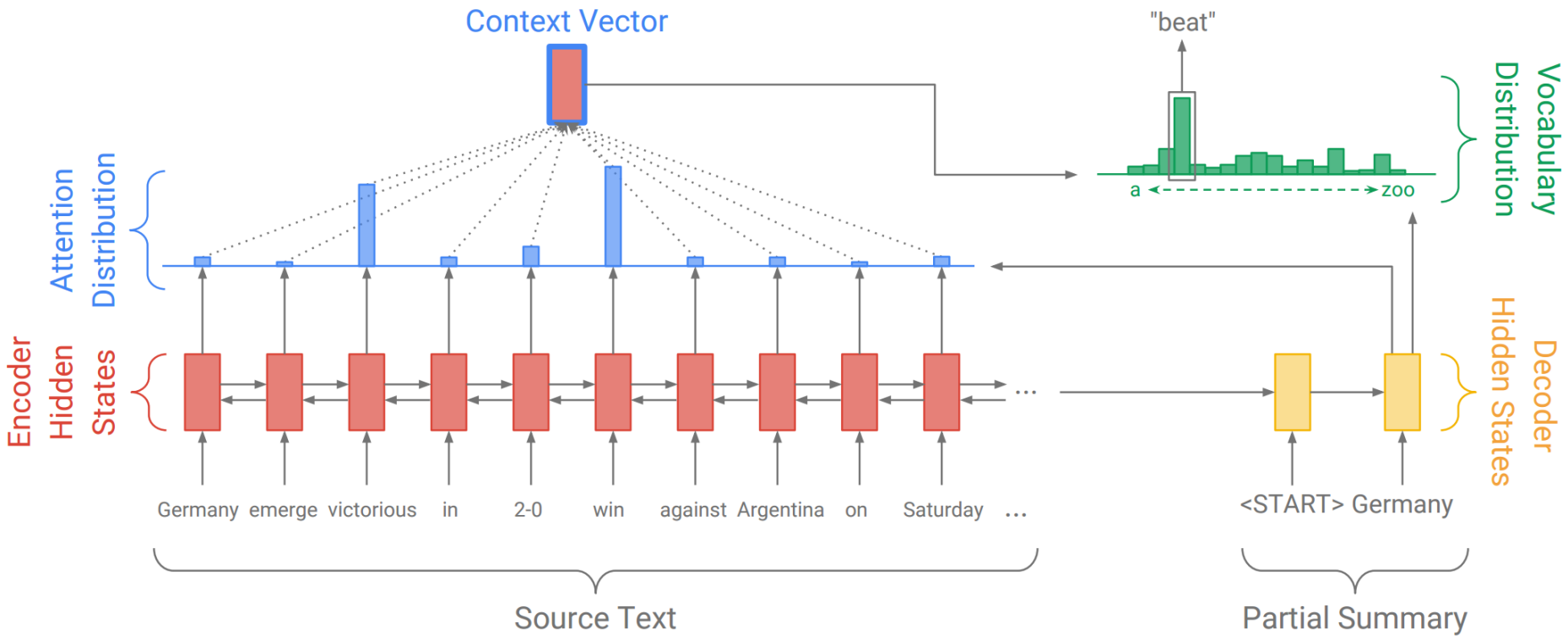
Summarization Results

The bottleneck is no longer access to information; now it's our ability to keep up.
AI can be trained on a variety of different types of texts and summary lengths.
A model that can generate long, coherent, and meaningful summaries remains an open research problem.

The last few decades have witnessed a fundamental change in the challenge of taking in new information. The bottleneck is no longer access to information; now it's our ability to keep up. We all have to read more and more to keep up-to-date with our jobs, the news, and social media. We've looked at how AI can improve people's work by helping with this information deluge and one potential answer is to have algorithms automatically summarize longer texts. Training a model that can generate long, coherent, and meaningful summaries remains an open research problem. In fact, generating any kind of longer text is hard for even the most advanced deep learning algorithms. In order to make summarization successful, we introduce two separate improvements: a more contextual word generation model and a new way of training summarization models via reinforcement learning (RL). The combination of the two training methods enables the system to create relevant and highly readable multi-sentence summaries of long text, such as news articles, significantly improving on previous results. Our algorithm can be trained on a variety of different types of texts and summary lengths. In this blog post, we present the main contributions of our model and an overview of the natural language challenges specific to text summarization.

Similar ideas explored simultaneously by Abi et al.

- *Get To The Point: Summarization with Pointer-Generator Networks*, Abigail See, Peter J. Liu, Christopher Manning, 2017



Blog post: http://www.abigailsee.com/2017/04/16/taming-rnns-for-better-summarization.html

Preview

- Hopefully, you can see how useful and versatile attention is
- Next lecture we will go even further and cover a model that *only* has attention (The Transformer)
- But, for now, we will cover some tips and tricks to actually **scale up** machine translation.

