Attention

M. Soleymani
Sharif University of Technology
Fall 2017

Most slides have been adopted from Fei Fei Li and colleagues lectures, cs231n, Stanford 2016 and some from John Canny, cs294-129, Berkeley, 2016.

Attention

Focusing on a subset of the given information.

2014: Neural Translation Breakthroughs

- Devlin et al, ACL’2014
- Cho et al EMNLP’2014
- Bahdanau, Cho & Bengio, arXiv sept. 2014
- Jean, Cho, Memisevic & Bengio, arXiv dec. 2014
- Sutskever et al NIPS’2014

Other Applications

- Ba et al 2014, **Visual attention for recognition**
- Mnih et al 2014, **Visual attention for recognition**
- Chorowski et al, 2014, **Speech recognition**
- Graves et al 2014, **Neural Turing machines**
- Yao et al 2015, **Video description generation**
- Vinyals et al, 2015, **Conversational Agents**
- Xu et al 2015, **Image caption generation**
- Xu et al 2015, **Visual Question Answering**

Soft vs Hard Attention Models

Hard attention:

- Attend to a single input location among the set of locations.
- Can’t use gradient descent.
- Need **reinforcement learning.**

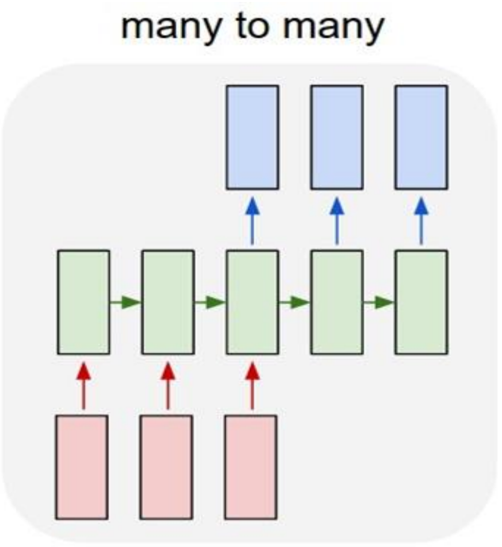
Soft attention:

- Compute a weighted combination (attention) over some inputs using an attention network.
- Can use backpropagation to train end-to-end.

Attention instead of simple encoder-decoder

- Encoder-decoder models
  - needs to be able to compress all the necessary information of a source sentence into a fixed-length vector
  - performance deteriorates rapidly as the length of an input sentence increases.

- Attention avoids this by:
  - allowing the RNN generating the output to focus on hidden states (generated by the first RNN) as they become relevant.

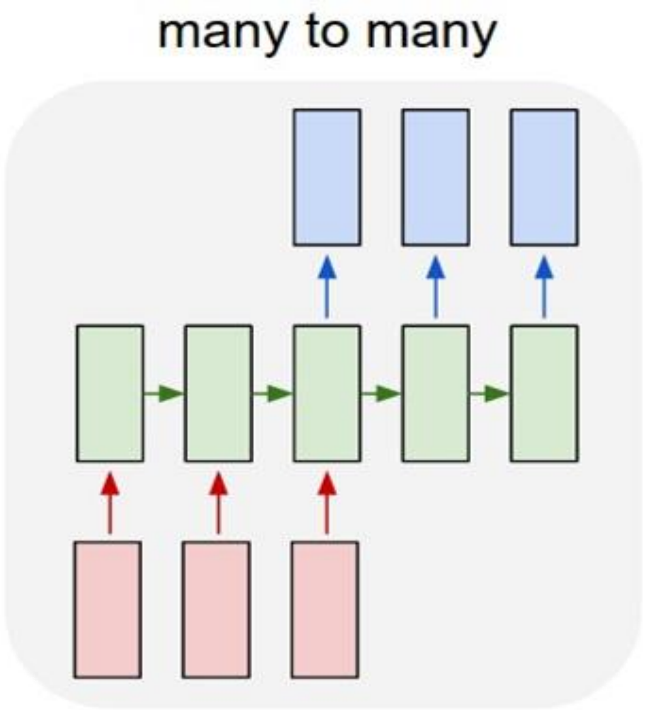


Bahdanau et al, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR 2015

Soft Attention for Translation

An RNN can attend over the output of another RNN. At every time step, it focuses on different positions in the other RNN.

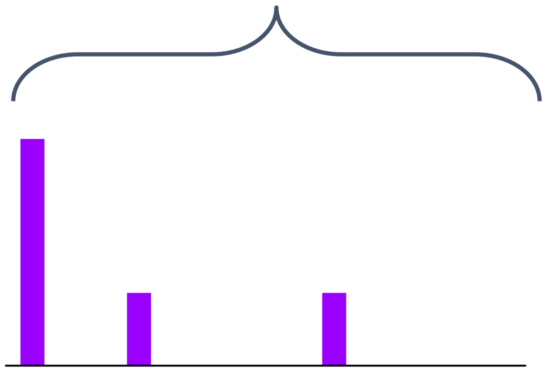
"I love coffee" -> "Me gusta el café"



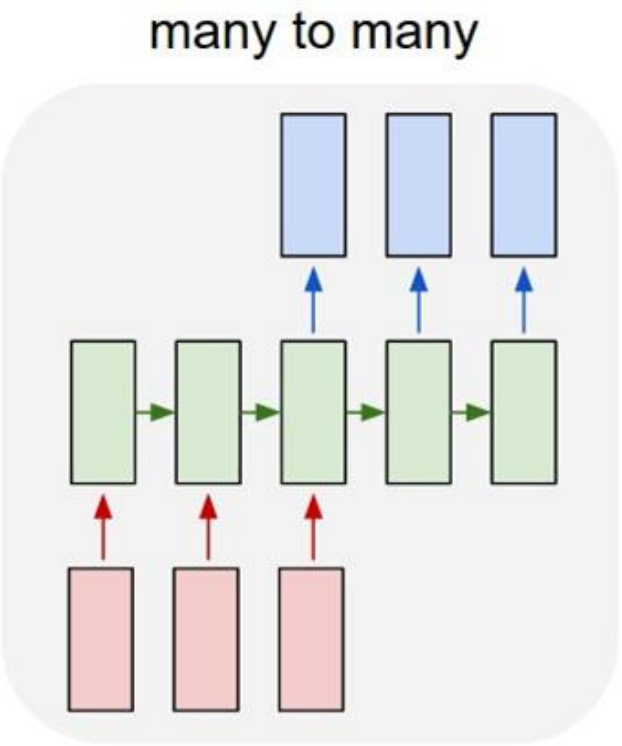
Bahdanau et al, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR 2015

Soft Attention for Translation

Distribution over
input words



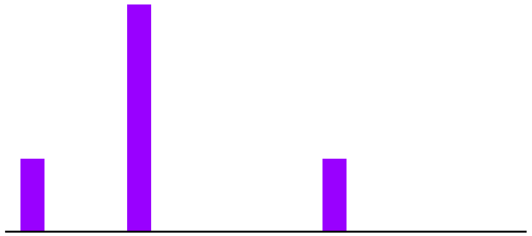
“I love coffee” -> “Me gusta el café”



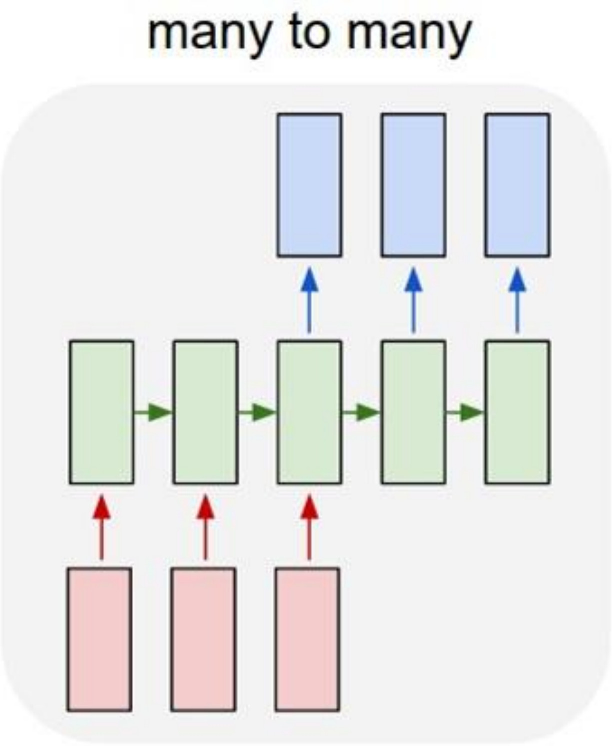
Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation

Distribution over input words



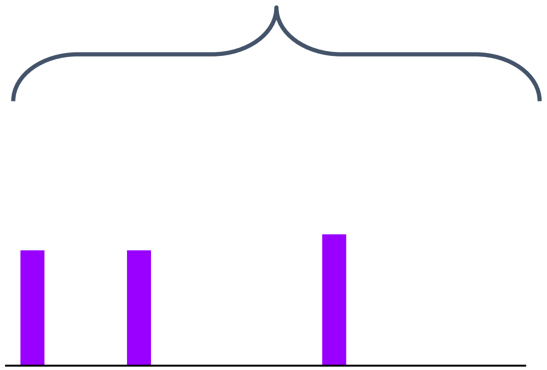
“I love coffee” -> “Me gusta el café”



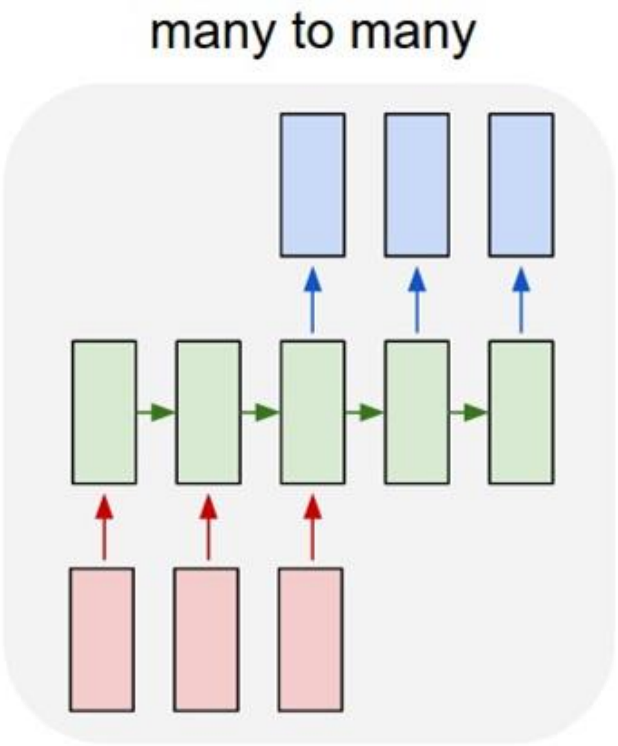
Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation

Distribution over input words



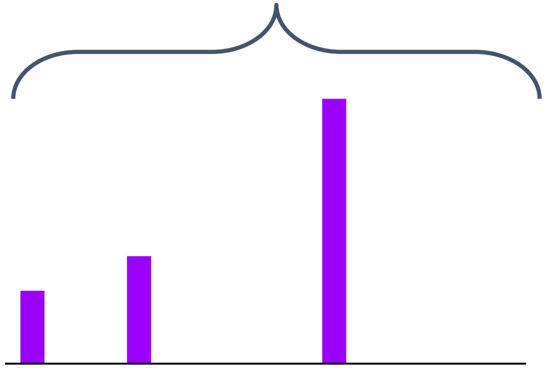
“I love coffee” -> “Me gusta el café”



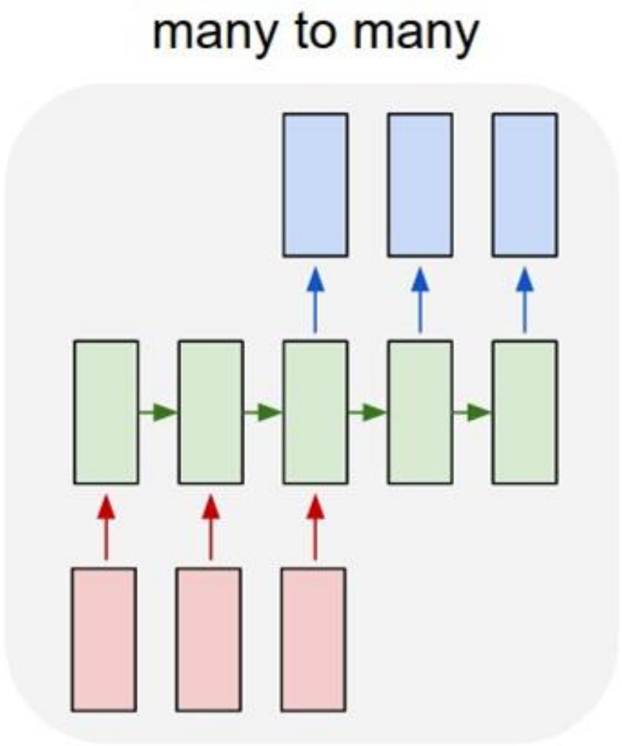
Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation

Distribution over input words



"I love coffee" -> "Me gusta el café"



Bahdanau et al, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR 2015

Soft Attention for Translation

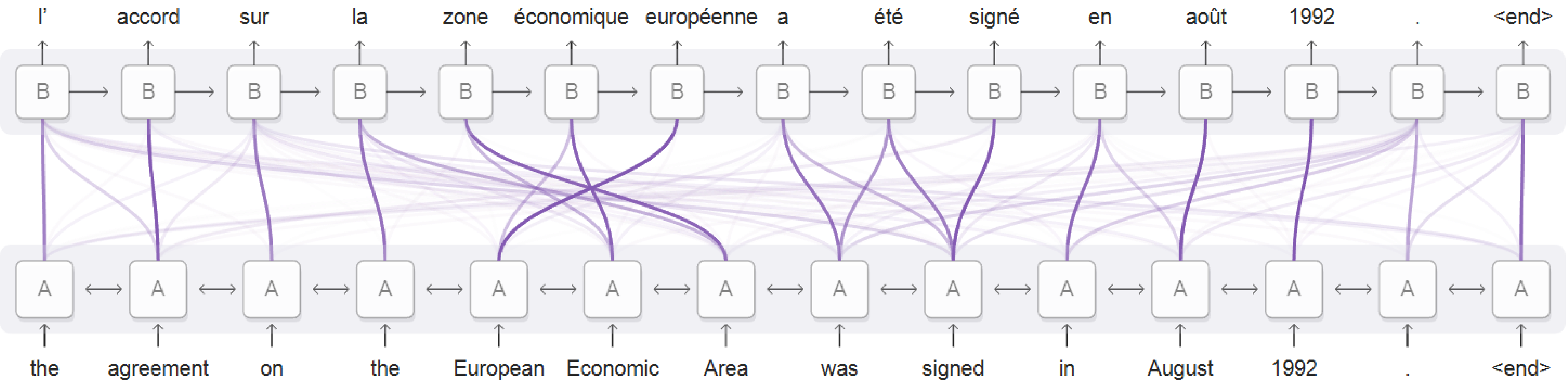


Diagram derived from Fig. 3 of Bahdanau, *et al.* 2014

Source: https://distill.pub/2016/augmented-rnns/

Soft Attention for voice recognition

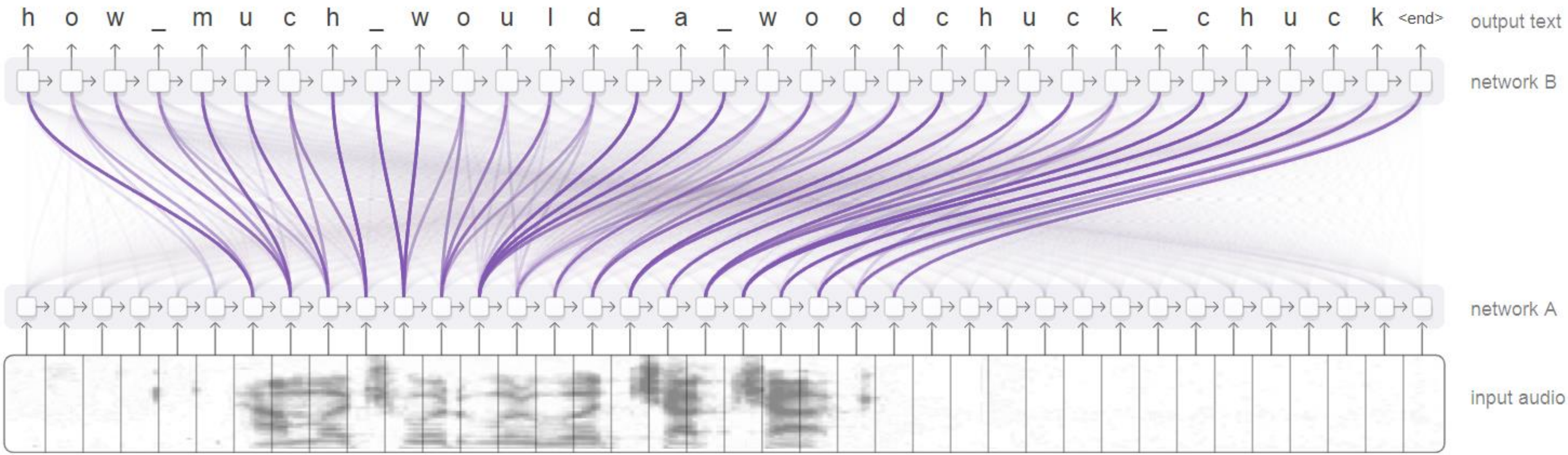
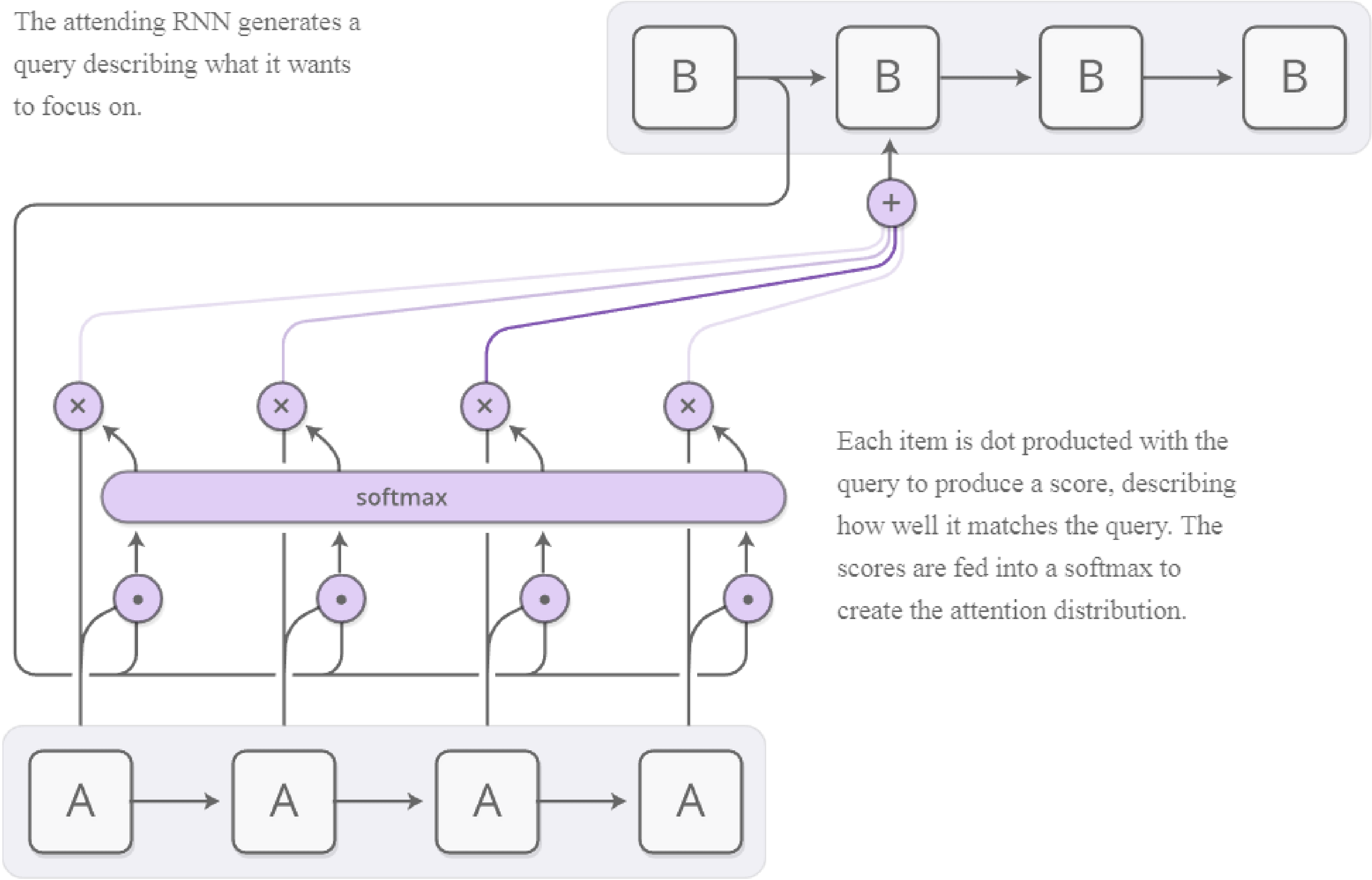