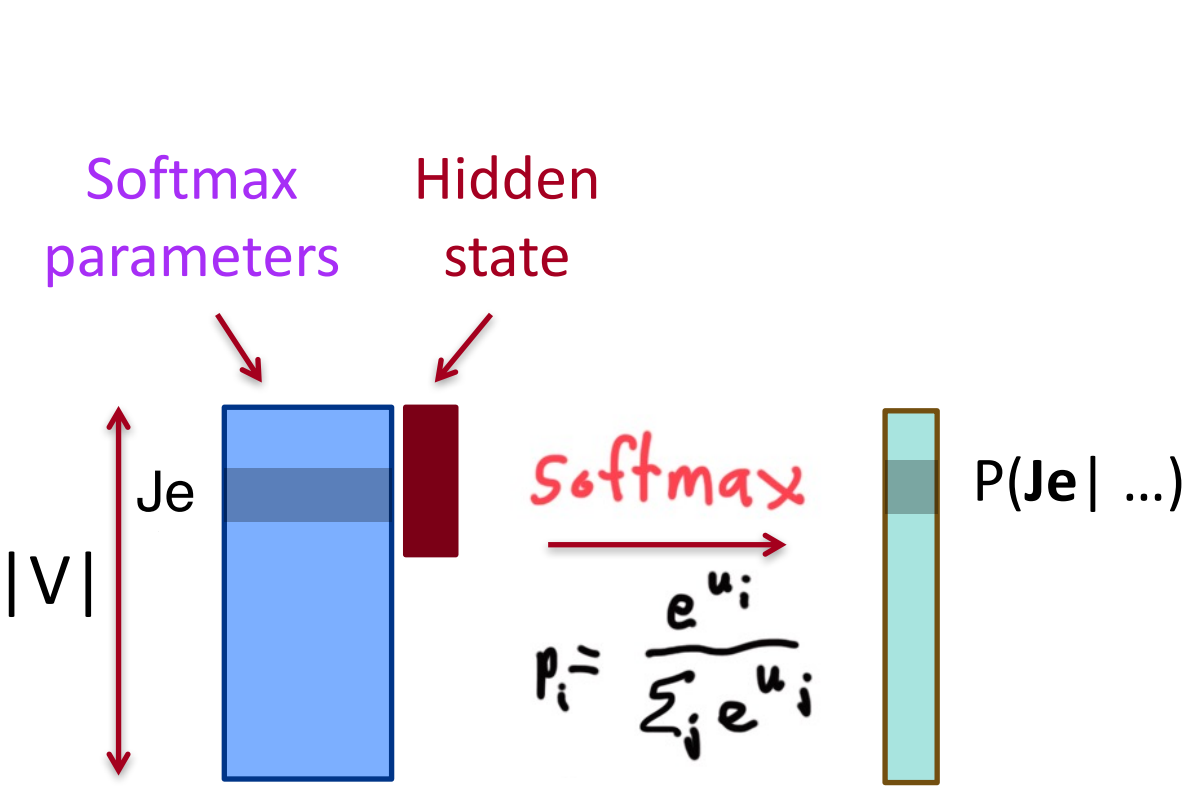
Extending NMT to more languages

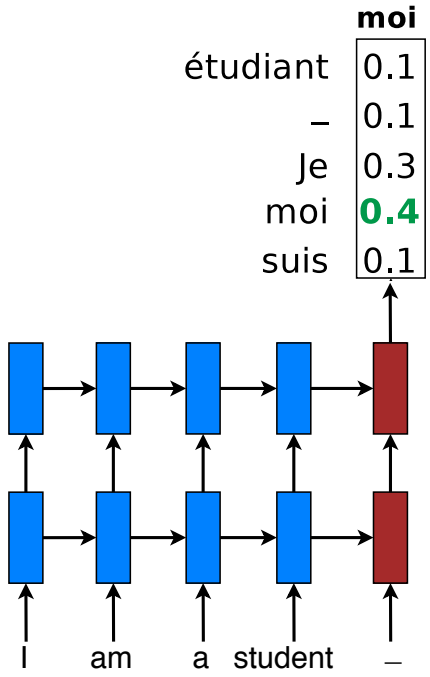
- "Copy" mechanisms are **not sufficient**.
  - Transliteration: Christopher ↦ Kryštof
  - Multi-word alignment: Solar system ↦ Sonnensystem

- Need to handle **large, open vocabulary**
  - Rich morphology:
    - nejneobhospodařovávatelnějšímu - Czech = "to the worst farmable one"
    - Donaudampfschiffahrtsgesellschaftskapitän – German = Danube steamship company captain
  - Informal spelling: goooooood morning !!!!!

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

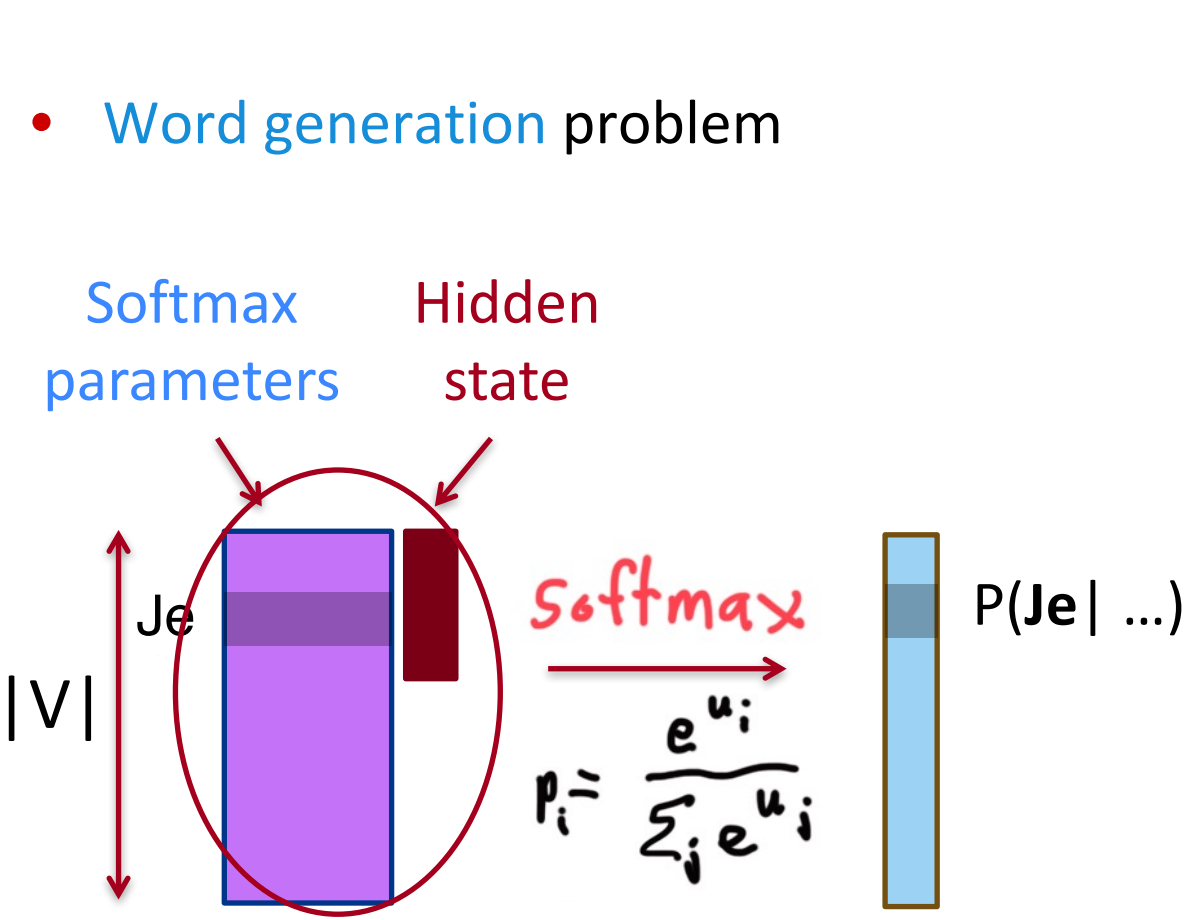
Dealing with a large output vocabulary in MT++

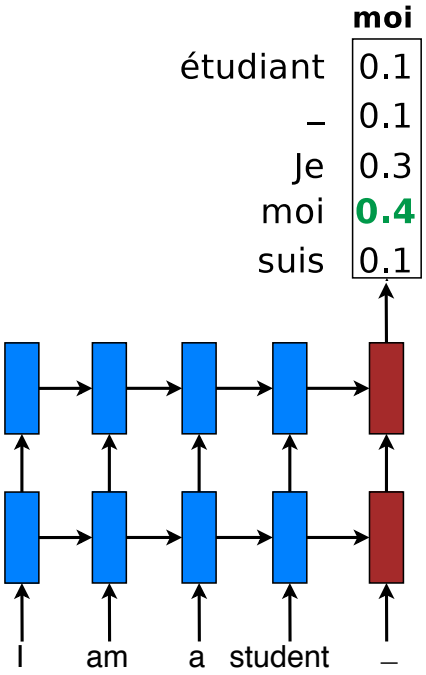




The word generation problem

- Word generation problem

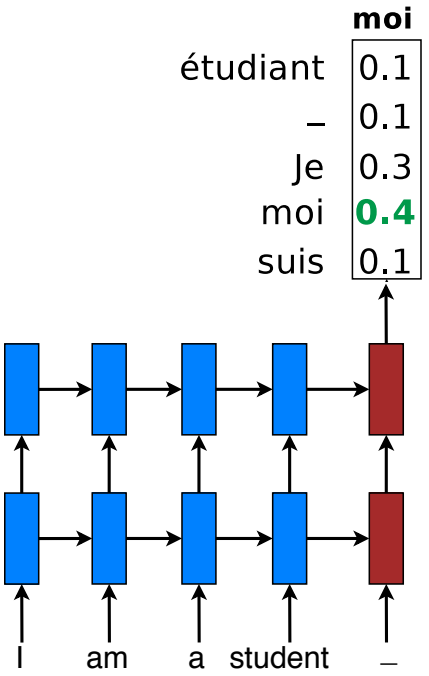




Softmax computation is expensive.

The word generation problem

- Word generation problem
  - If vocabs are modest, e.g., 50K



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

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

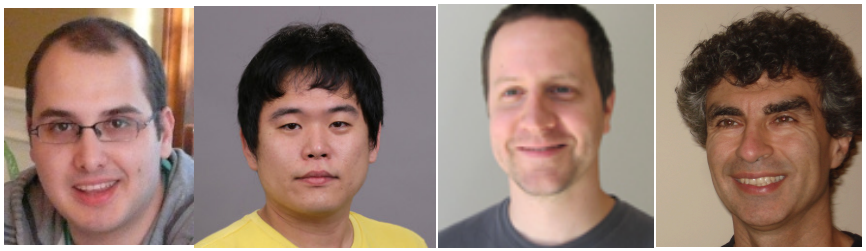
First thought: scale the softmax

- Lots of ideas from the neural LM literature!
- *Hierarchical models*: tree-structured vocabulary
  - [Morin & Bengio, AISTATS'05], [Mnih & Hinton, NIPS'09].
  - Complex, sensitive to tree structures.
- *Noise-contrastive estimation*: binary classification
  - [Mnih & Teh, ICML'12], [Vaswani et al., EMNLP'13].
  - Different noise samples per training example.*

Not GPU-friendly

*We'll mention a simple fix for this!

Large-vocab NMT



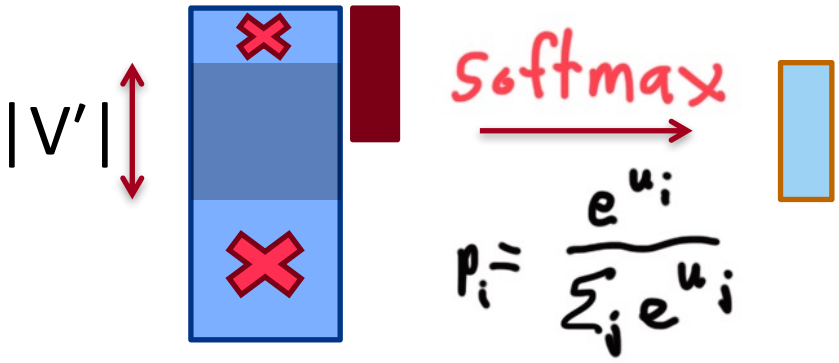
- GPU-friendly.
- *Training*: a subset of the vocabulary at a time.
- *Testing*: smart on the set of possible translations.

Fast at both train & test time.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, Yoshua Bengio. ***On Using Very Large Target Vocabulary for Neural Machine Translation****. ACL'15.*

Training

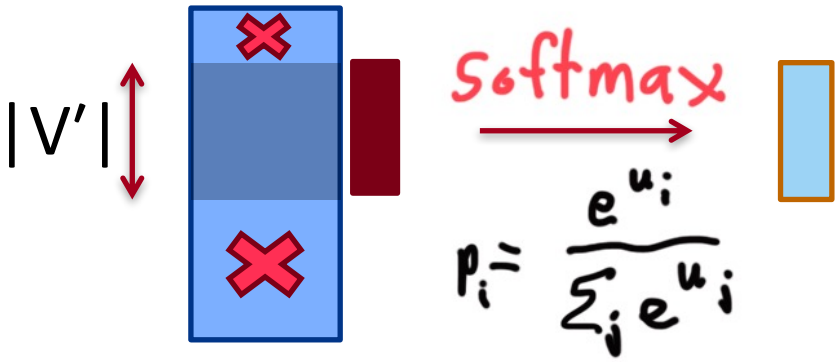
- Each time train on a smaller vocab V′ ≪ V



How do we select V′?

Training

- Each time train on a smaller vocab $V' \ll V$



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

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, cats, he, likes}

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}

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}

- *Practice*: $|V| = 500K$, $|V'| = 30K$ or $50K$.

Testing – *Select candidate words*

- **K** most frequent words: unigram prob.