Figure derived from Chan, *et al.* 2015

Source: https://distill.pub/2016/augmented-rnns/

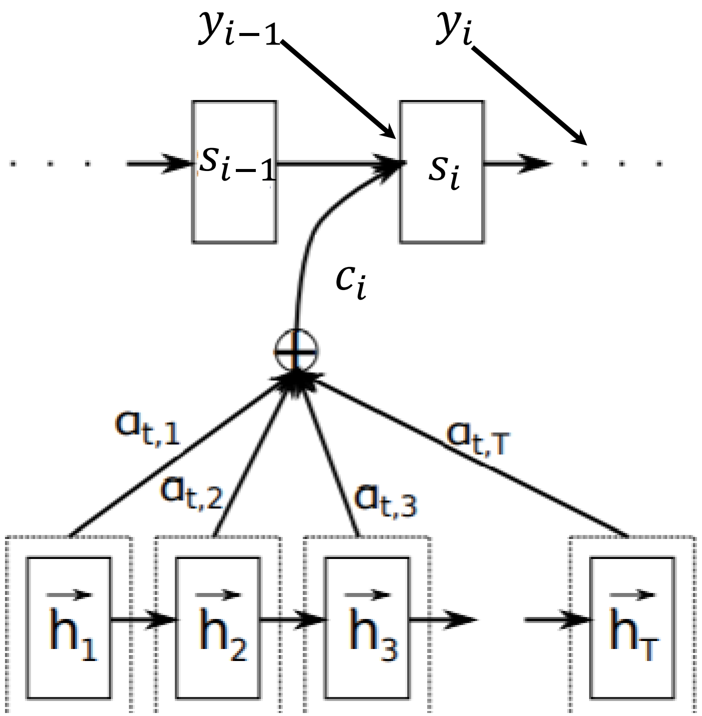
Simple soft attention mechanism



Source: https://distill.pub/2016/augmented-rnns/

Soft Attention for Translation

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$



Context vector (input to decoder):

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

Mixture weights:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

Alignment score (how well do input words near j match output words at position i):

$$e_{ij} = a(s_{i-1}, h_j)$$

alignment model: s a feedforward neural network which is
jointly trained with all the other components of the proposed system

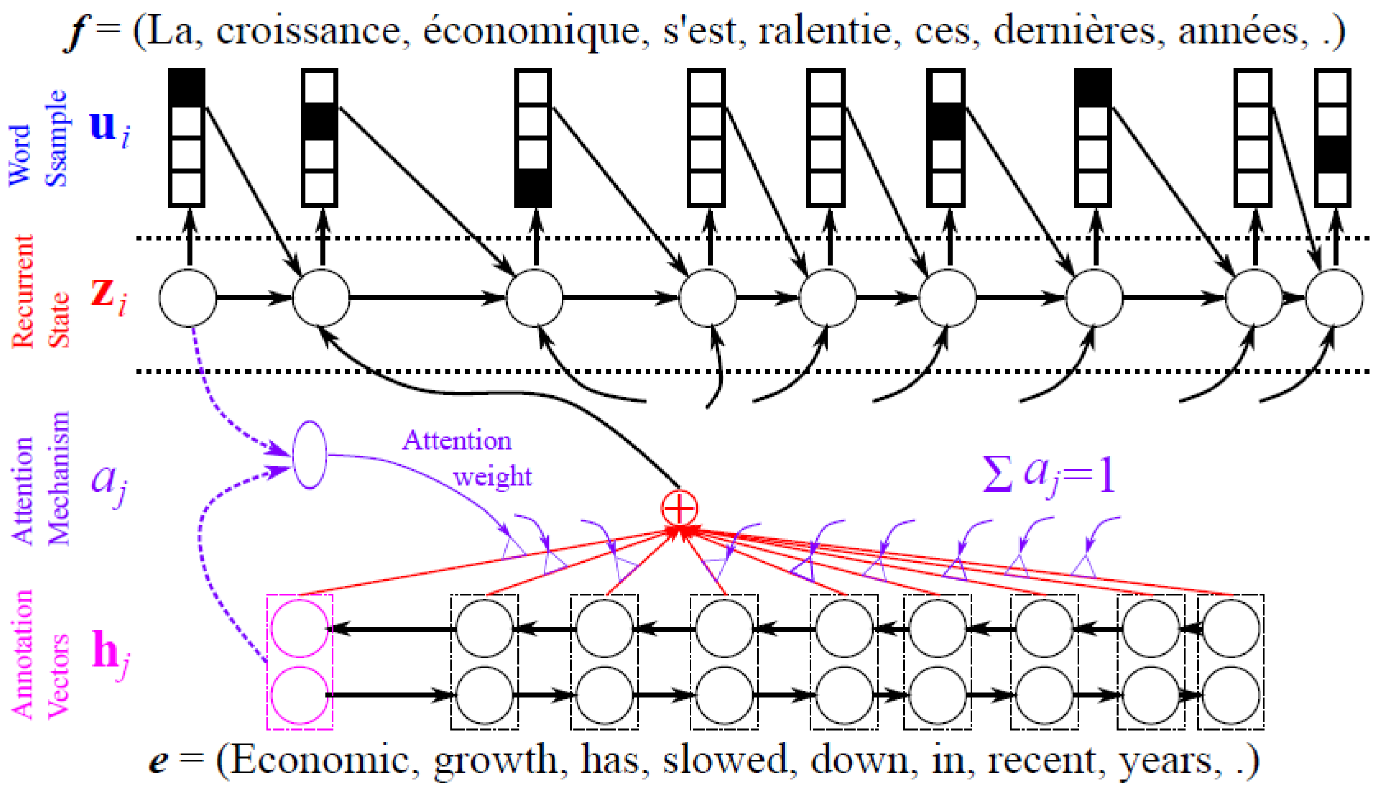
Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Alleviate fixed length encoding

- The decoder decides parts of the source sentence to pay attention to.

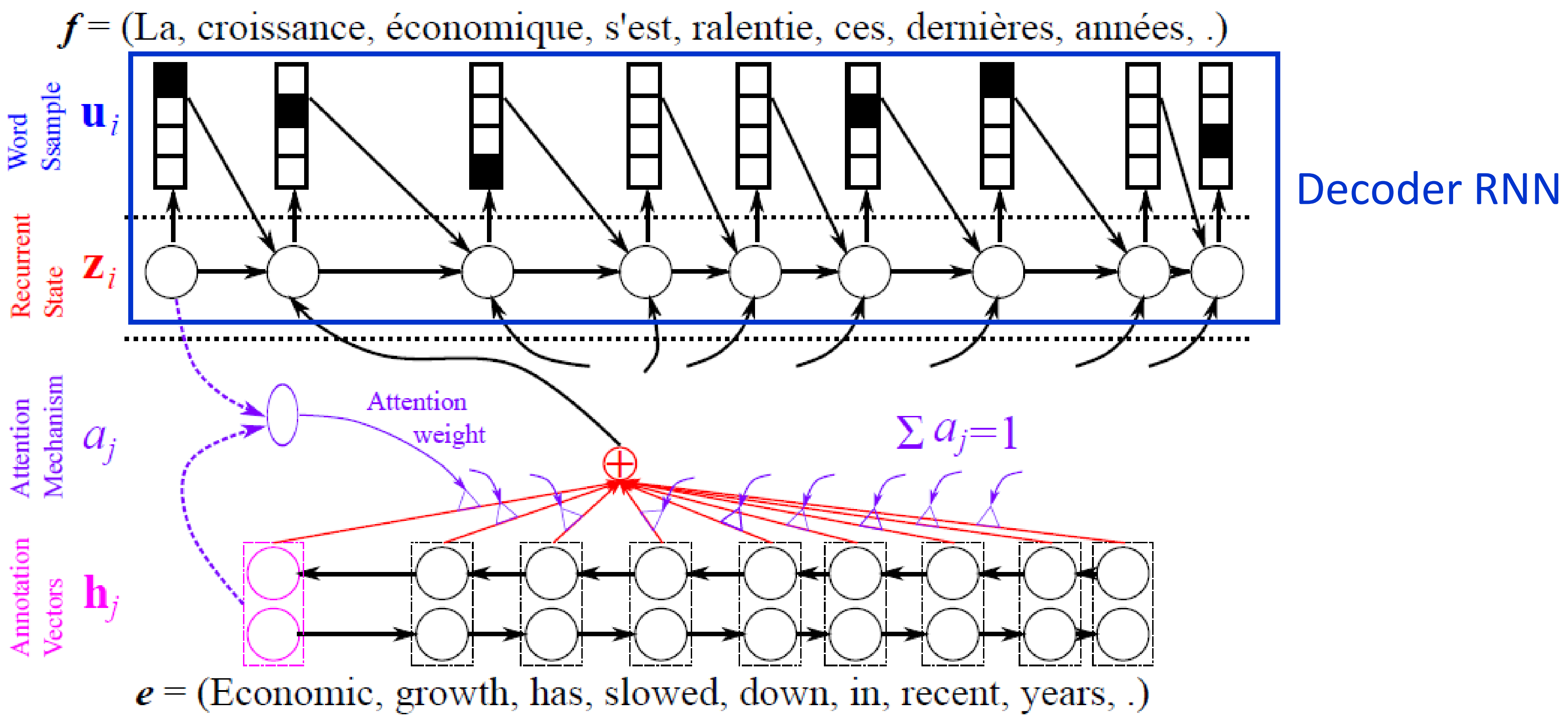
- By letting the decoder have an attention mechanism, we relieve the encoder from the burden of source sentence into a fixed length vector
  - the information can be spread throughout the sequence
  - and can be selectively retrieved by the decoder accordingly.