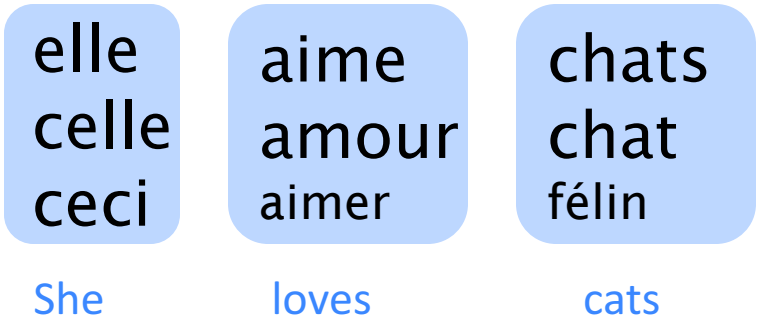
de,
,
la
.
et
des
les
…

Testing – *Select candidate words*

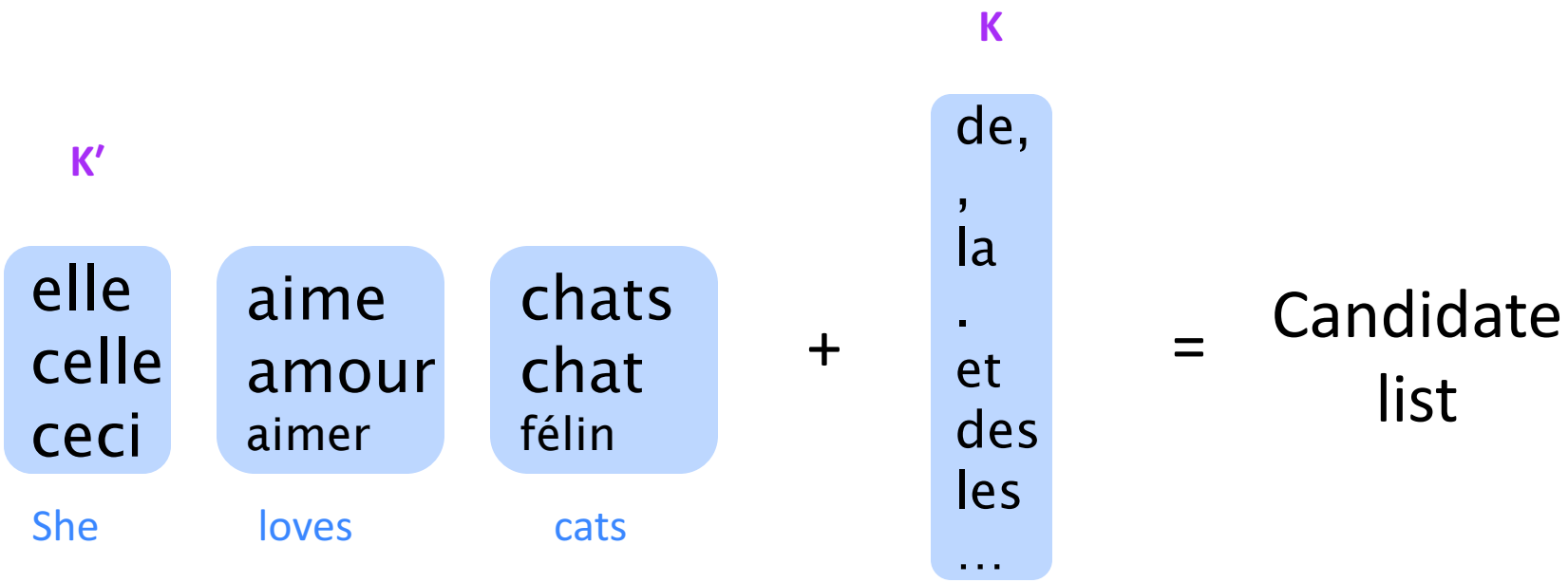
- **K** most frequent words: unigram prob.

- Candidate target words
  - **K′** choices per source word. K′ = 3.

de,
,
la
.
et
des
les
…



Testing – *Select candidate words*



- Produce translations within the candidate list
- *Practice*: K′ = 10 or 20, K = 15k, 30k, or 50k.

More on large-vocab techniques

- "BlackOut: Speeding up Recurrent Neural Network Language Models with very Large Vocabularies" – [Ji, Vishwanathan, Satish, Anderson, Dubey, ICLR'16].
  - Good survey over many techniques.

- "Simple, Fast Noise Contrastive Estimation for Large RNN Vocabularies" – [Zoph, Vaswani, May, Knight, NAACL'16].
  - Use the same samples per minibatch. GPU efficient.

Sub-word NMT: two trends

- Same seq2seq architecture:
  - Use smaller units.
  - [Sennrich, Haddow, Birch, ACL'16a], [Chung, Cho, Bengio, ACL'16].

- Hybrid architectures:
  - RNN for *words* + something else for *characters*.
  - [Costa-Jussà & Fonollosa, ACL'16], [Luong & Manning, ACL'16].

Byte Pair Encoding

- A compression algorithm:
  - Most frequent byte pair ↦ a new byte.

Replace bytes with character ngrams

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

Byte Pair Encoding

- A word segmentation algorithm:
  - Start with a vocabulary of characters.
  - Most frequent ngram pairs ↦ a new ngram.

Byte Pair Encoding

- A word segmentation algorithm:
  - Start with a vocabulary of characters.
  - Most frequent ngram pairs ↦ a new ngram.

Dictionary

5 l o w
2 l o w e r
6 n e w e s t
3 w i d e s t

Vocabulary

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

Start with all characters in vocab

(Example from Sennrich)

Byte Pair Encoding

- A word segmentation algorithm:
  - Start with a vocabulary of characters.
  - Most frequent ngram pairs ↦ a new ngram.

Dictionary

5 l o w
2 l o w e r
6 n e w **es** t
3 w i d **es** t

Vocabulary

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

Add a pair (e, s) with freq 9

(Example from Sennrich)

Byte Pair Encoding

- A word segmentation algorithm:
  - Start with a vocabulary of characters.
  - Most frequent ngram pairs ↦ a new ngram.

Dictionary

5 l o w
2 l o w e r
6 n e w **est**
3 w i d **est**

Vocabulary

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

Add a pair (es, t) with freq 9

(Example from Sennrich)

Byte Pair Encoding

- A word segmentation algorithm:
  - Start with a vocabulary of characters.
  - Most frequent ngram pairs ↦ a new ngram.

Dictionary

5 **lo** w
2 **lo** w e r
6 n e w est
3 w i d est

Vocabulary

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

Add a pair (l, o) with freq 7

(Example from Sennrich)

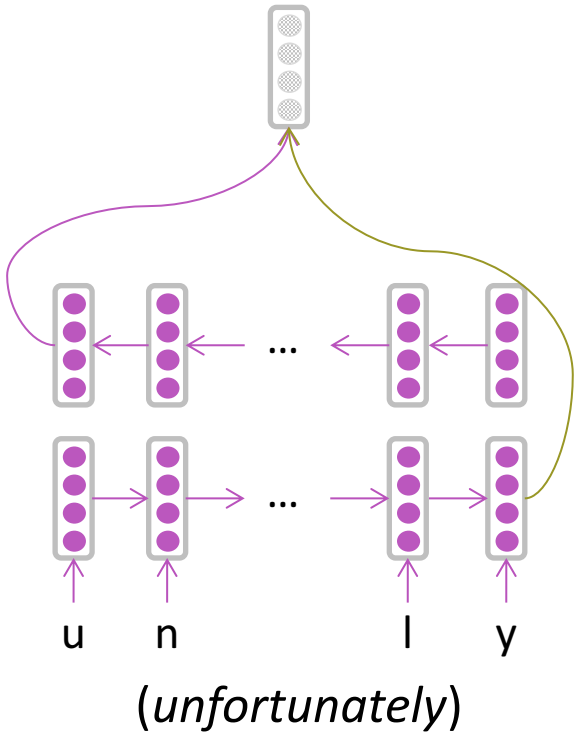
Byte Pair Encoding

- A word segmentation algorithm:
  - Start with a vocabulary of characters.
  - Most frequent ngram pairs ↦ a new ngram.
- Automatically decide vocabs for NMT

Top places in WMT 2016!

https://github.com/rsennrich/nematus

Character-based LSTM

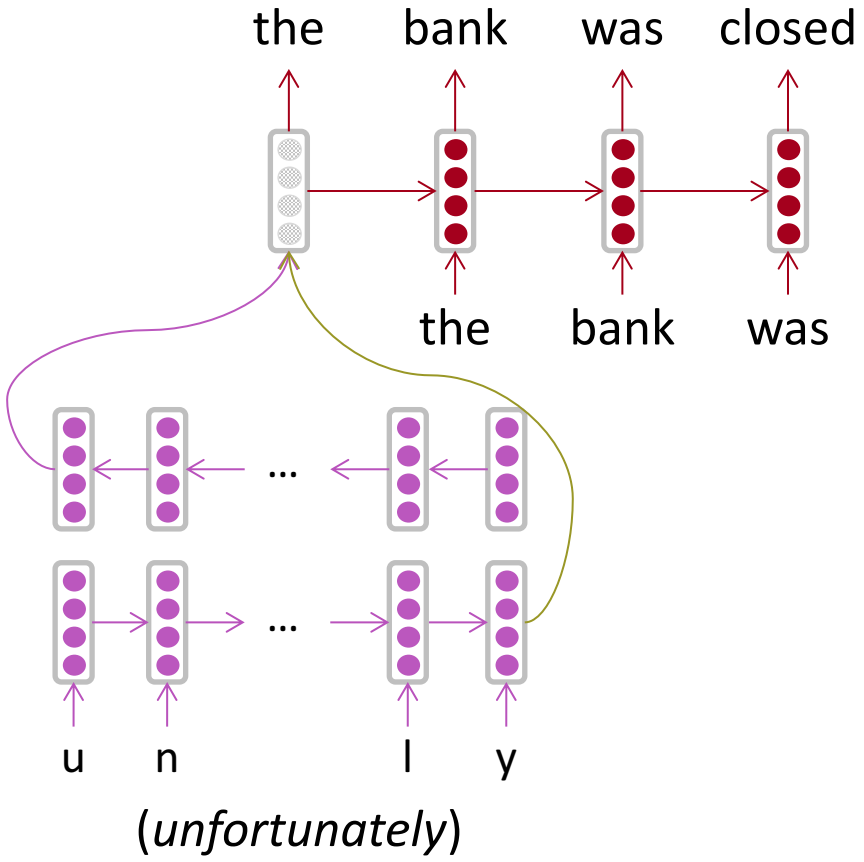


(*unfortunately*)