Soft Attention for Translation



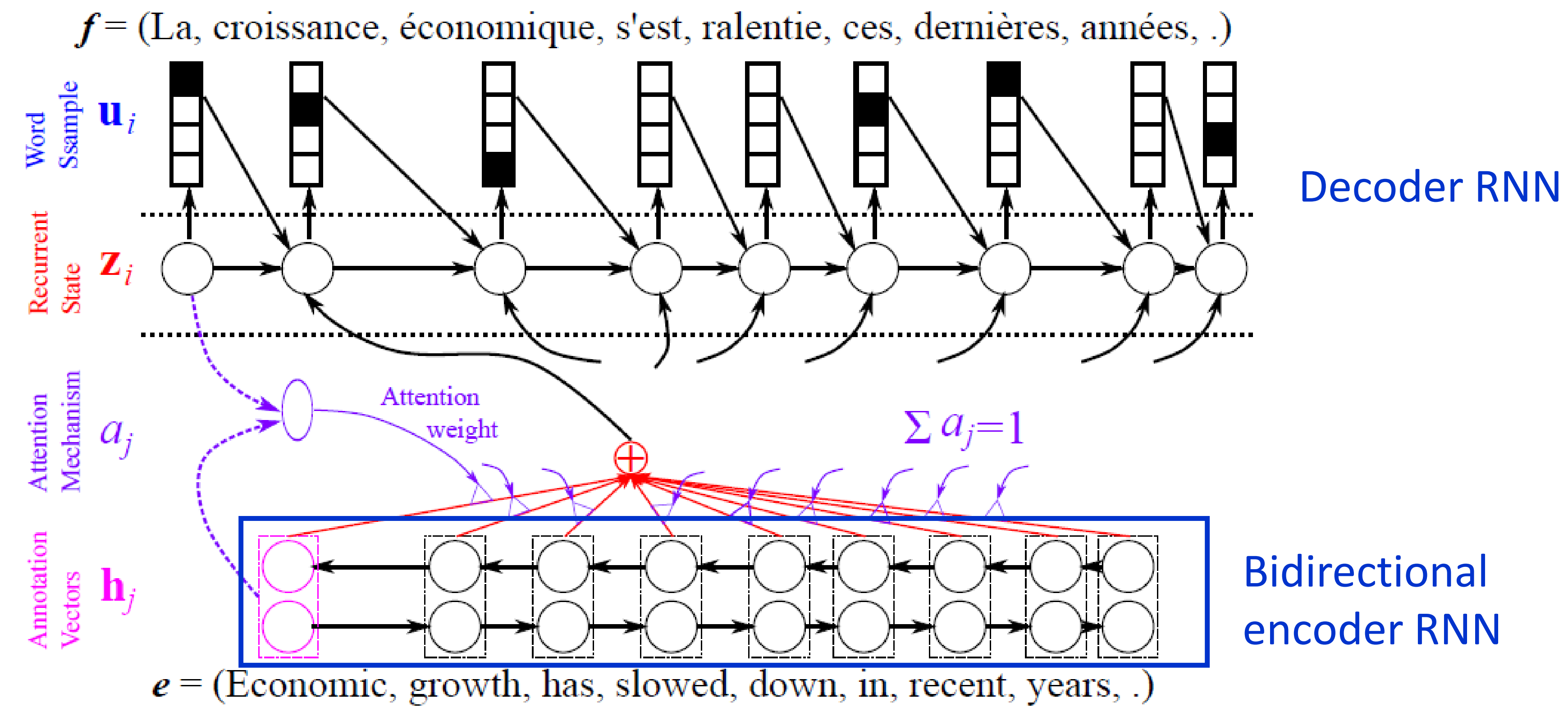
From Y. Bengio CVPR 2015 Tutorial

Soft Attention for Translation



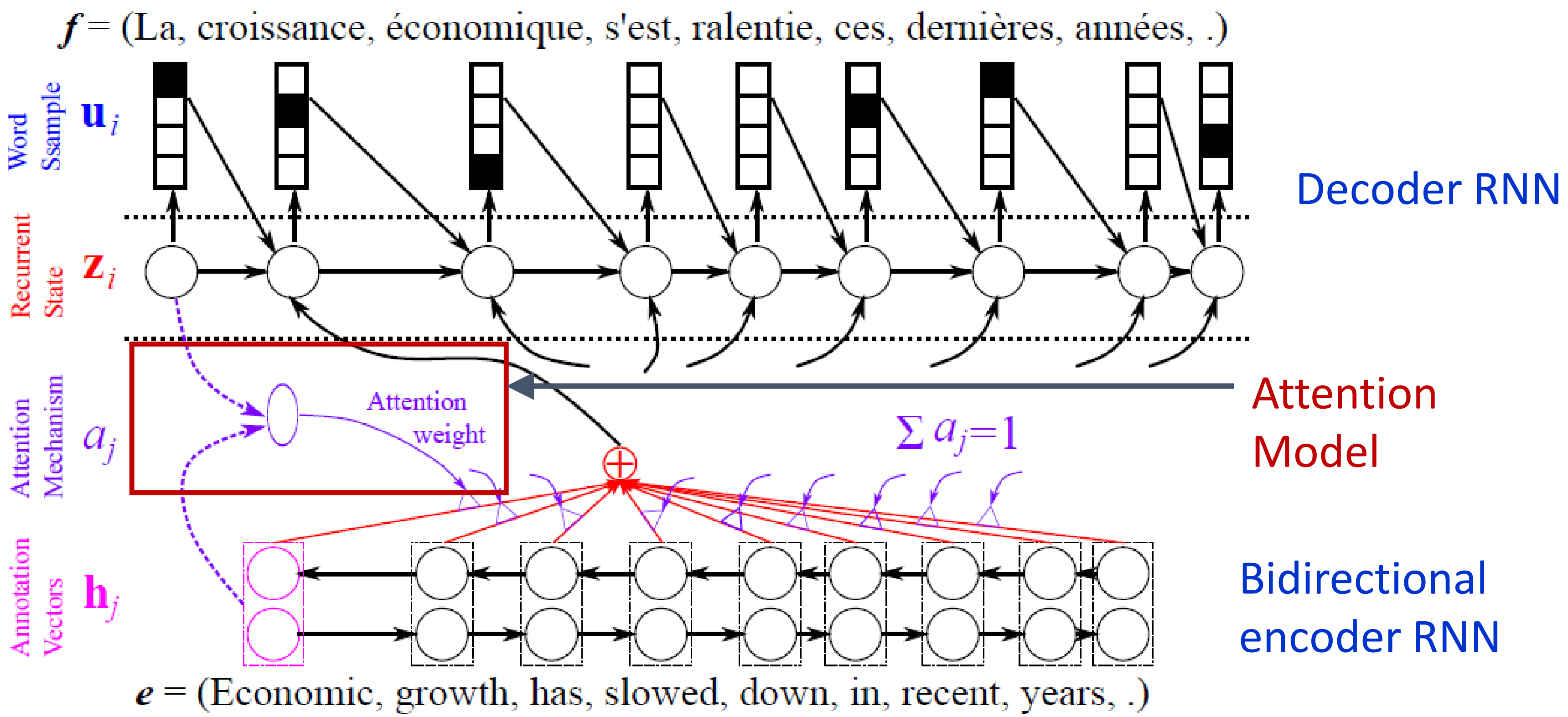
From Y. Bengio CVPR 2015 Tutorial

Soft Attention for Translation



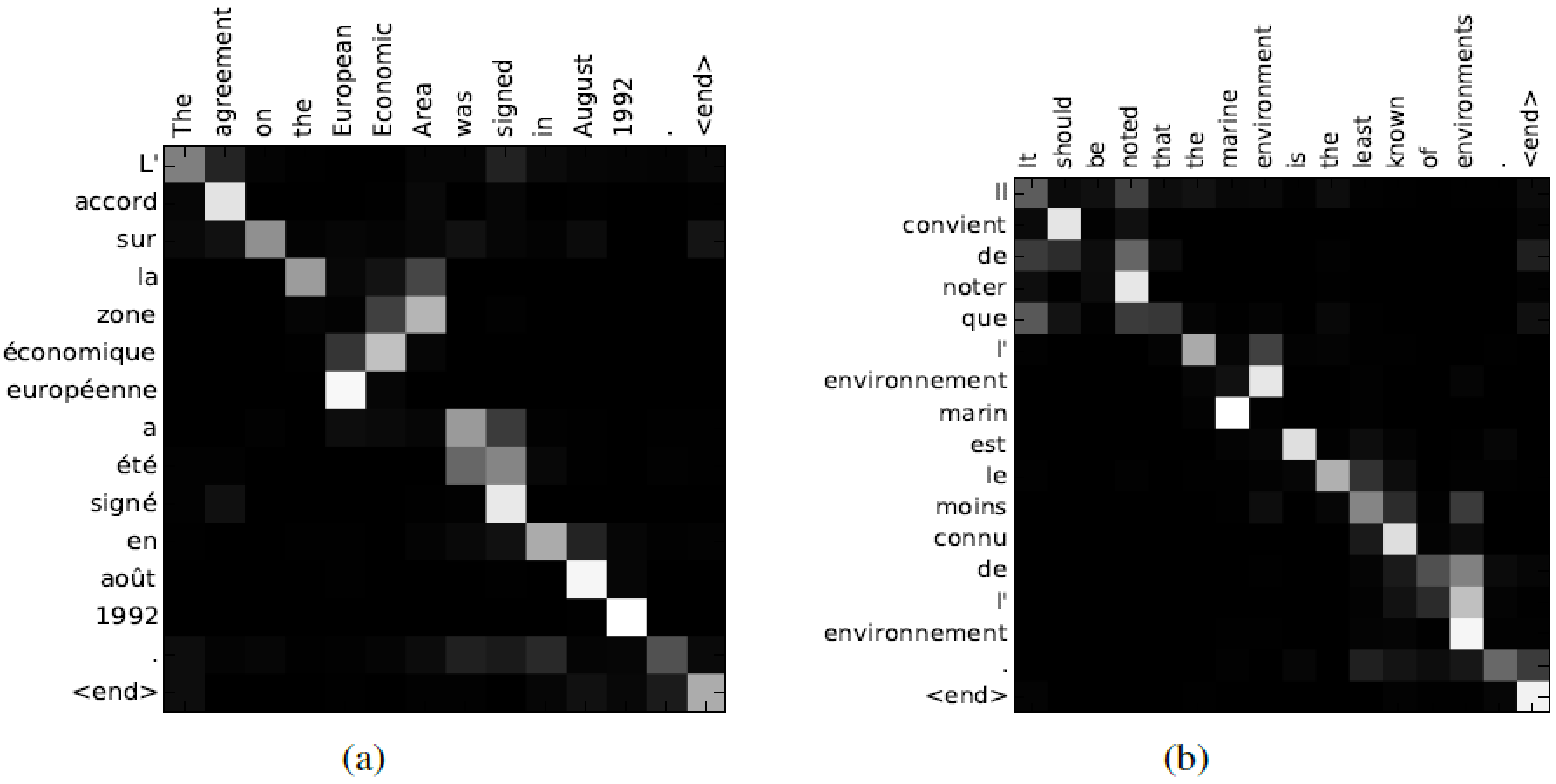
From Y. Bengio CVPR 2015 Tutorial

Soft Attention for Translation



From Y. Bengio CVPR 2015 Tutorial

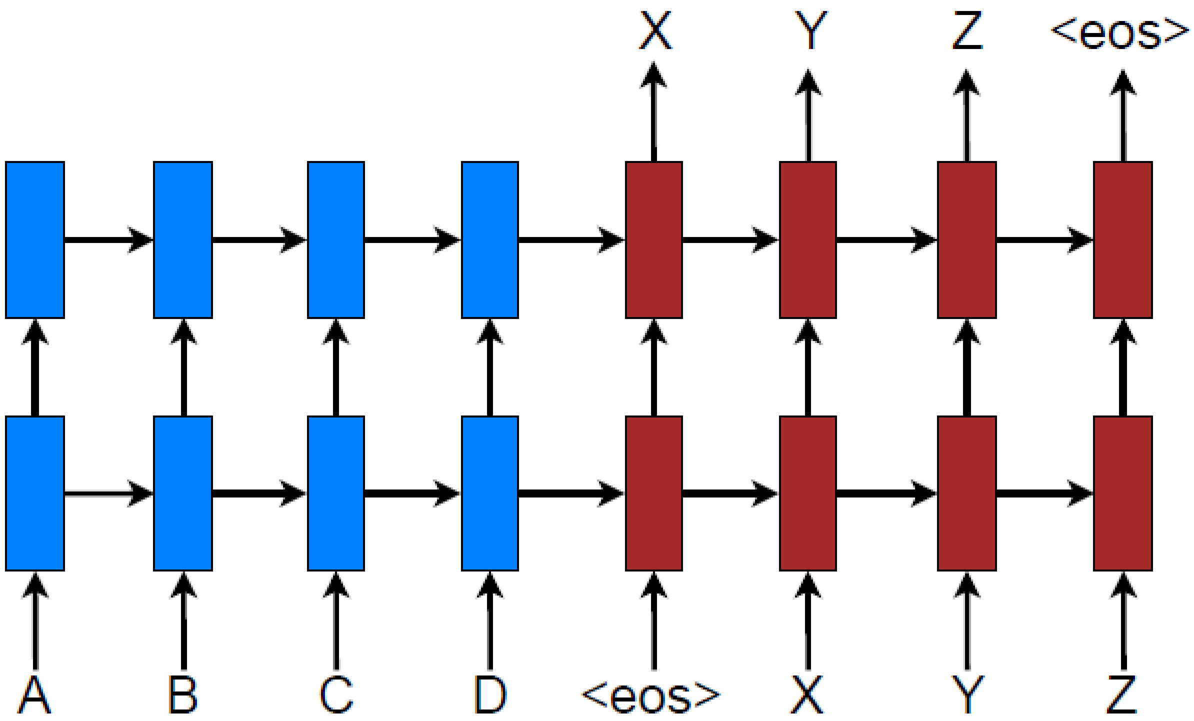
Soft Attention for Translation



Bahdanau et al, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR 2015

Luong, Pham and Manning 2015

Stacked LSTM (c.f. bidirectional flat encoder in Bahdanau et al):



Effective Approaches to Attention-based Neural Machine Translation
Minh-Thang Luong, Hieu Pham, Christopher D. Manning, EMNLP 15

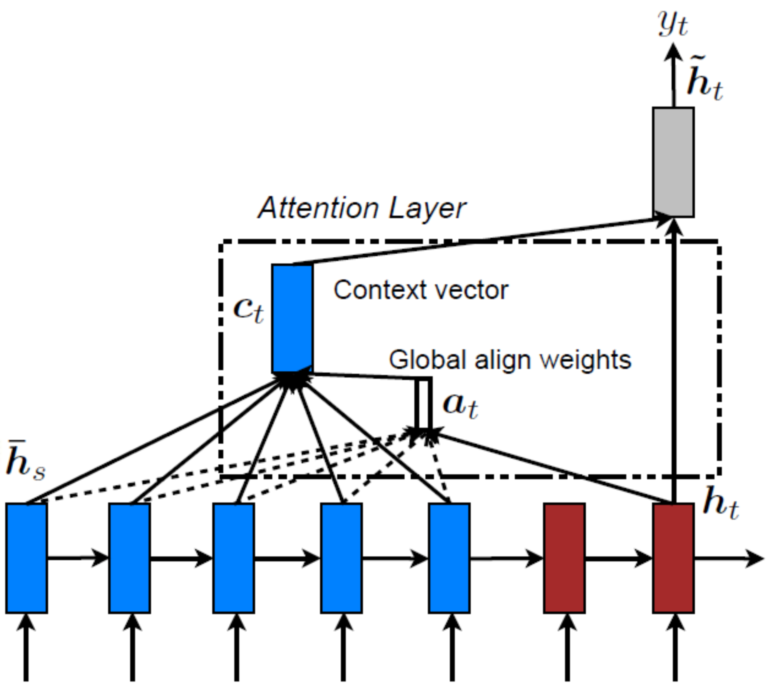
Global Attention Model

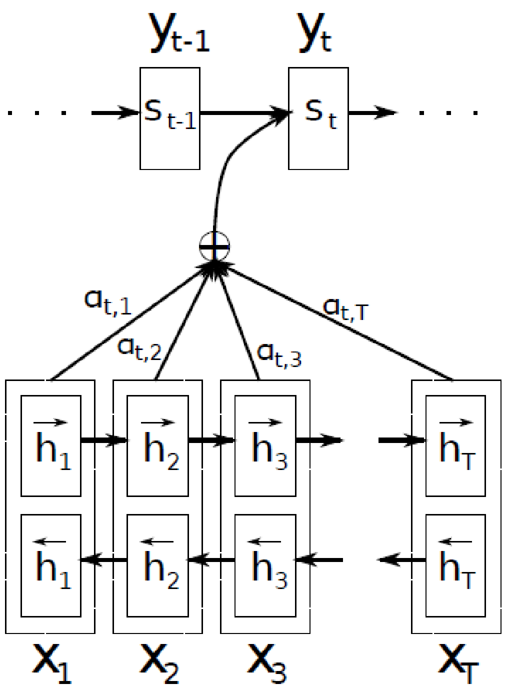
Global attention model is similar but simpler than Badanau's:

Different word matching functions were used

$$a_t(s) = \mathrm{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \frac{\exp\left(\mathrm{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)\right)}{\sum_{s'} \exp\left(\mathrm{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_{s'})\right)}$$

$$\mathrm{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \bar{\boldsymbol{h}}_s & \textit{dot} \\ \boldsymbol{h}_t^\top \boldsymbol{W}_a \bar{\boldsymbol{h}}_s & \textit{general} \\ \boldsymbol{v}_a^\top \tanh\left(\boldsymbol{W}_a [\boldsymbol{h}_t; \bar{\boldsymbol{h}}_s]\right) & \textit{concat} \end{cases}$$



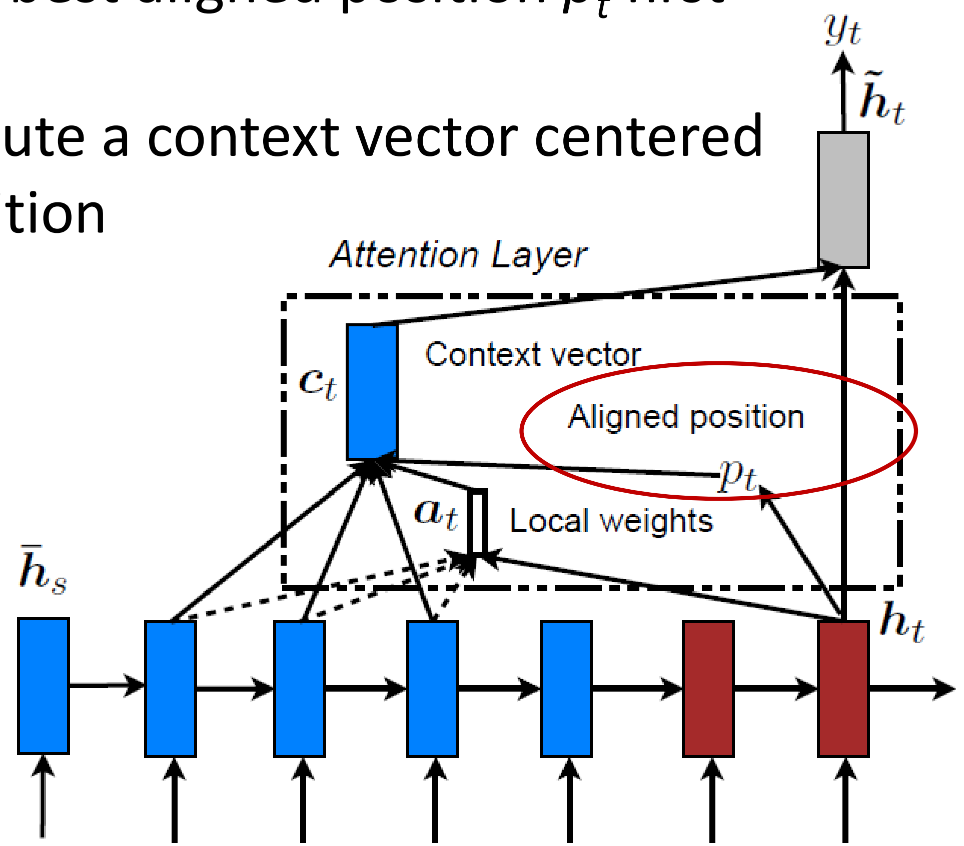


Effective Approaches to Attention-based Neural Machine Translation
Minh-Thang Luong Hieu Pham Christopher D. Manning, EMNLP 15

Local Attention Model

- Compute a best aligned position p_t first

- Then compute a context vector centered at that position



$$p_t = S \cdot \mathrm{sigmoid}(\boldsymbol{v}_p^\top \tanh(\boldsymbol{W_p h}_t)),$$

$$\boldsymbol{a}_t(s) = \mathrm{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) \exp\left(-\frac{(s-p_t)^2}{2\sigma^2}\right)$$

Effective Approaches to Attention-based Neural Machine Translation
Minh-Thang Luong Hieu Pham Christopher D. Manning, EMNLP 15

Results

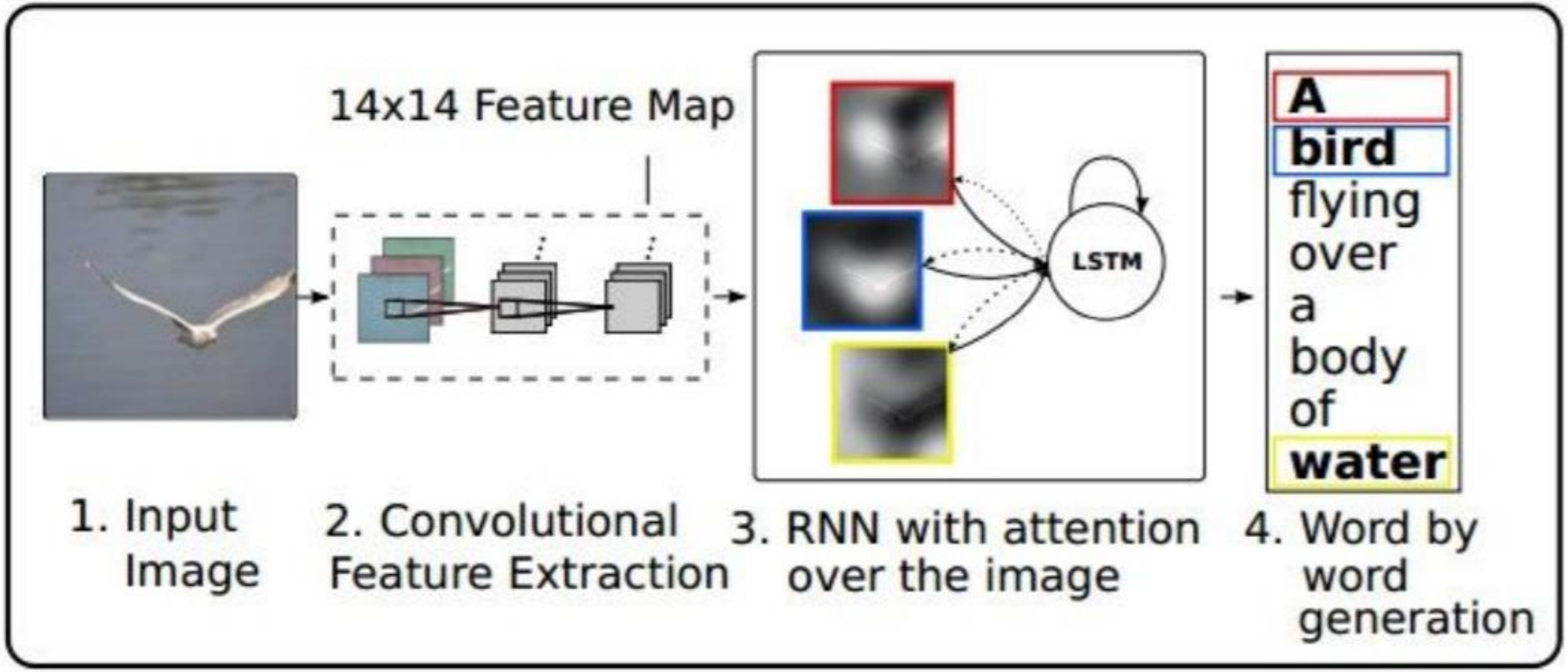
System	Ppl	BLEU
Winning WMT'14 system – *phrase-based* + *large LM* (Buck et al., 2014)		20.7
Existing NMT systems		
RNNsearch (Jean et al., 2015)		16.5
RNNsearch + unk replace (Jean et al., 2015)		19.0
RNNsearch + unk replace + large vocab + *ensemble* 8 models (Jean et al., 2015)		**21.6**
Our NMT systems		
Base	10.6	11.3
Base + reverse	9.9	12.6 (+*1.3*)
Base + reverse + dropout	8.1	14.0 (+*1.4*)
Base + reverse + dropout + global attention (*location*)	7.3	16.8 (+*2.8*)
Base + reverse + dropout + global attention (*location*) + feed input	6.4	18.1 (+*1.3*)
Base + reverse + dropout + local-p attention (*general*) + feed input	5.9	19.0 (+*0.9*)
Base + reverse + dropout + local-p attention (*general*) + feed input + unk replace		20.9 (+*1.9*)
Ensemble 8 models + unk replace		**23.0** (+*2.1*)

Local ***and*** global models

Effective Approaches to Attention-based Neural Machine Translation
Minh-Thang Luong Hieu Pham Christopher D. Manning, EMNLP 15

Image Captioning with Attention

RNN focuses its attention at a different spatial location when generating each word