Bi-LSTM builds word representations

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

Character-based LSTM



Recurrent Language Model

Bi-LSTM builds word representations

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

Hybrid NMT

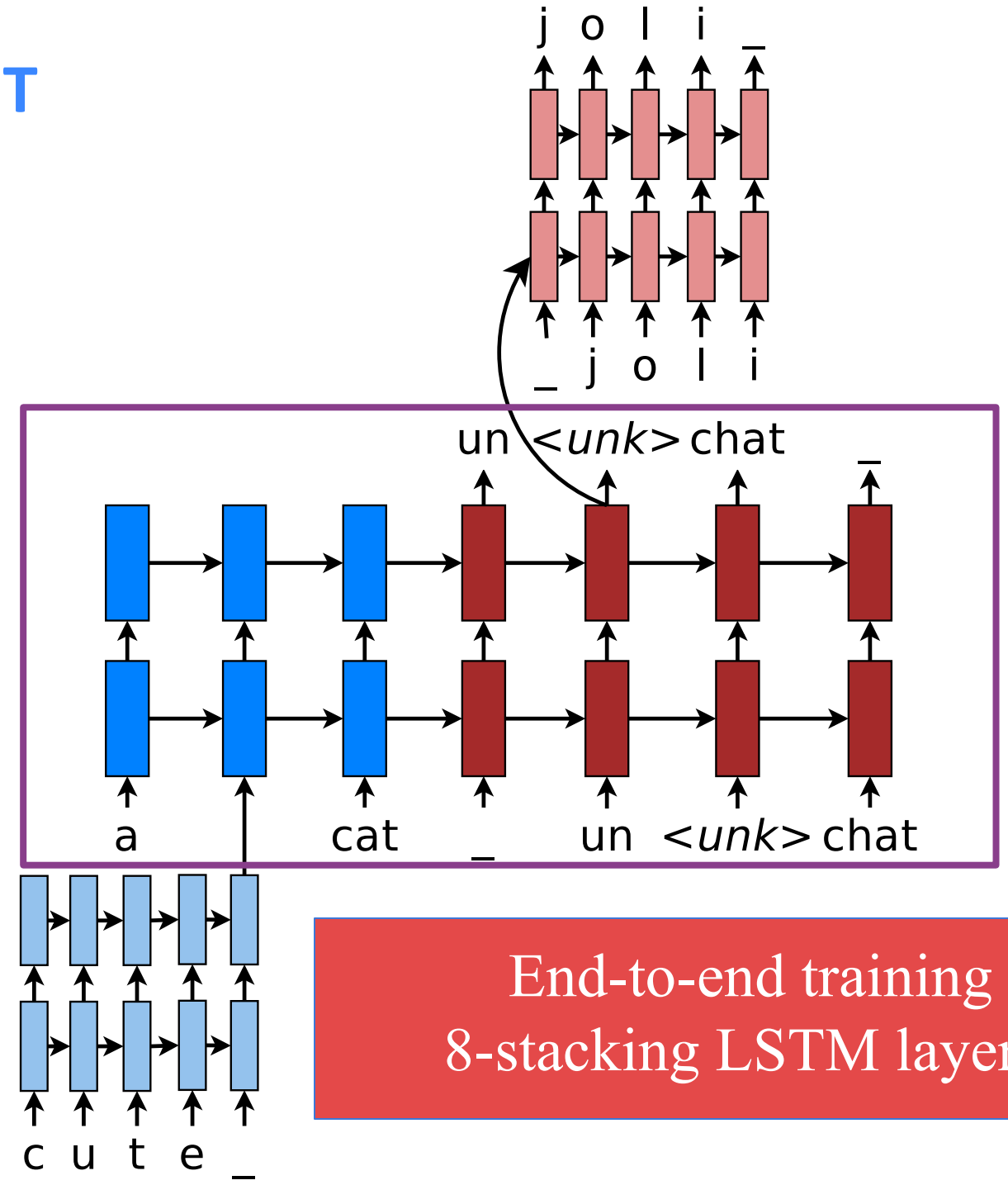

- A *best-of-both-worlds* architecture:
  - Translate mostly at the word level
  - Only go to the character level when needed.

- More than 2 BLEU improvement over a copy mechanism.

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

Hybrid NMT

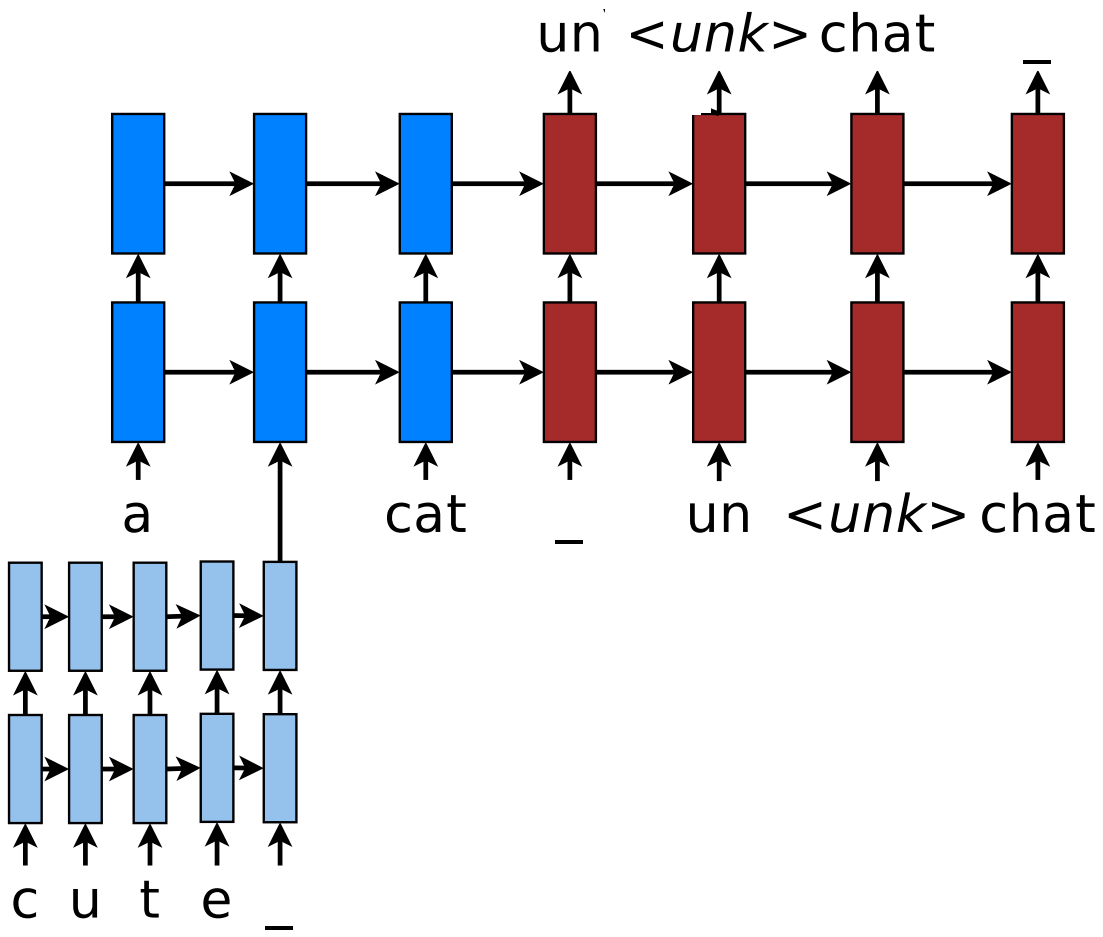
Word-level
(4 layers)



End-to-end training
8-stacking LSTM layers.

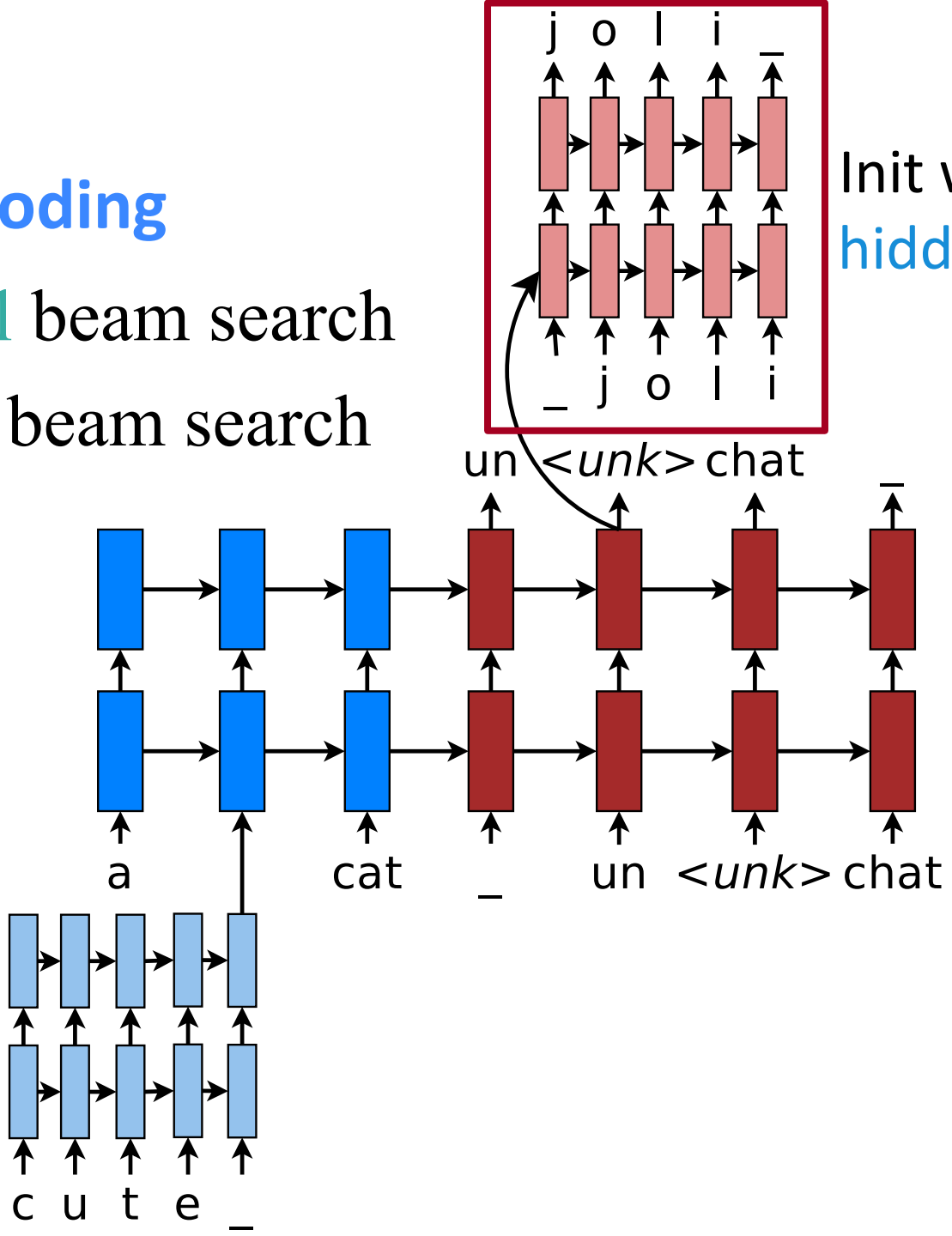
2-stage Decoding

- Word-level beam search



2-stage Decoding

- Word-level beam search
- Char-level beam search for <unk>.



English-Czech Results

- Train on WMT'15 data (12M sentence pairs)
  - newstest2015

Systems	BLEU
Winning WMT'15 (Bojar & Tamchyna, 2015)	18.8
Word-level NMT (Jean et al., 2015)	18.3

30x data
3 systems

Large vocab
+ copy mechanism

English-Czech Results

- Train on WMT'15 data (12M sentence pairs)
  - newstest2015

Systems	BLEU
Winning WMT'15 (Bojar & Tamchyna, 2015)	18.8
Word-level NMT (Jean et al., 2015)	18.3
Hybrid NMT (Luong & Manning, 2016)*	**20.7**

30x data
3 systems

Large vocab
+ copy mechanism

Sample English-Czech translations

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 **zvláštní**
word	Její <unk> dcera <unk> <unk> řekla , že je to trochu divné
	Její **11-year-old** dcera **Shani ,** řekla , že je to trochu ***divné***
hybrid	Její <unk> dcera , <unk> <unk> , řekla , že je to <unk> <unk>
	Její **jedenáctiletá** dcera , **Graham** ***Bart*** , řekla , že cítí trochu ***divný***

- Word-based: identity copy fails.

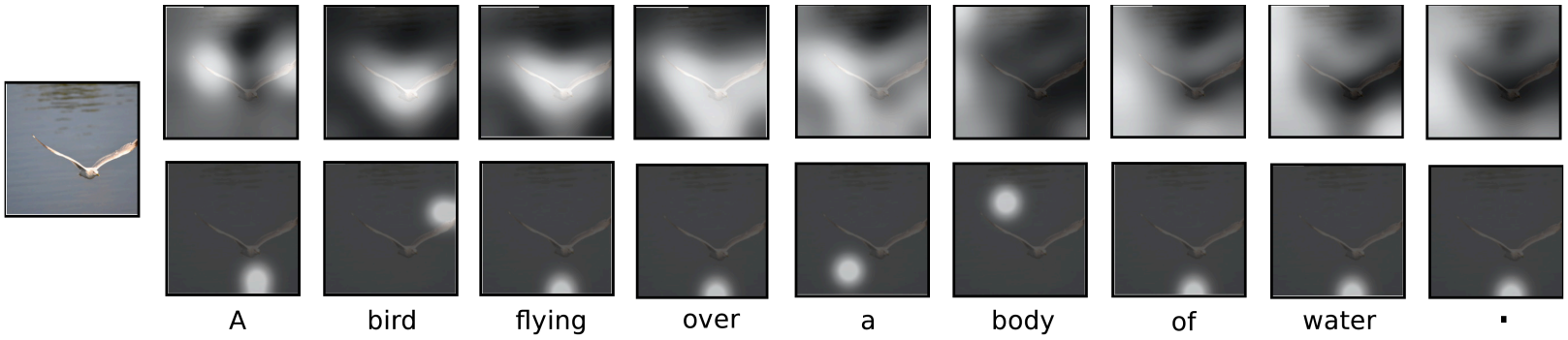
Sample English-Czech translations

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 **zvláštní**
word	Její <unk> dcera <unk> <unk> řekla , že je to trochu divné
	Její **11-year-old** dcera **Shani ,** řekla , že je to trochu ***divné***
hybrid	Její <unk> dcera , <unk> <unk> , řekla , že je to <unk> <unk>
	Její **jedenáctiletá** dcera , **Graham** ***Bart*** , řekla , že cítí trochu ***divný***

- Hybrid: correct, ***11-year-old*** – **jedenáctiletá**.