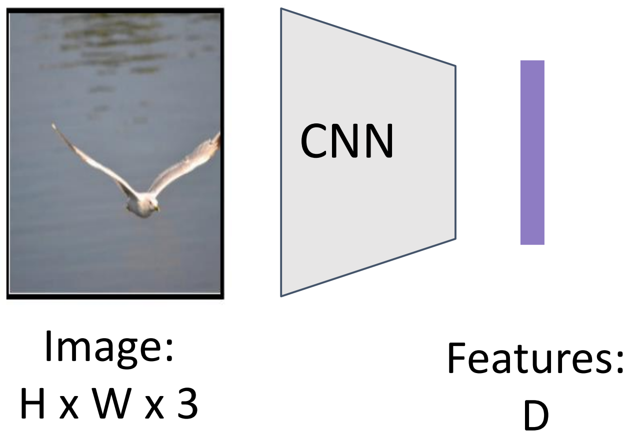


Xu et al, "Show, Attend, and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

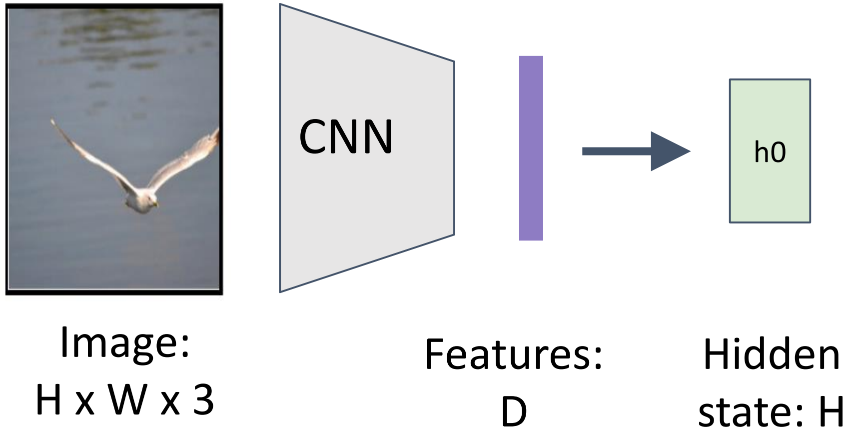
Recall: RNN for Captioning

Image:
H x W x 3

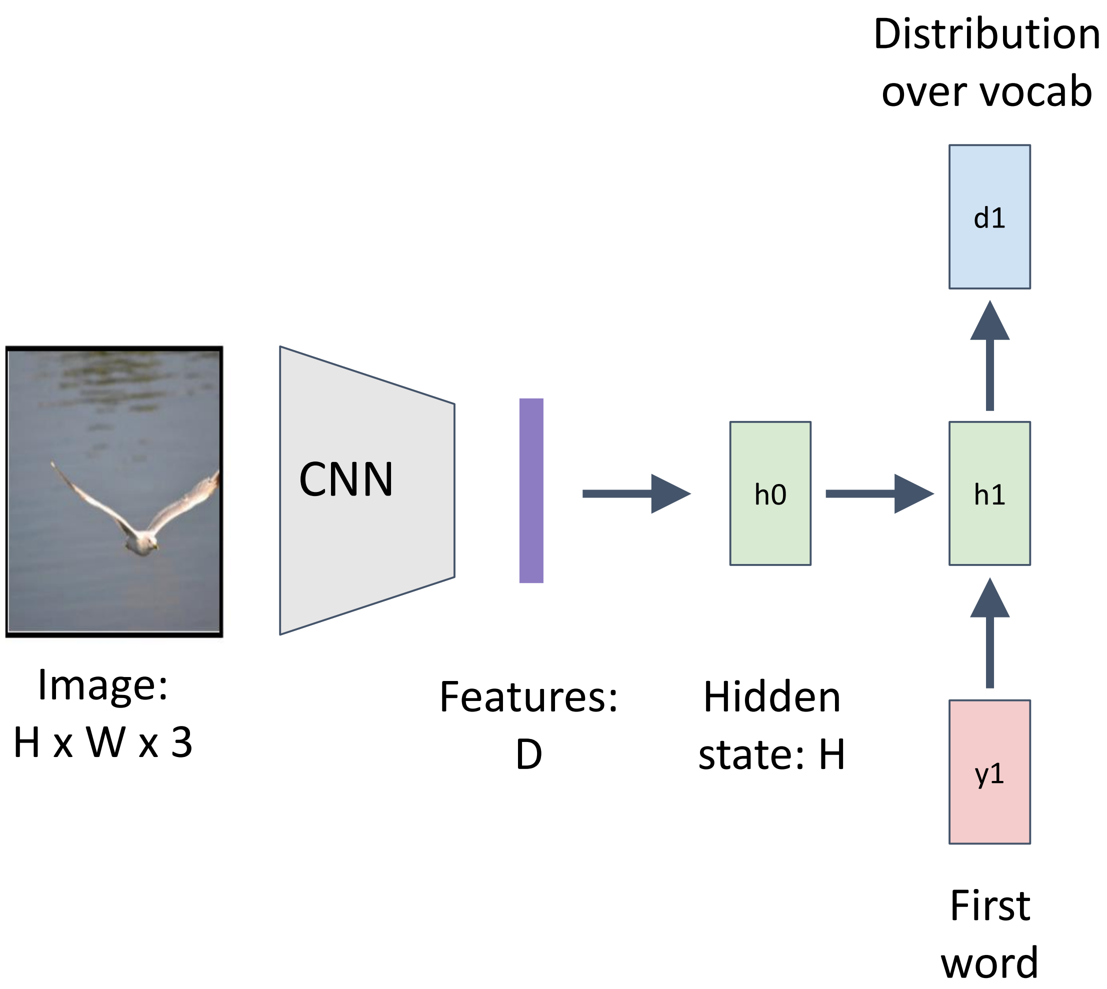
Recall: RNN for Captioning



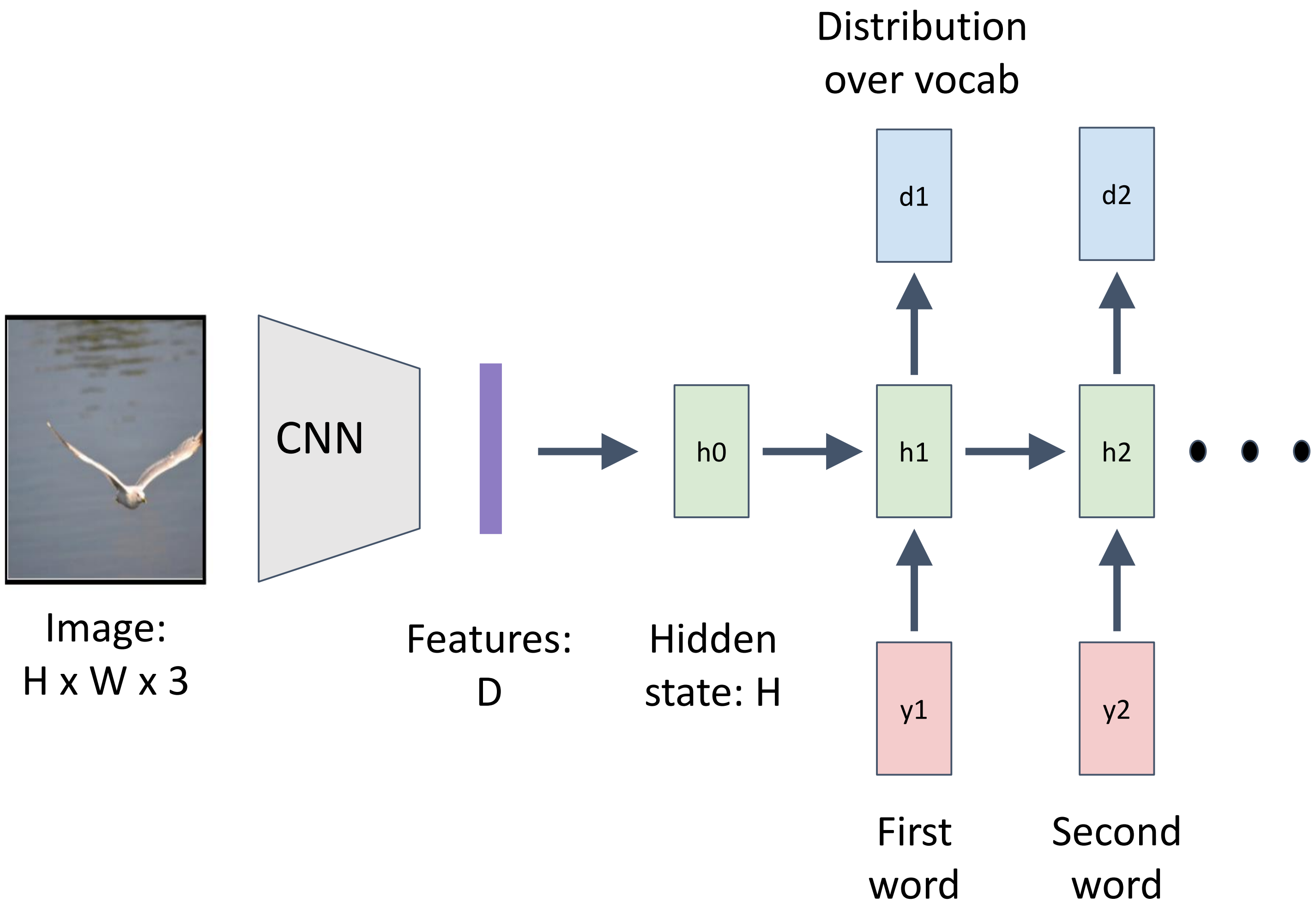
Recall: RNN for Captioning



Recall: RNN for Captioning

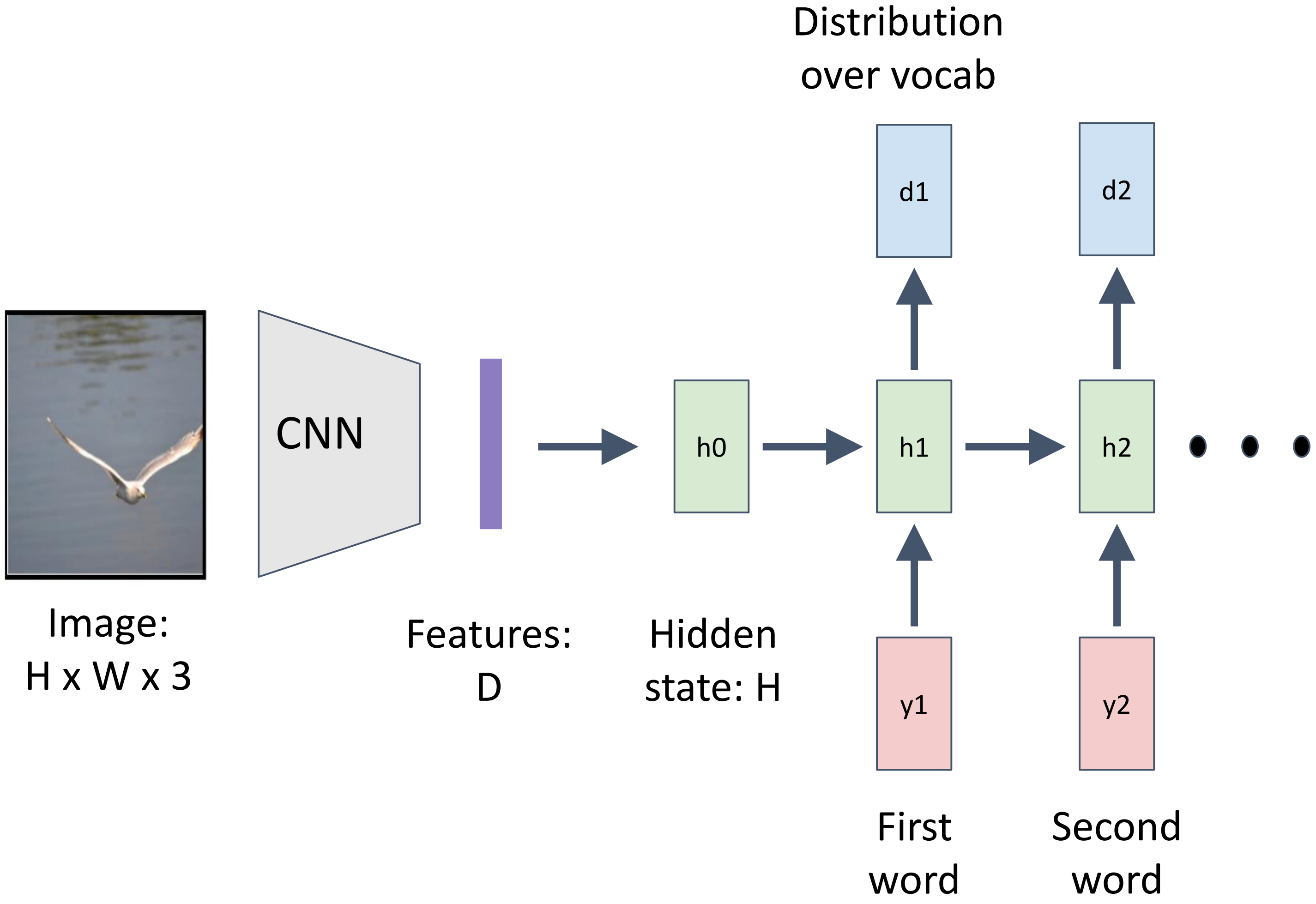


Recall: RNN for Captioning

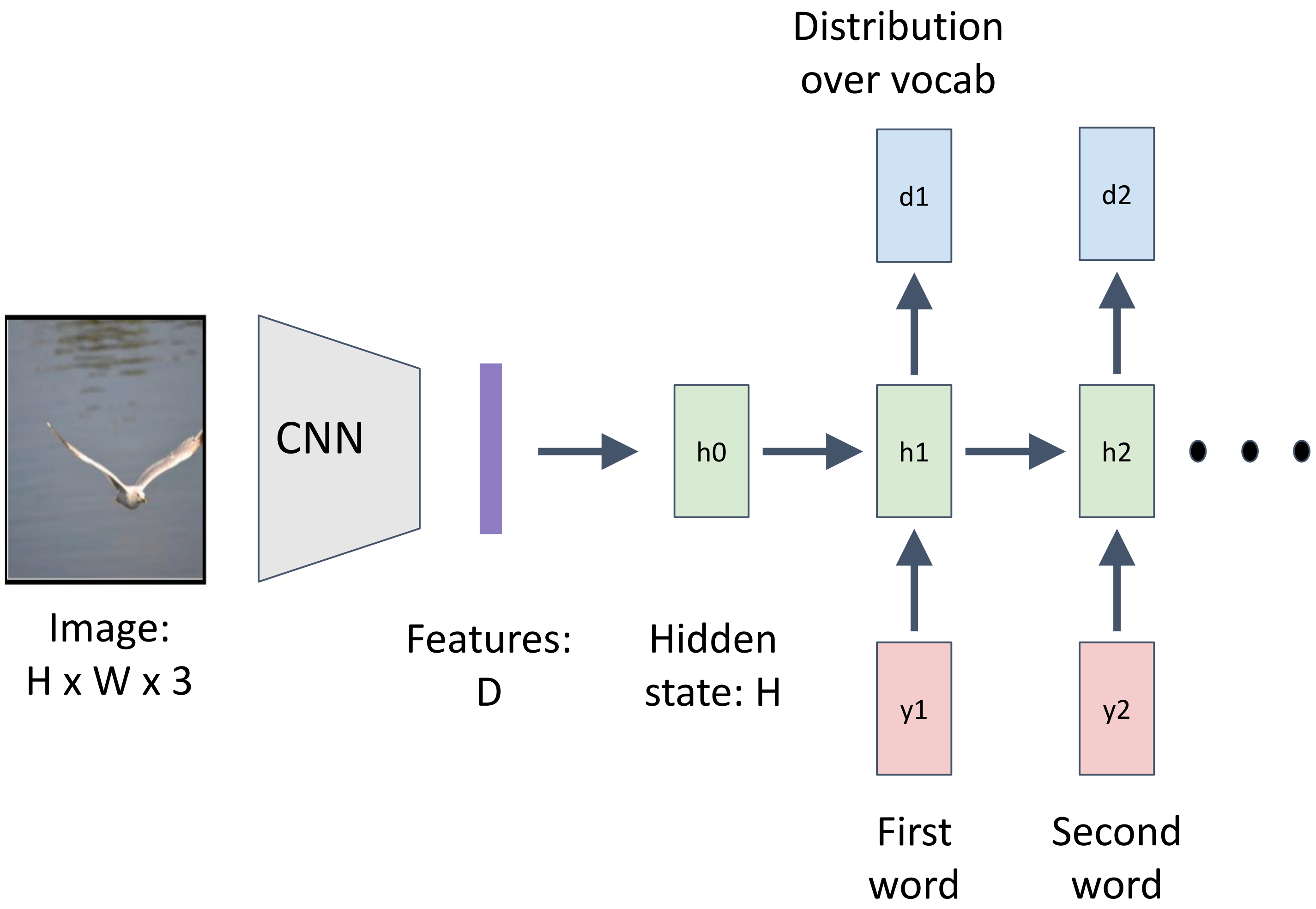


Recall: RNN for Captioning

RNN only looks at whole image, once



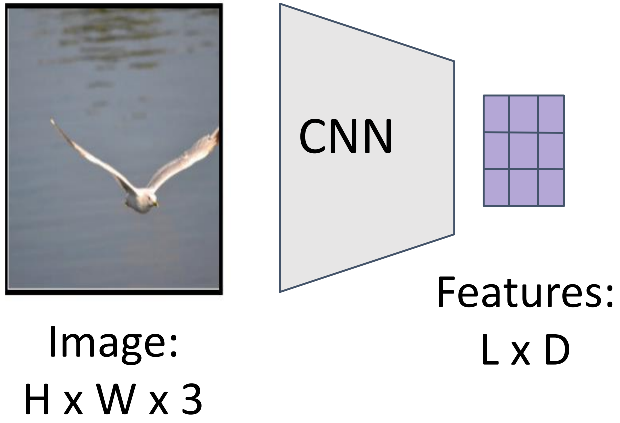
Recall: RNN for Captioning



RNN only looks at whole image, once

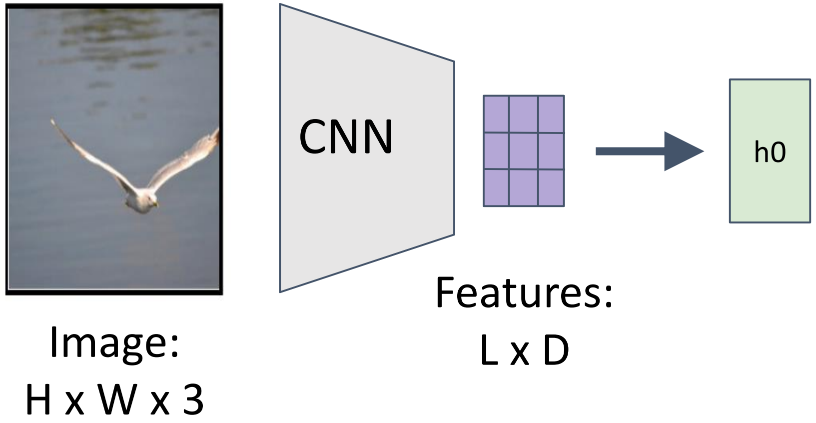
What if the RNN looks at different parts of the image at each timestep?

Soft Attention for Captioning



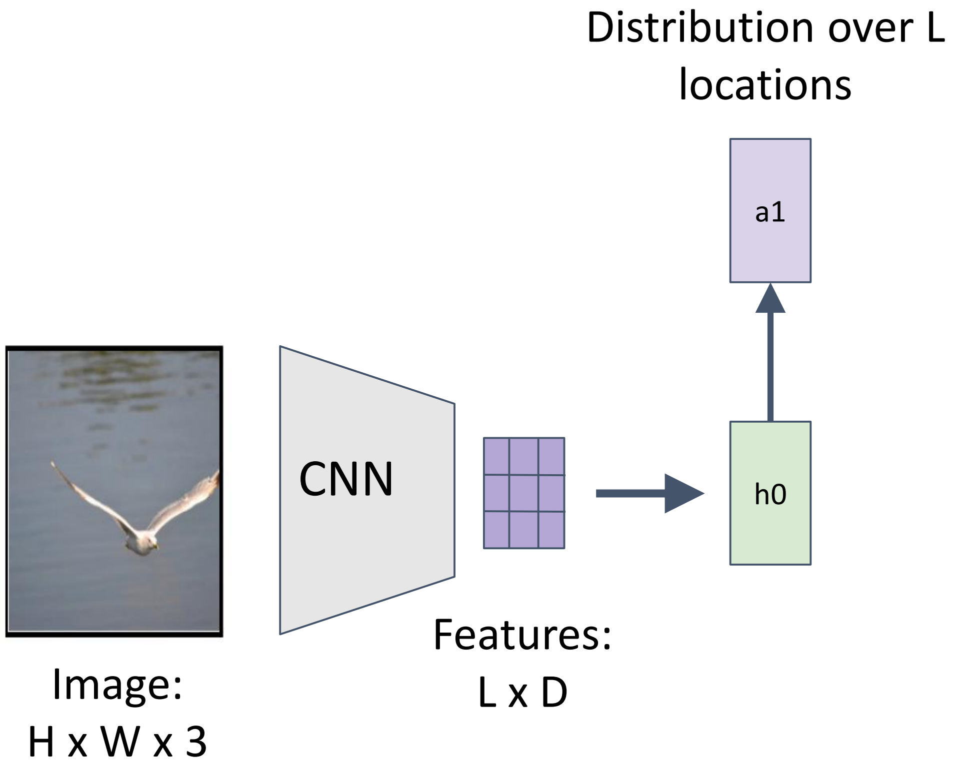
Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Soft Attention for Captioning



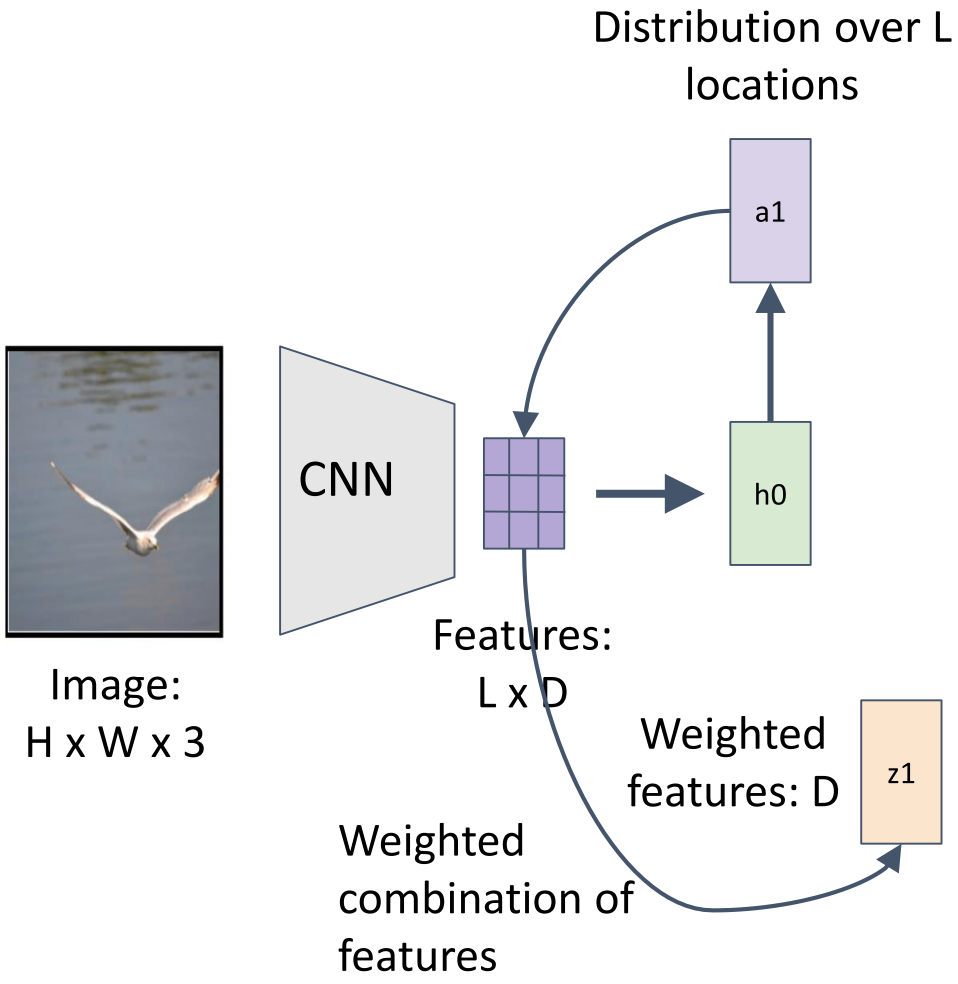
Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Soft Attention for Captioning