Using attention for coverage

- Caption generation



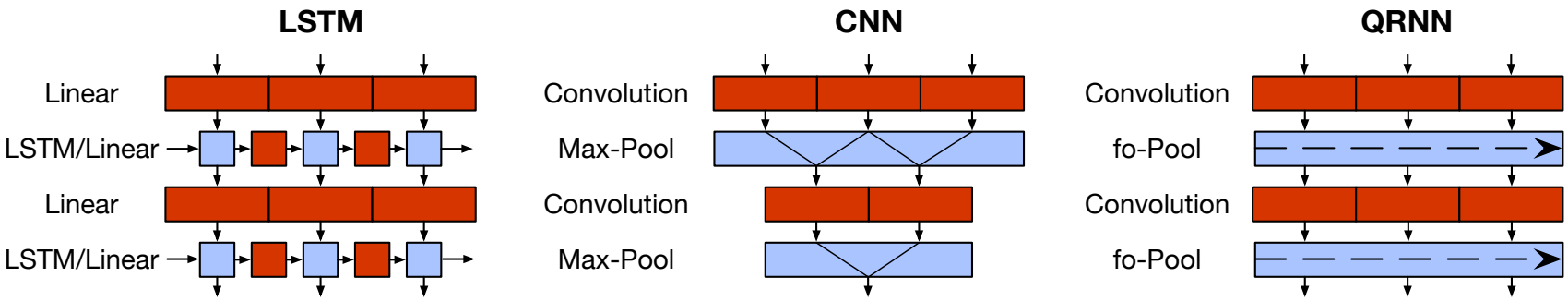
How to not miss an important image patch?

Xu, Ba, Kiros, Cho, Courville, Salakhutdinov, Zemel, Bengio. ***Show, Attend and Tell: Neural Image Caption Generation with Visual Attention****. ICML'15*

RNNs are Slow

- RNNs are the basic building block for deepNLP
- Idea: Take the best and parallelizable parts of RNNs and CNNs
- Quasi-Recurrent Neural Networks by
  James Bradbury, Stephen Merity, Caiming Xiong & Richard Socher

Quasi-Recurrent Neural Network



- Parallelism computation across time:

$$\mathbf{z}_t = \tanh(\mathbf{W}_z^1 \mathbf{x}_{t-1} + \mathbf{W}_z^2 \mathbf{x}_t)$$
$$\mathbf{f}_t = \sigma(\mathbf{W}_f^1 \mathbf{x}_{t-1} + \mathbf{W}_f^2 \mathbf{x}_t)$$
$$\mathbf{o}_t = \sigma(\mathbf{W}_o^1 \mathbf{x}_{t-1} + \mathbf{W}_o^2 \mathbf{x}_t).$$

$$\mathbf{Z} = \tanh(\mathbf{W}_z * \mathbf{X})$$
$$\mathbf{F} = \sigma(\mathbf{W}_f * \mathbf{X})$$
$$\mathbf{O} = \sigma(\mathbf{W}_o * \mathbf{X}),$$

- Element-wise gated recurrence for parallelism across channels:

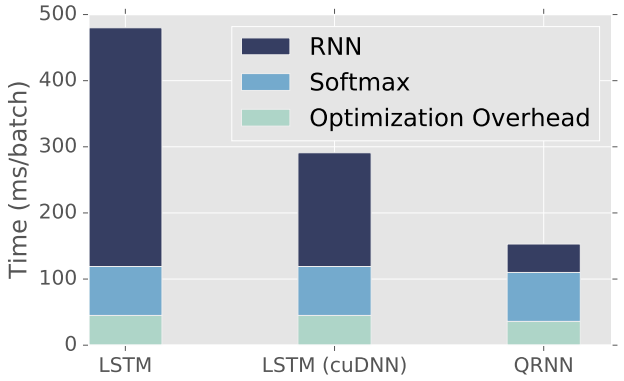
$$\mathbf{h}_t = \mathbf{f}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{f}_t) \odot \mathbf{z}_t,$$

Q-RNNs for Language Modeling

- Better

Model	Parameters	Validation	Test
LSTM (medium) (Zaremba et al., 2014)	20M	86.2	82.7
Variational LSTM (medium) (Gal & Ghahramani, 2016)	20M	81.9	79.7
LSTM with CharCNN embeddings (Kim et al., 2016)	19M	–	78.9
Zoneout + Variational LSTM (medium) (Merity et al., 2016)	20M	84.4	80.6
Our models			
LSTM (medium)	20M	85.7	82.0
QRNN (medium)	18M	82.9	79.9
QRNN + zoneout ($p = 0.1$) (medium)	18M	82.1	78.3

- Faster



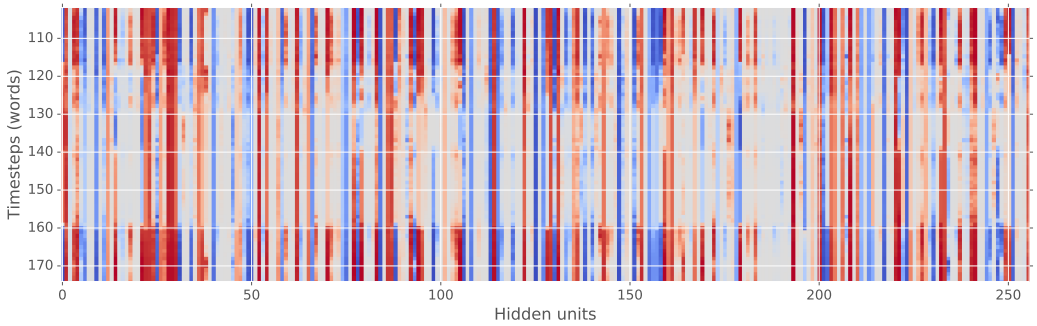
		Sequence length				
		32	64	128	256	512
Batch size	8	**5.5x**	**8.8x**	**11.0x**	**12.4x**	**16.9x**
	16	**5.5x**	**6.7x**	**7.8x**	**8.3x**	**10.8x**
	32	**4.2x**	**4.5x**	**4.9x**	**4.9x**	**6.4x**
	64	**3.0x**	**3.0x**	**3.0x**	**3.0x**	**3.7x**
	128	**2.1x**	**1.9x**	**2.0x**	**2.0x**	**2.4x**
	256	**1.4x**	**1.4x**	**1.3x**	**1.3x**	**1.3x**

Q-RNNs for Sentiment Analysis

- Often better and faster than LSTMs

Model	Time / Epoch (s)	Test Acc (%)
BSVM-bi (Wang & Manning, 2012)	–	91.2
2 layer sequential BoW CNN (Johnson & Zhang, 2014)	–	92.3
Ensemble of RNNs and NB-SVM (Mesnil et al., 2014)	–	92.6
2-layer LSTM (Longpre et al., 2016)	–	87.6
Residual 2-layer bi-LSTM (Longpre et al., 2016)	–	90.1
Our models		
Deeply connected 4-layer LSTM (cuDNN optimized)	480	90.9
Deeply connected 4-layer QRNN	150	91.4
D.C. 4-layer QRNN with $k = 4$	160	91.1

- More interpretable



- Example:
- Initial positive review
- *Review starts out positive*
  At 117: "*not exactly a bad story*"
  At 158: "*I recommend this movie to everyone, even if you've never played the game*"