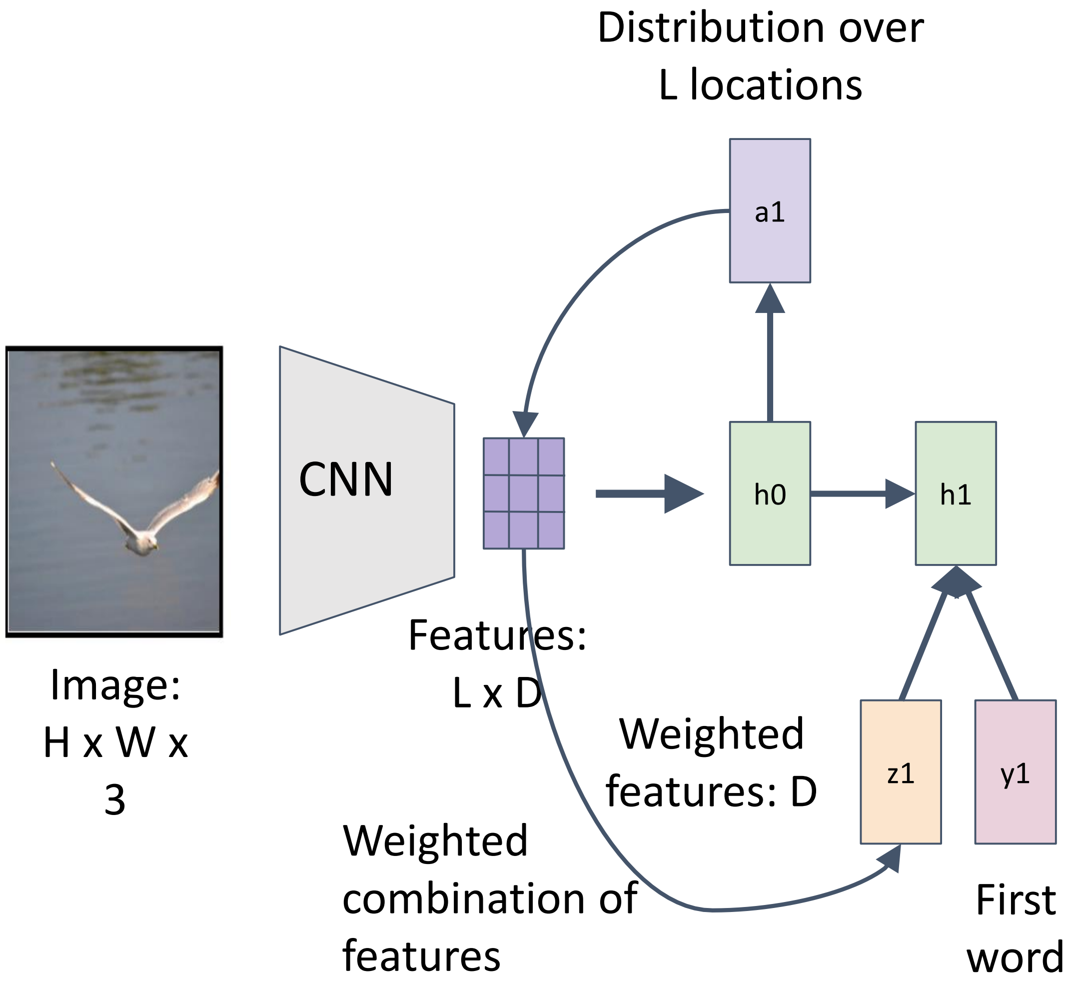


Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

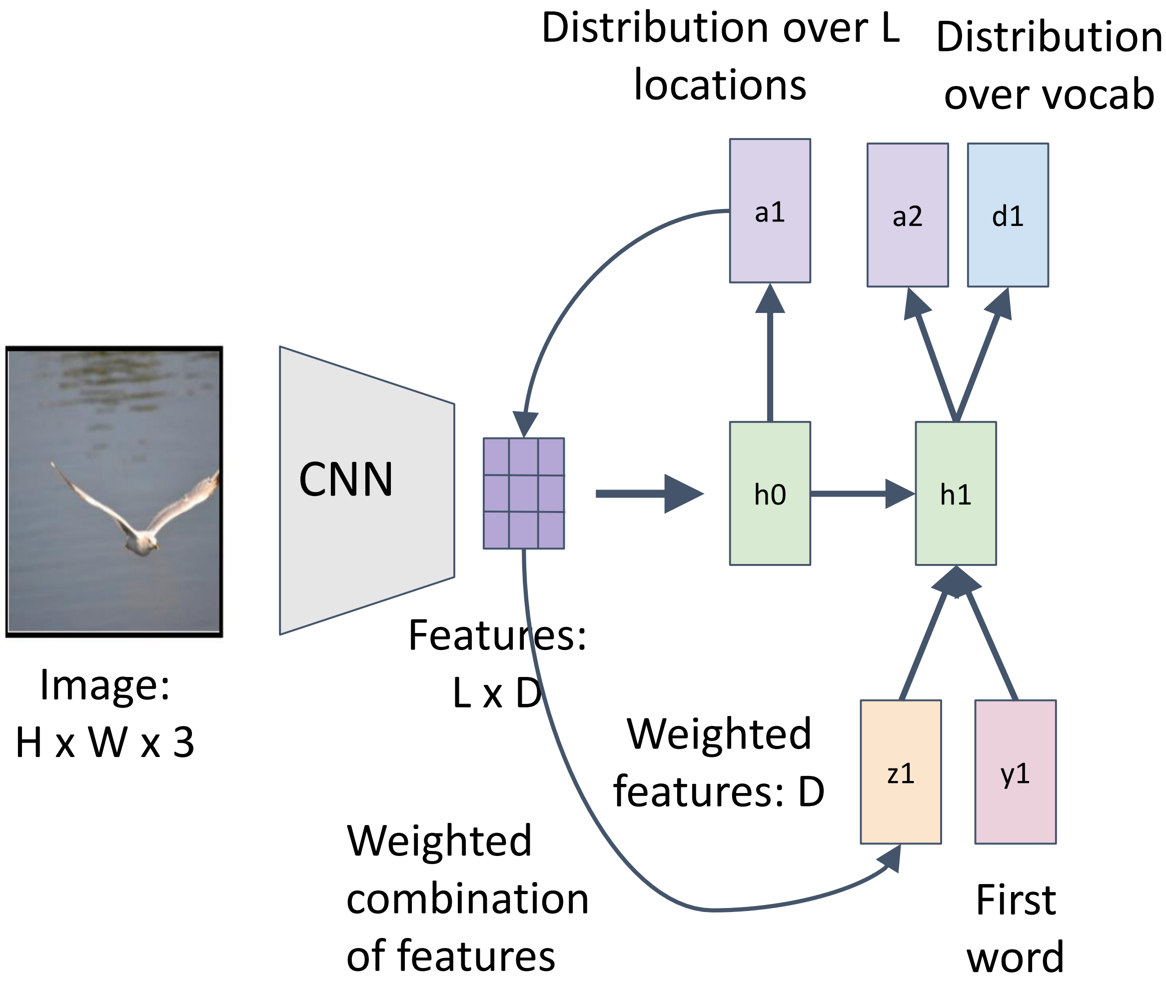
Soft Attention for Captioning



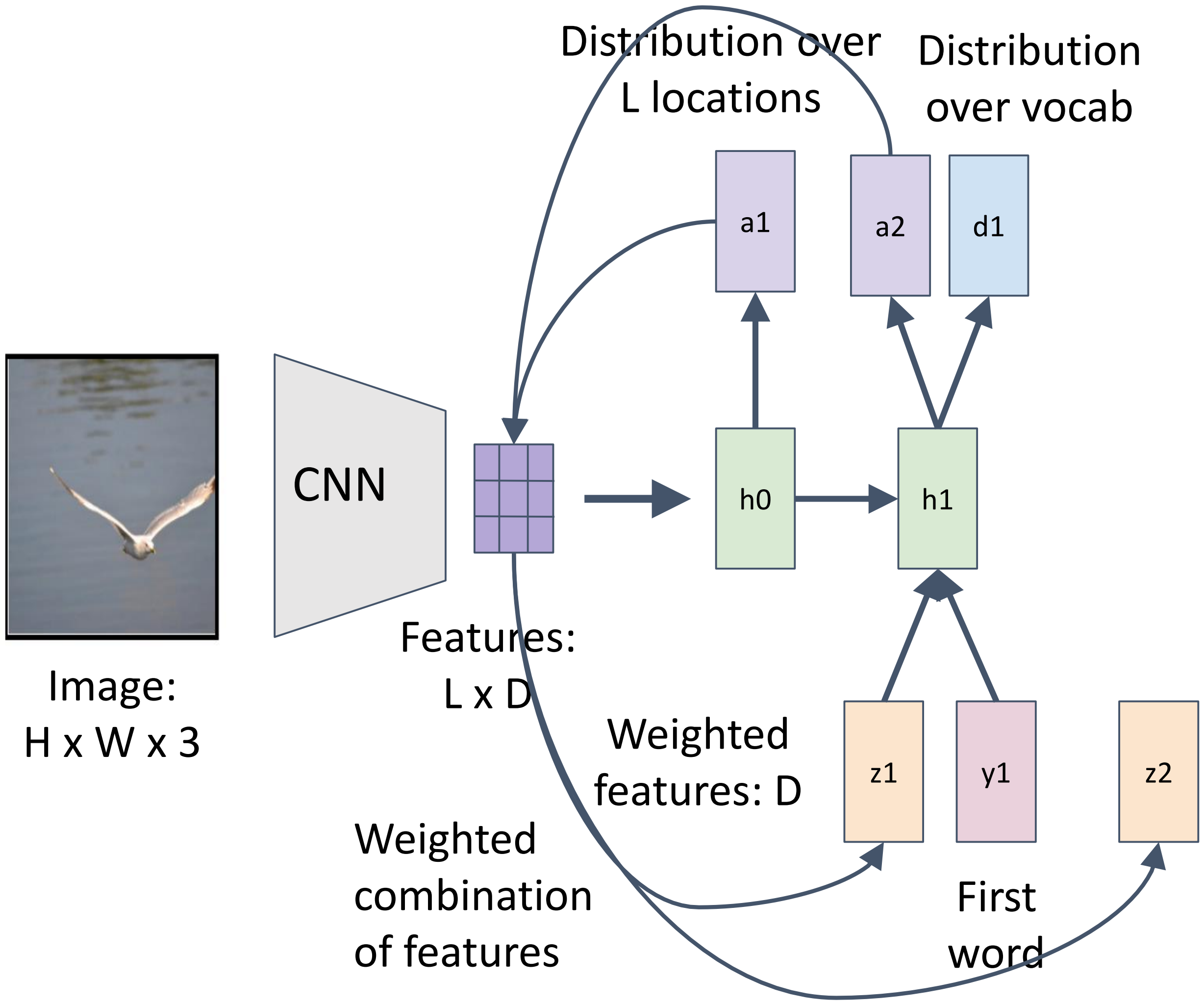
Soft Attention for Captioning



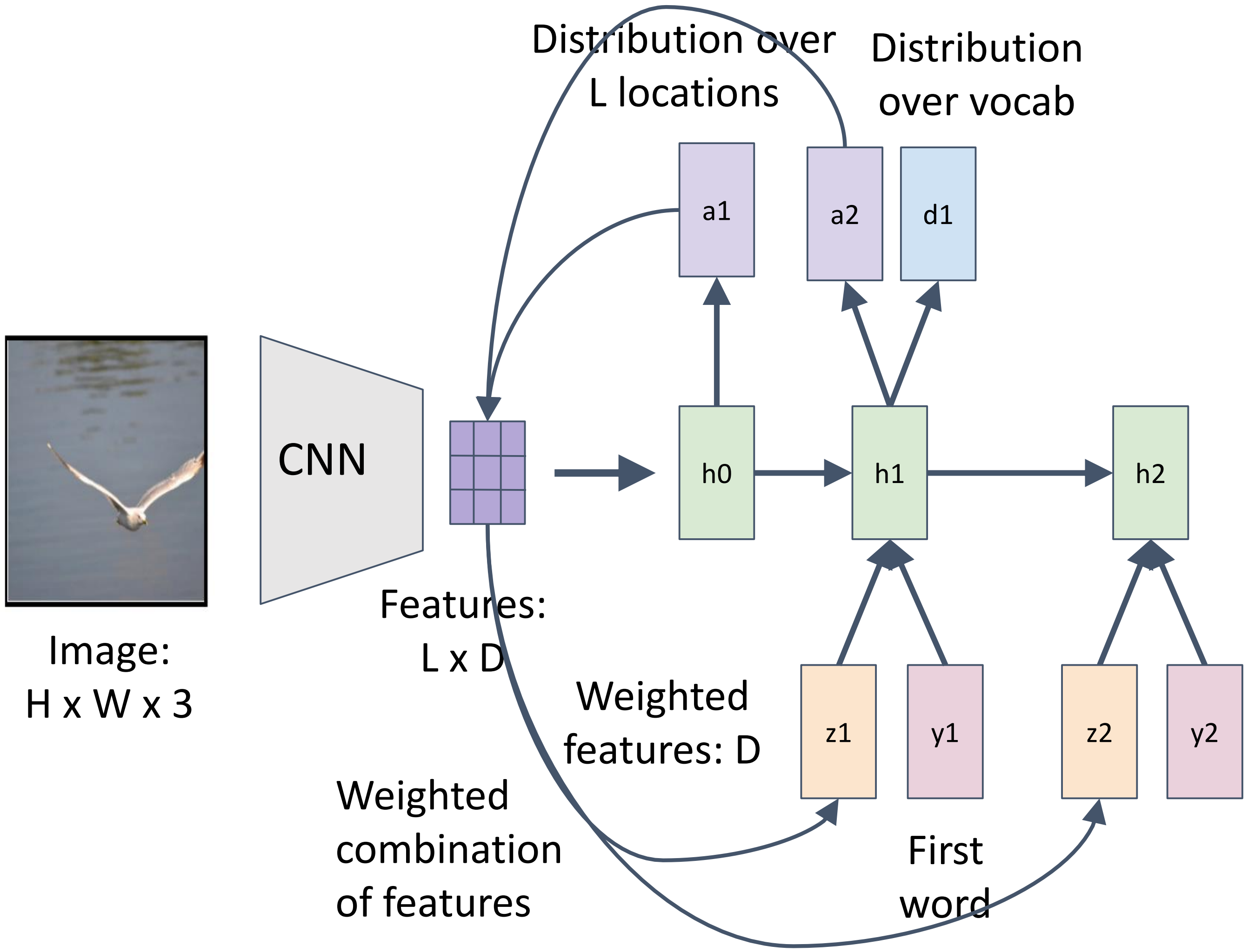
Soft Attention for Captioning



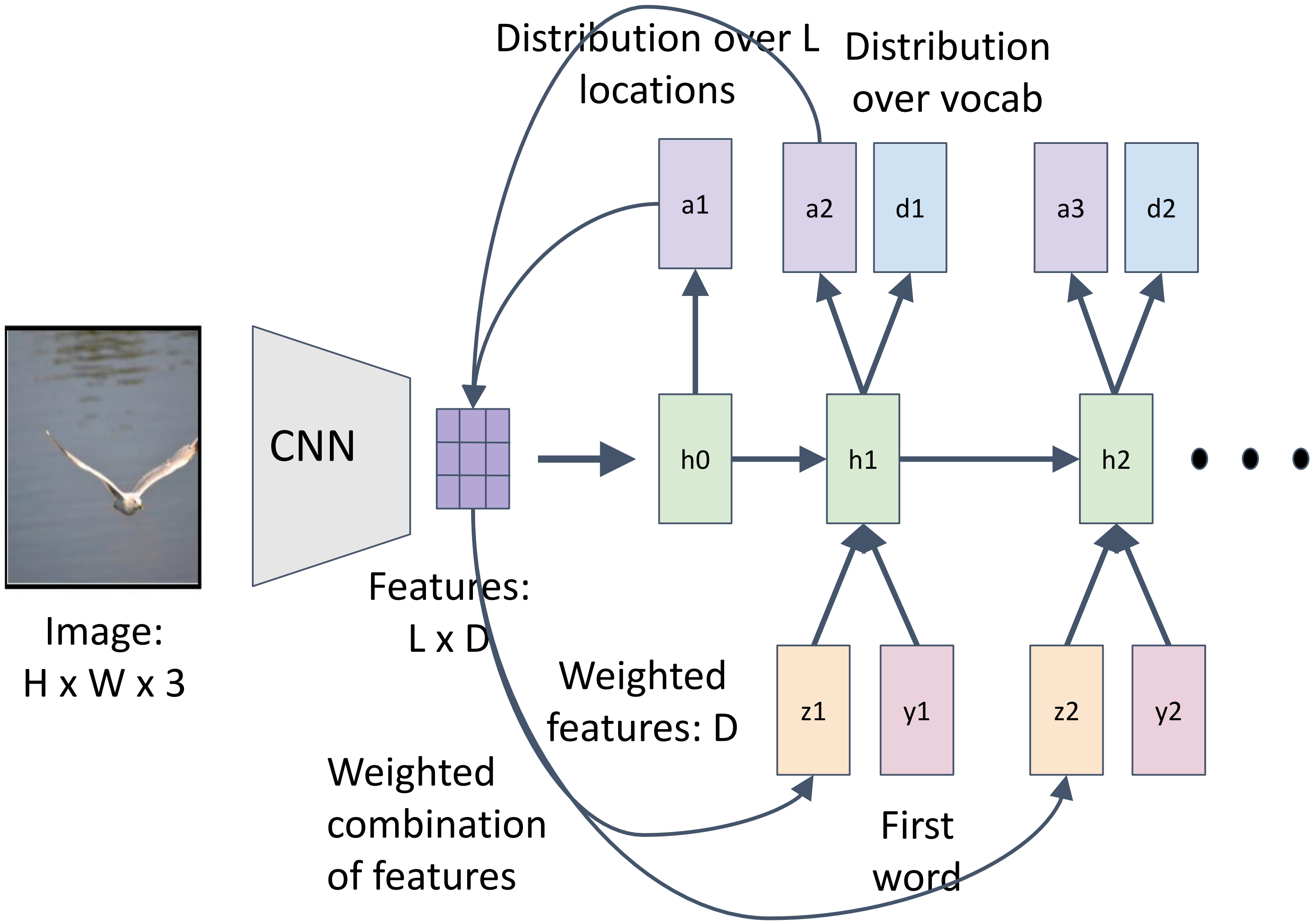
Soft Attention for Captioning



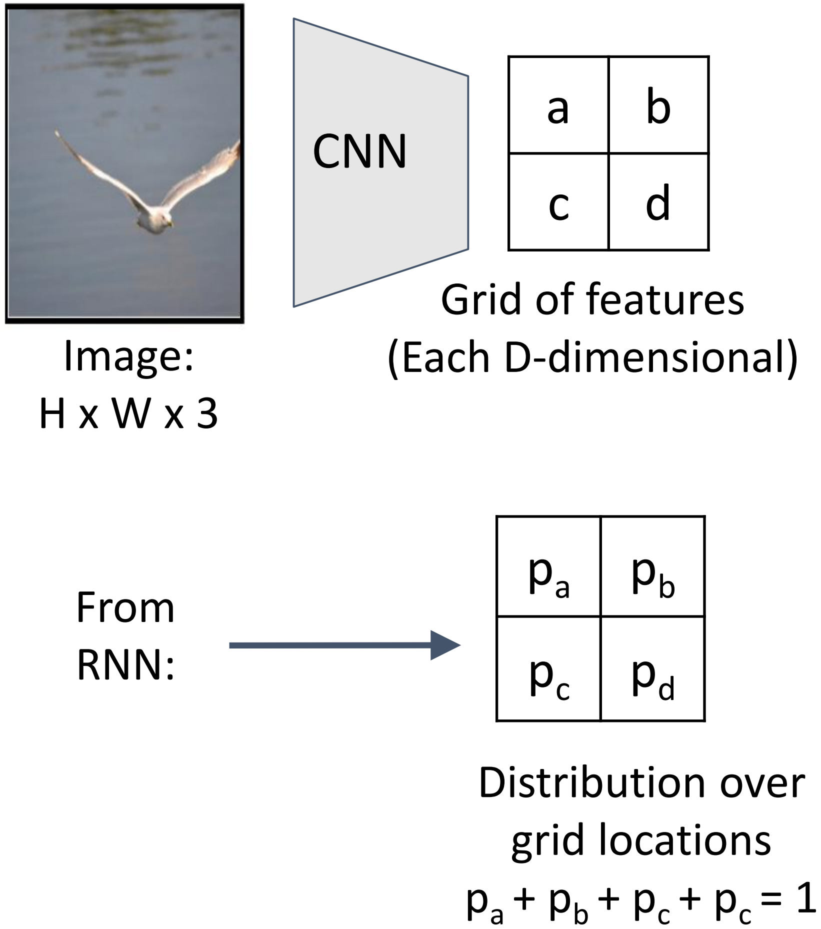
Soft Attention for Captioning



Soft Attention for Captioning

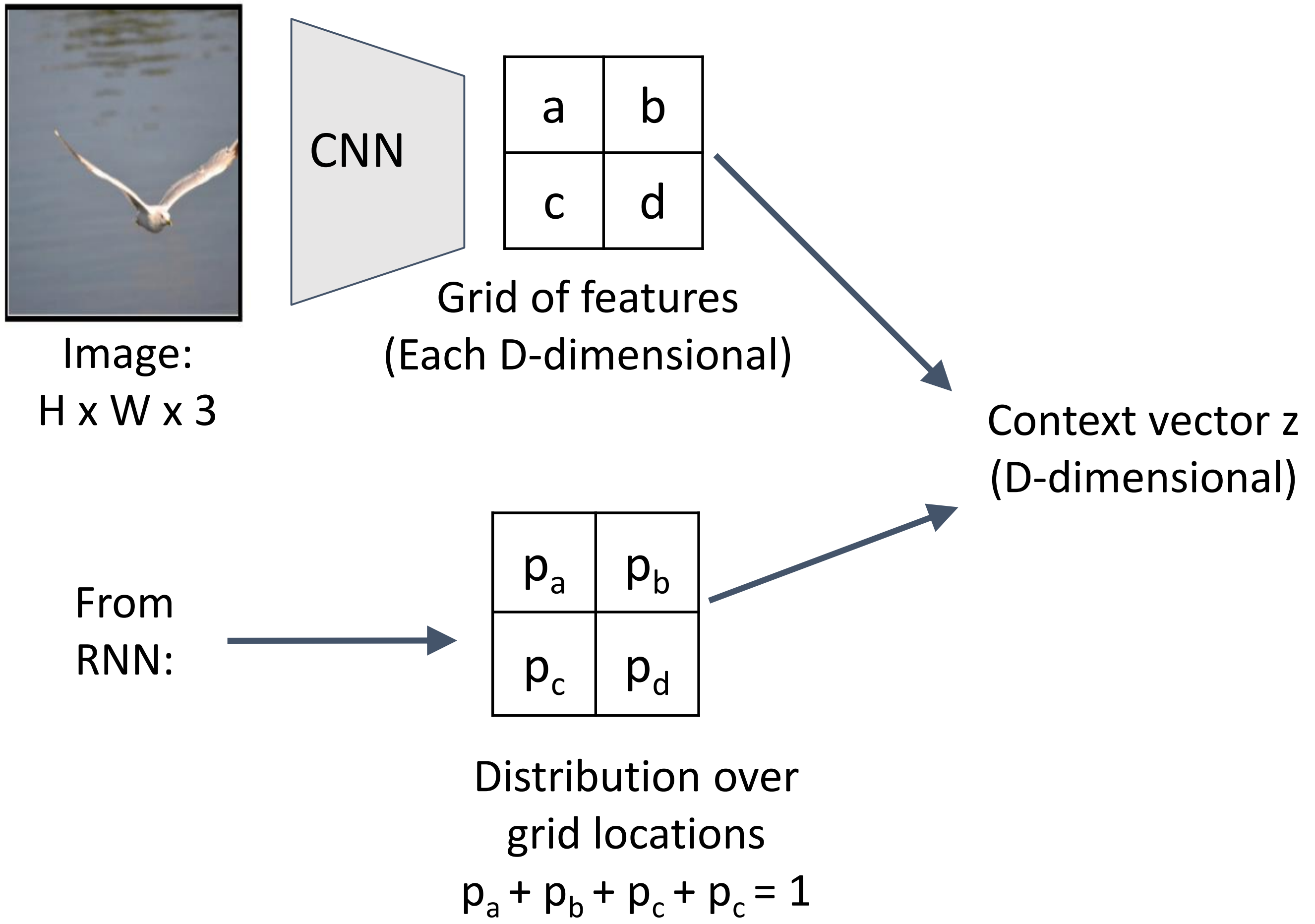


Soft vs Hard Attention



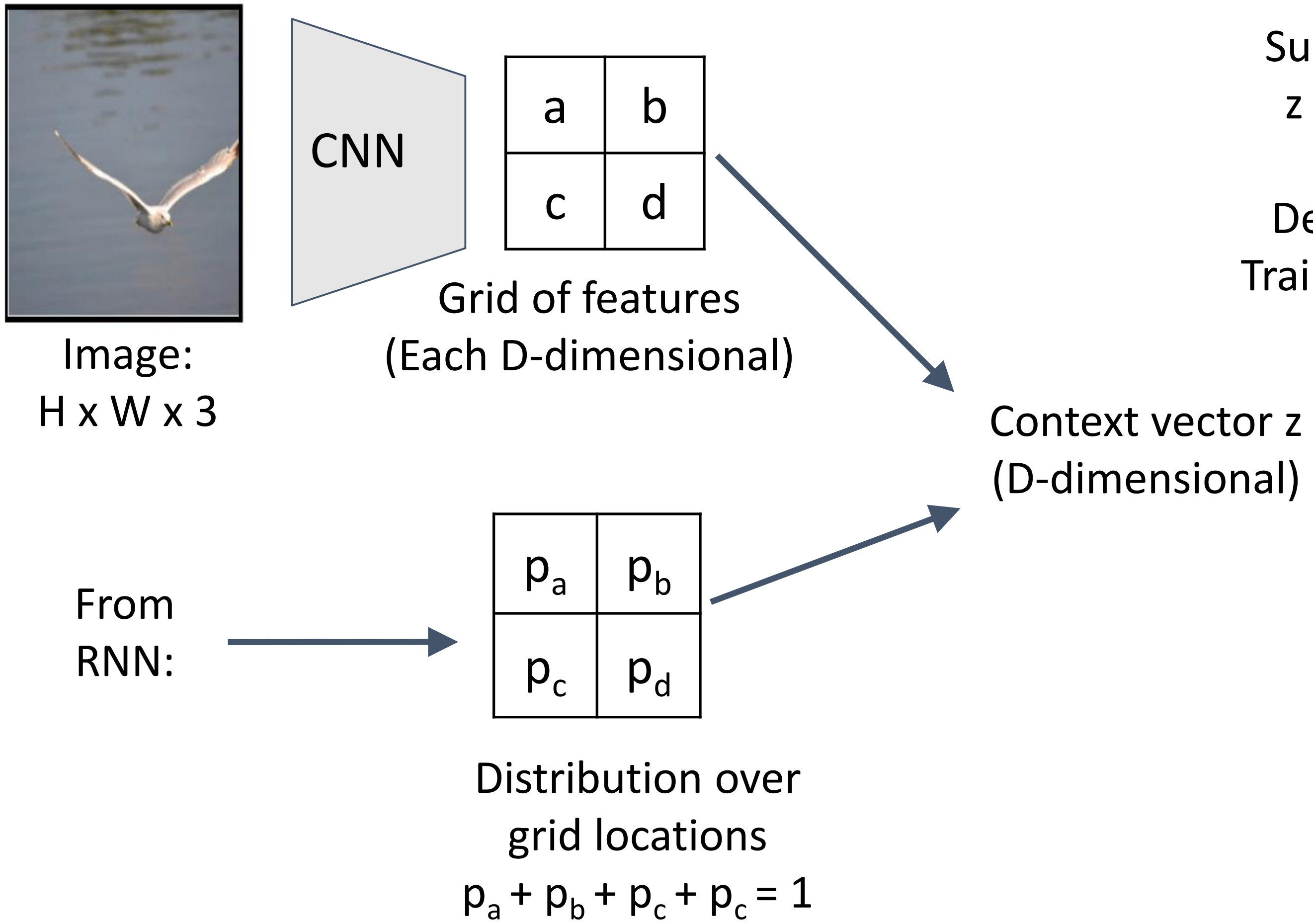
Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft vs Hard Attention



Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft vs Hard Attention



Soft attention:

Summarize ALL locations

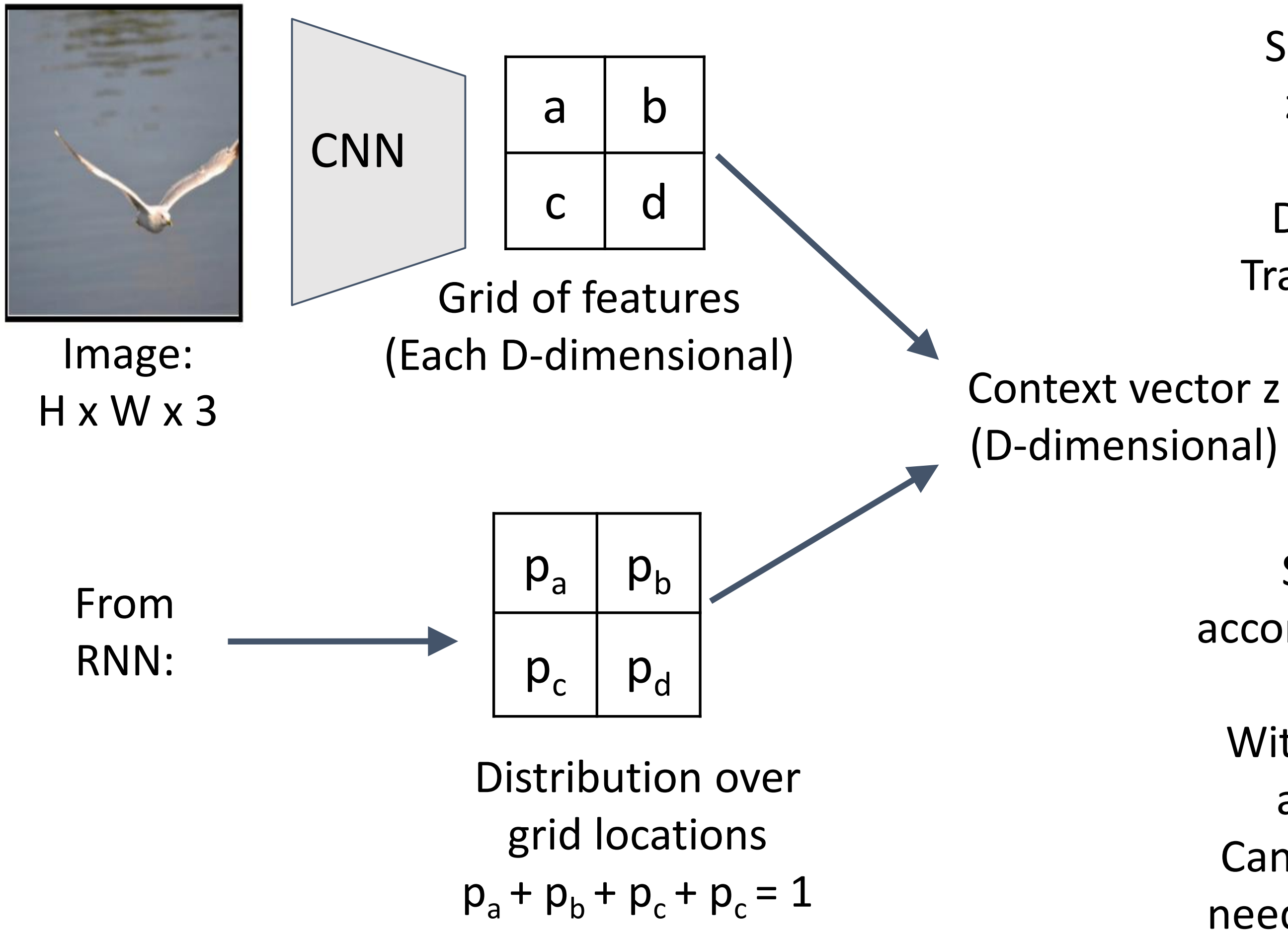
$z = p_a a + p_b b + p_c c + p_d d$

Derivative dz/dp is nice!

Train with gradient descent

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Soft vs Hard Attention



Soft attention:
Summarize ALL locations
$z = p_a a + p_b b + p_c c + p_d d$

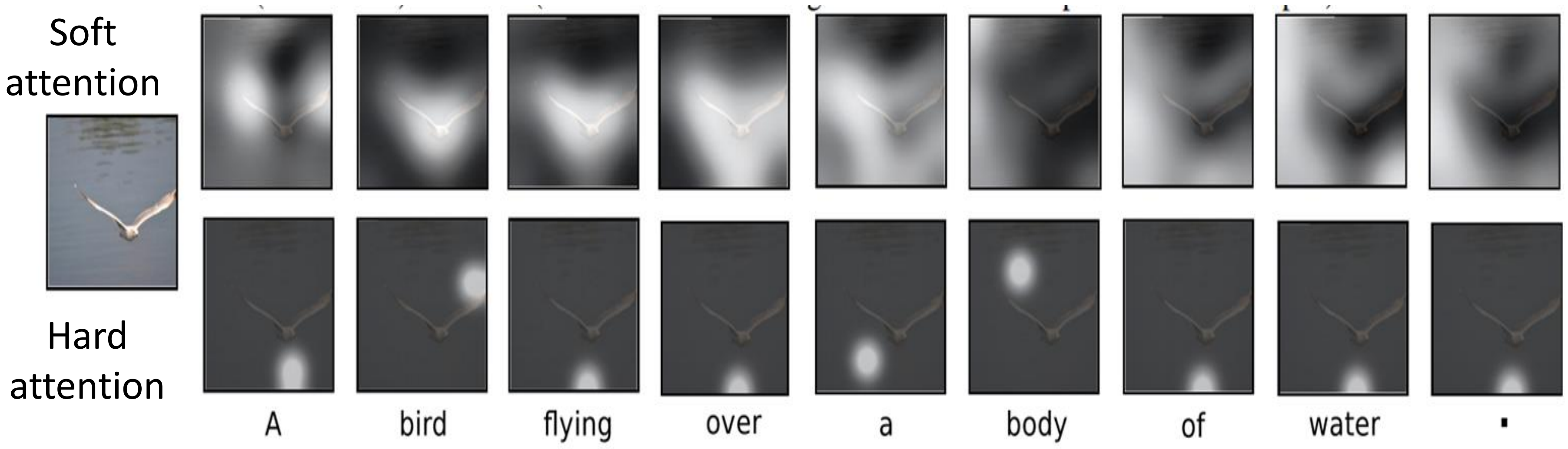
Derivative dz/dp is nice!
Train with gradient descent

Hard attention:
Sample ONE location
according to p, z = that vector

With argmax, dz/dp is zero
almost everywhere …
Can't use gradient descent;
need reinforcement learning

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Soft Attention for Captioning



Model want to attend to salient part of an image while generating its caption

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning

A woman is throwing a <u>frisbee</u> in a park.

A <u>dog</u> is standing on a hardwood floor.

A <u>stop</u> sign is on a road with a mountain in the background.

A little <u>girl</u> sitting on a bed with a teddy bear.

A group of <u>people</u> sitting on a boat in the water.

A giraffe standing in a forest with <u>trees</u> in the background.

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Visual Question Answering

Q: What endangered animal is featured on the truck?

A: A bald eagle.
A: A sparrow.
A: A humming bird.
A: A raven.

Q: Where will the driver go if turning right?

A: Onto 24 ¾ Rd.
A: Onto 25 ¾ Rd.
A: Onto 23 ¾ Rd.
A: Onto Main Street.

Q: When was the picture taken?

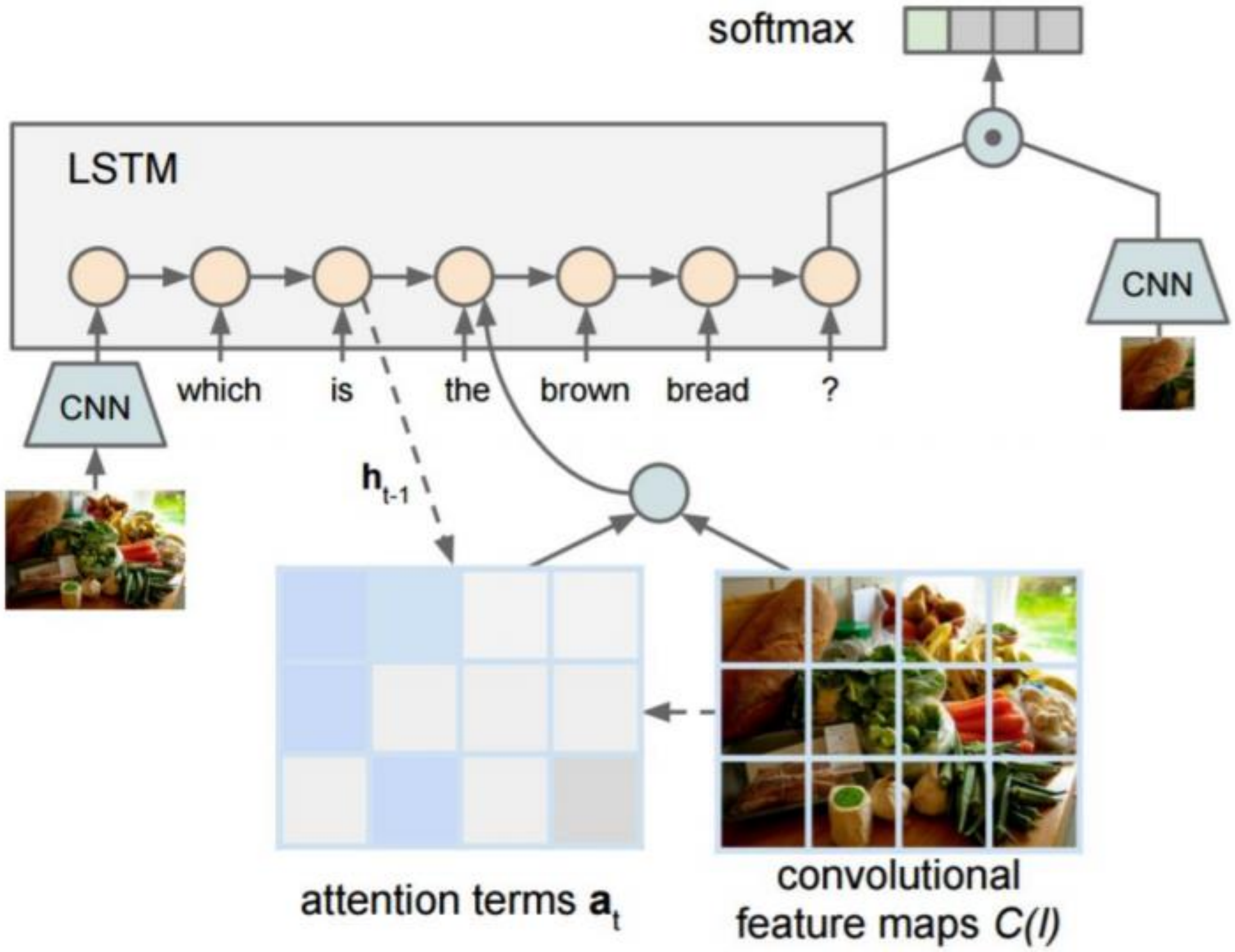
A: During a wedding.
A: During a bar mitzvah.
A: During a funeral.
A: During a Sunday church service

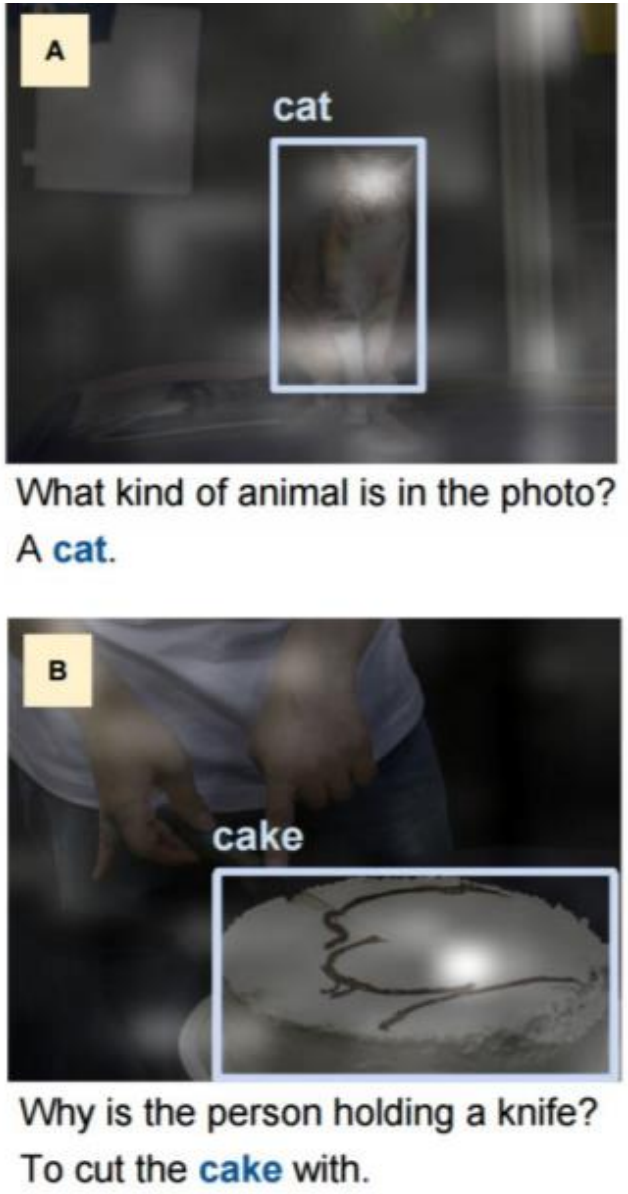
Q: Who is under the umbrella?

A: Two women.
A: A child.
A: An old man.
A: A husband and a wife.

Agrawal et al, "VQA: Visual Question Answering", ICCV 2015
Zhu et al, "Visual 7W: Grounded Question Answering in Images", CVPR 2016

Visual Question Answering: RNNs with Attention





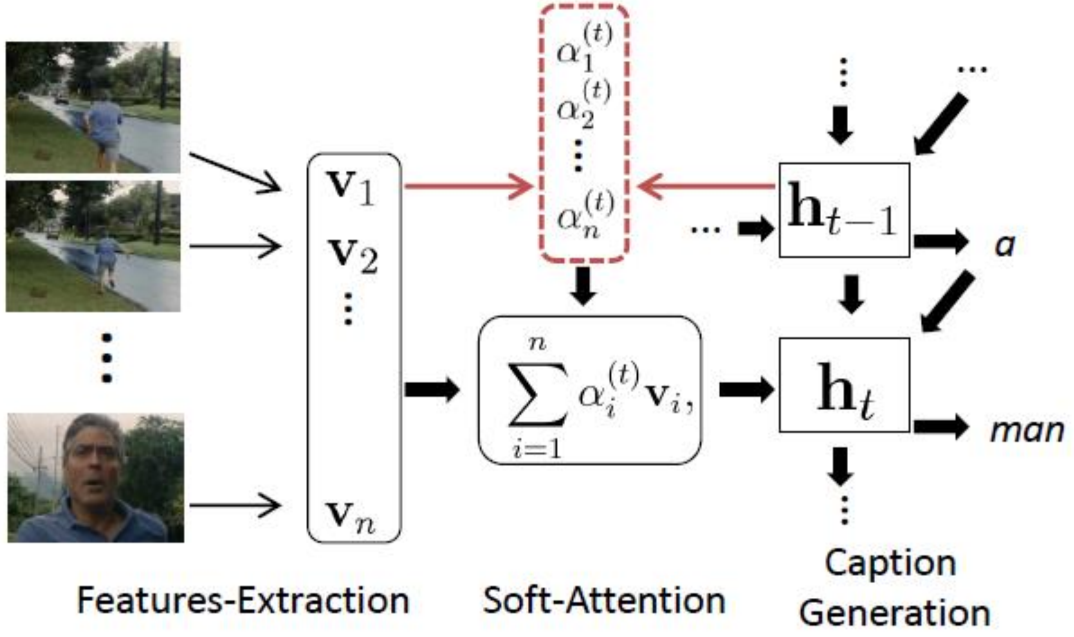
Zhu et al, "Visual 7W: Grounded Question Answering in Images", CVPR 2016

Soft Attention for Video

"Describing Videos by Exploiting Temporal Structure," Li Yao et al, arXiv 2015.

Soft Attention for Video

The attention model:



"Describing Videos by Exploiting Temporal Structure," Li Yao et al, arXiv 2015.

Soft Attention for Video

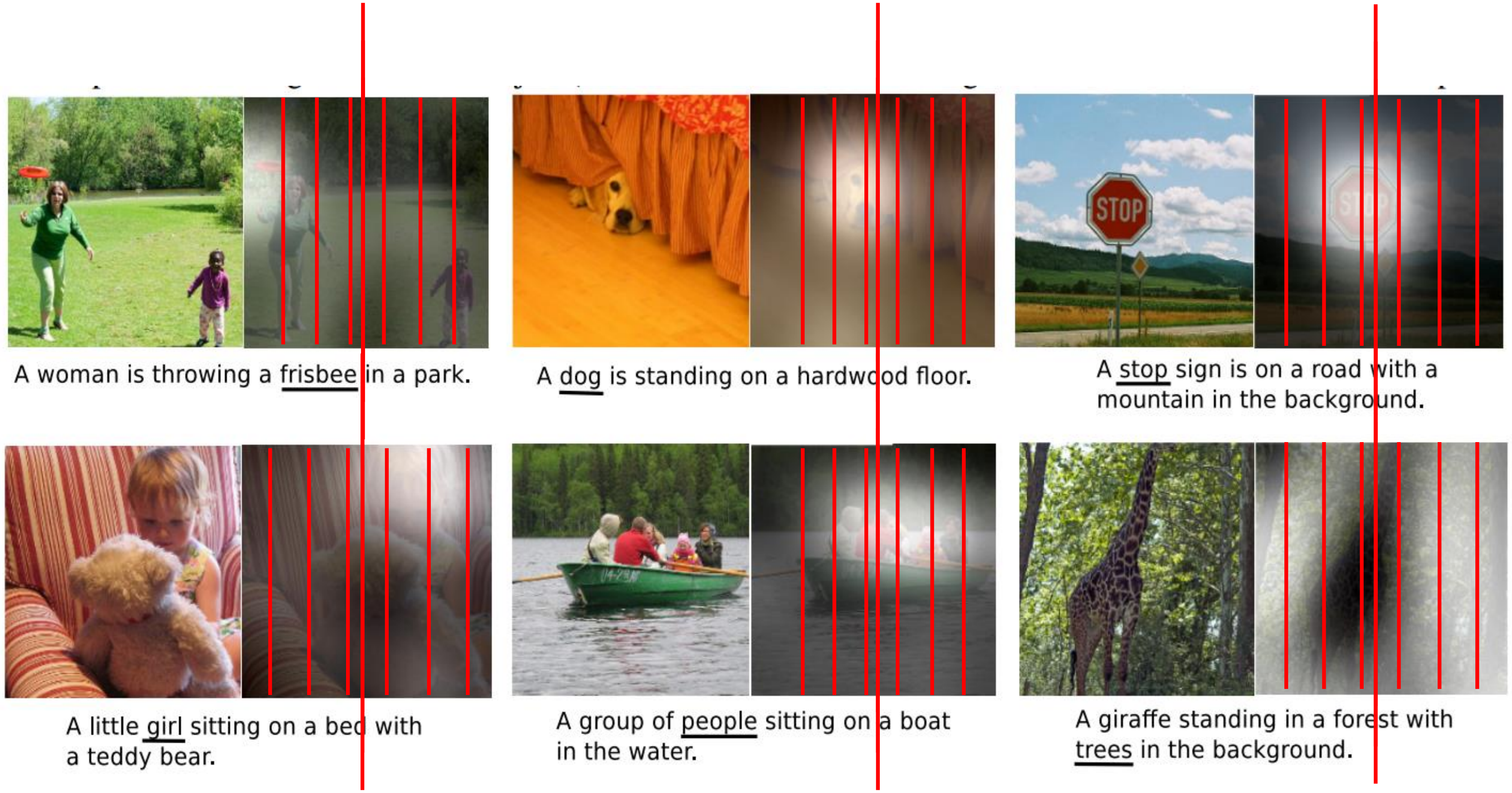
Table 1. Performance of different variants of the model on the Youtube2Text and DVS datasets.

Model	Youtube2Text				DVS			
	BLEU	METEOR	CIDEr	Perplexity	BLEU	METEOR	CIDEr	Perplexity
Enc-Dec (Basic)	0.3869	0.2868	0.4478	33.09	0.003	0.044	0.044	88.28
+ Local (3-D CNN)	0.3875	0.2832	0.5087	33.42	0.004	0.051	0.050	84.41
+ Global (Temporal Attention)	0.4028	0.2900	0.4801	27.89	0.003	0.040	0.047	66.63
+ Local + Global	**0.4192**	**0.2960**	**0.5167**	**27.55**	**0.007**	**0.057**	**0.061**	**65.44**
Venugopalan *et al.* [41]	0.3119	0.2687	-	-	-	-	-	-
+ Extra Data (Flickr30k, COCO)	0.3329	0.2907	-	-	-	-	-	-
Thomason *et al.* [37]	0.1368	0.2390	-	-	-	-	-	-

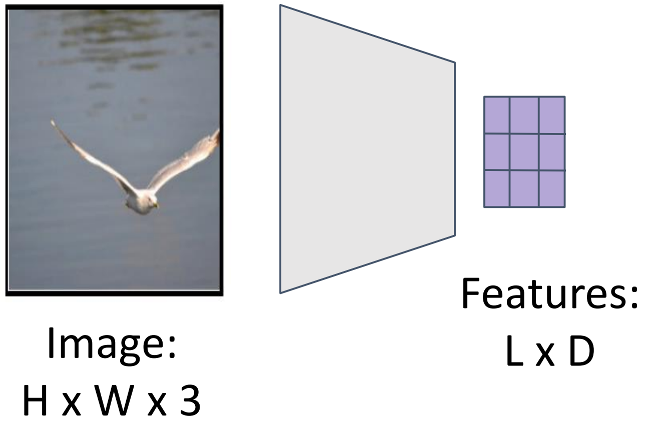
“Describing Videos by Exploiting Temporal Structure,” Li Yao et al, arXiv 2015.

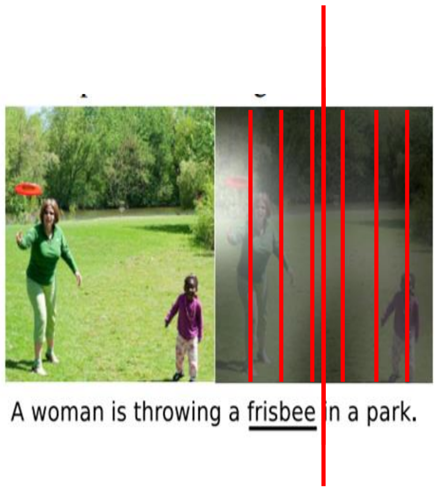
Soft Attention for Captioning

Attention constrained to fixed grid! We’ll come back to this ….



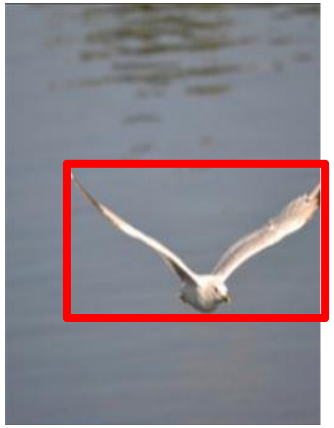
Attending to arbitrary regions?



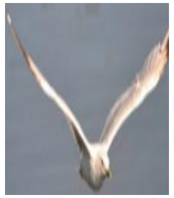


Attention mechanism from Show, Attend, and Tell only lets us softly attend to fixed grid positions … can we do better?

Spatial Transformer Networks



Input
image:
H x W x 3

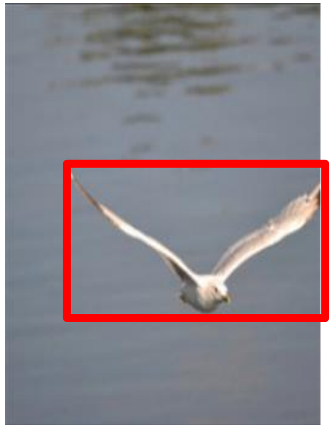


Cropped and
rescaled image:
X x Y x 3

Box
Coordinates:
(xc, yc, w, h)

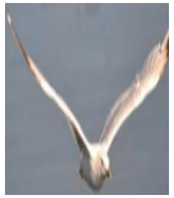
Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks



Input image: H x W x 3

Can we make this function differentiable?

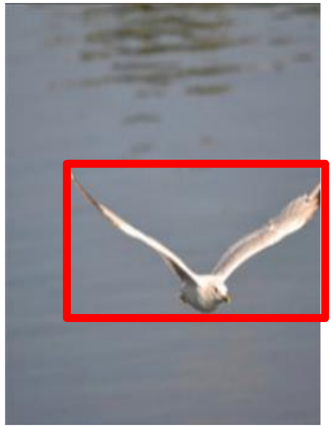


Cropped and rescaled image: X x Y x 3

Box Coordinates: (xc, yc, w, h)

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

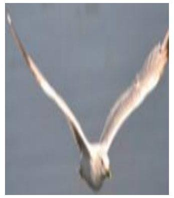
Spatial Transformer Networks



Input image: H x W x 3

Box Coordinates: (xc, yc, w, h)

Can we make this function differentiable?



Cropped and rescaled image: X x Y x 3

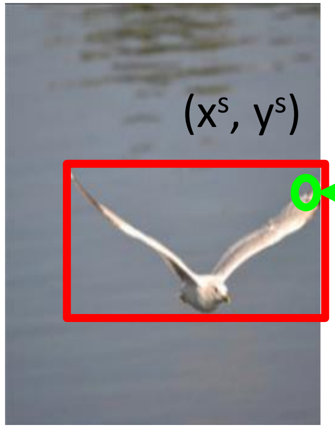
Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

Jaderberg et al, "Spatial Transformer Networks", NIPS 2015

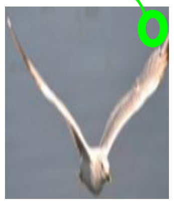
Spatial Transformer Networks

(x^s, y^s)



Can we make this function differentiable?

(x^t, y^t)



Input image:
H x W x 3

Cropped and rescaled image:
X x Y x 3

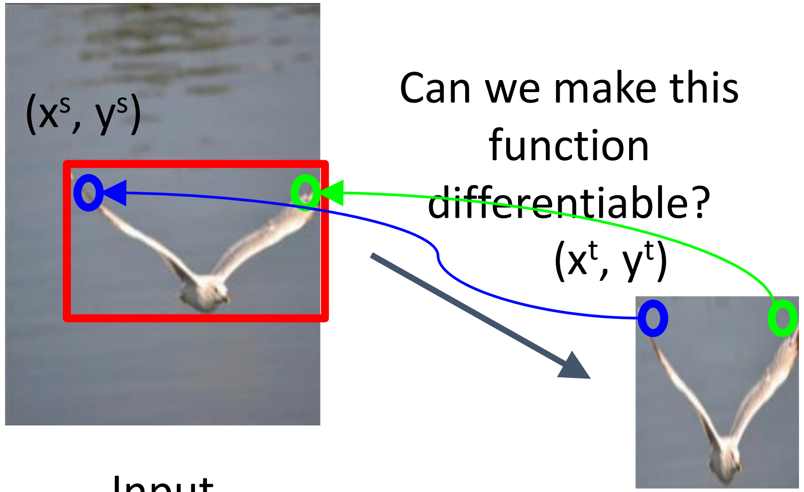
Box Coordinates:
(xc, yc, w, h)

Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

Jaderberg et al, "Spatial Transformer Networks", NIPS 2015

Spatial Transformer Networks



Input image:
H x W x 3

Cropped and rescaled image:
X x Y x 3

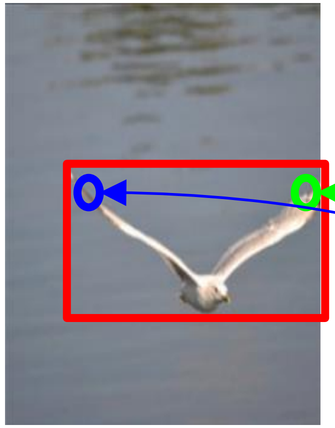
Box Coordinates:
(xc, yc, w, h)

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

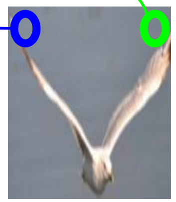
Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

Spatial Transformer Networks



Can we make this function differentiable?



Input image:
H x W x 3

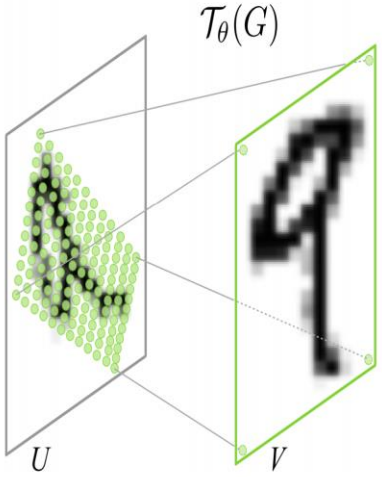
Cropped and rescaled image:
X x Y x 3

Box Coordinates:
(xc, yc, w, h)

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

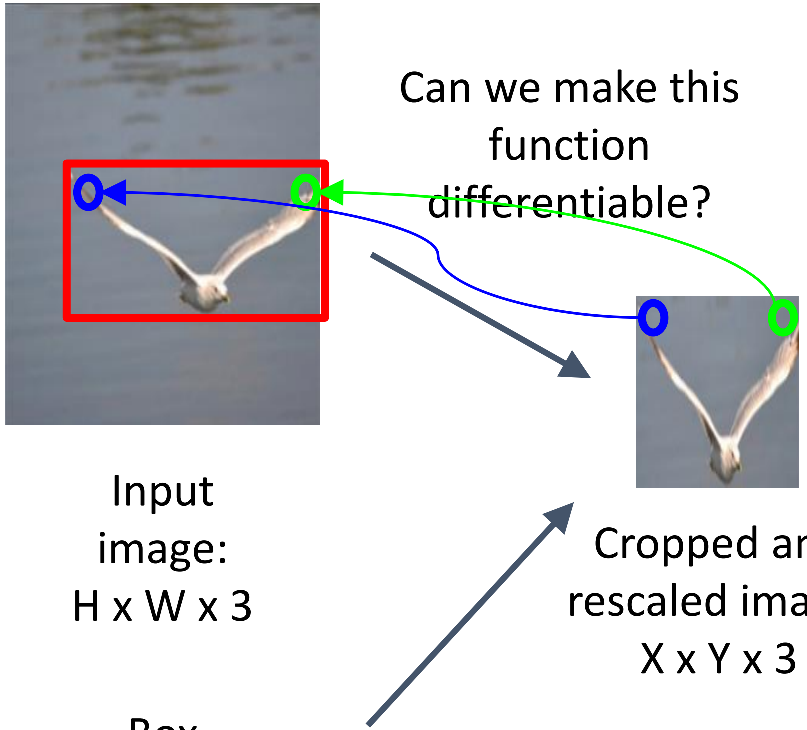
Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$



Repeat for all pixels in *output* to get a **sampling grid**

Spatial Transformer Networks



Can we make this function differentiable?

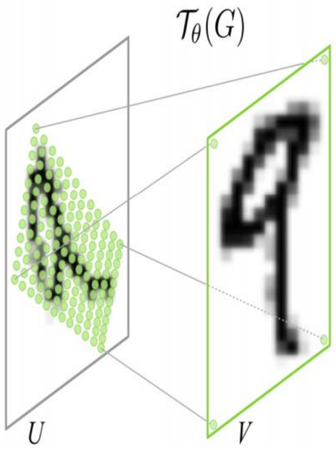
Input image: H x W x 3

Box Coordinates: (xc, yc, w, h)

Cropped and rescaled image: X x Y x 3

Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

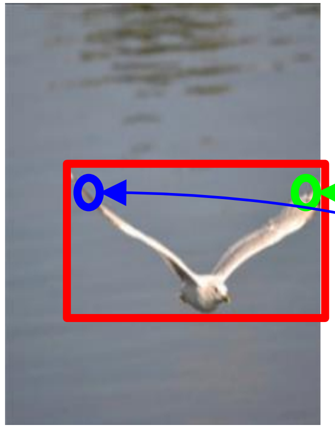


Repeat for all pixels in *output* to get a **sampling grid**

Then use **bilinear interpolation** to compute output

Jaderberg et al, "Spatial Transformer Networks", NIPS 2015

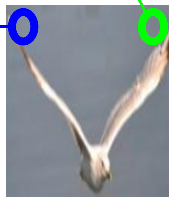
Spatial Transformer Networks



Input
image:
H x W x 3

Box
Coordinates:
(xc, yc, w, h)

Can we make this function differentiable?

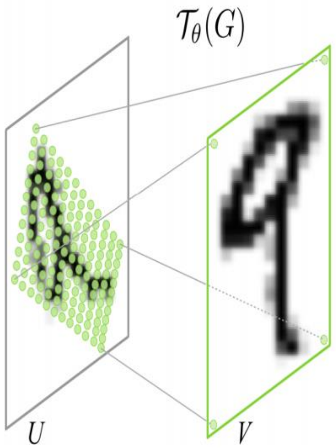


Cropped and
rescaled image:
X x Y x 3

Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

Network attends to input by predicting θ

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$



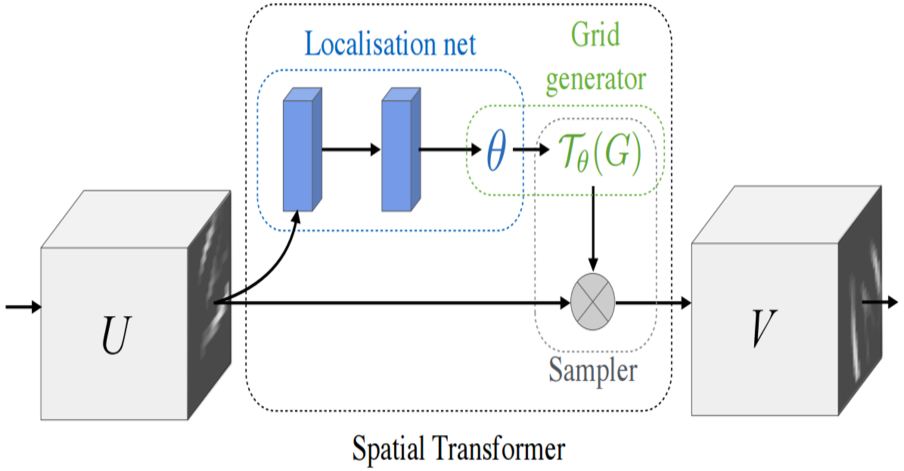
Repeat for all pixels in *output* to get a **sampling grid**

Then use **bilinear interpolation** to compute output

Jaderberg et al, "Spatial Transformer Networks", NIPS 2015

Spatial Transformer Networks

Input: Full image



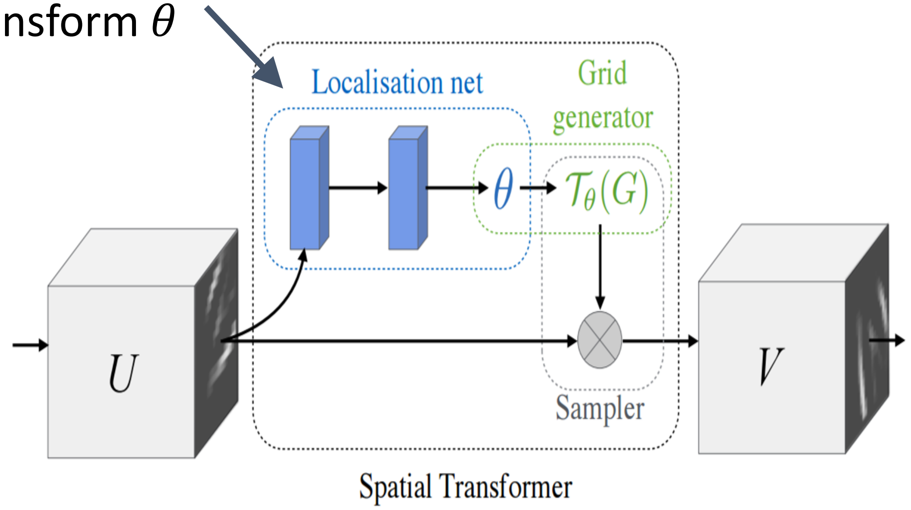
Output: Region of interest from input

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks

A small
Localization network
predicts transform θ

Input:
Full image



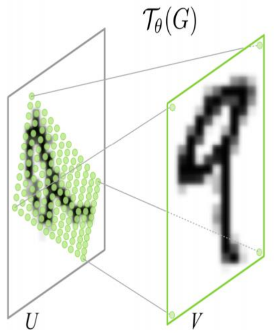
Output: Region of interest from input

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks

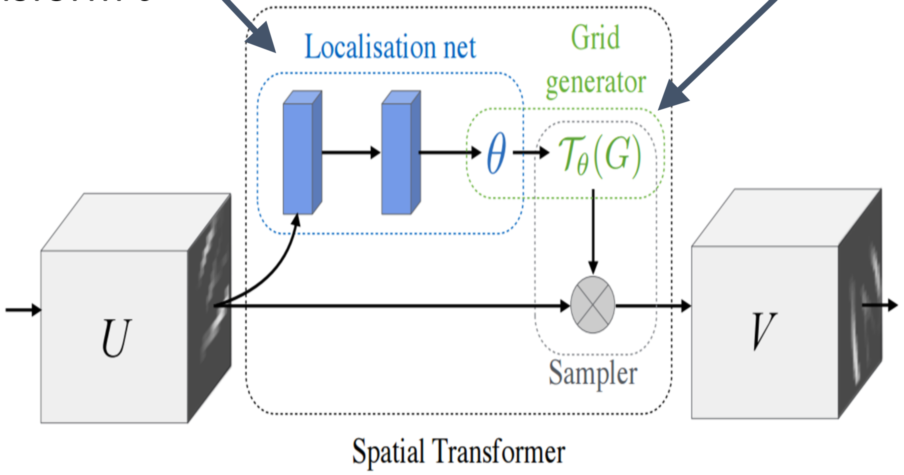
Grid generator uses θ to compute sampling grid

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$



A small **Localization network** predicts transform θ

Input: Full image



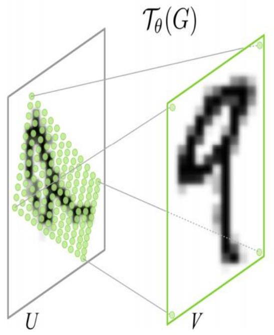
Output: Region of interest from input

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks

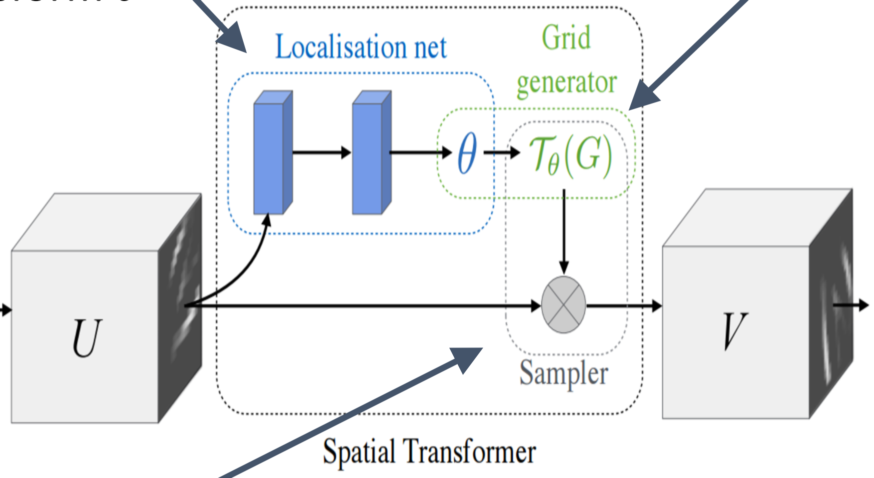
Grid generator uses θ to compute sampling grid

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$



A small **Localization network** predicts transform θ

Input: Full image



Output: Region of interest from input

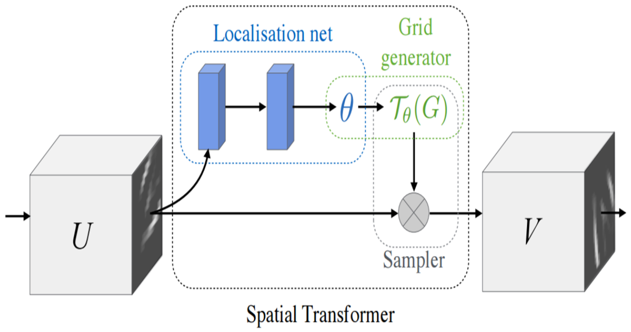
Sampler uses bilinear interpolation to produce output

$$V_i^c = \sum_n^H \sum_m^W U_{nm}^c \max(0, 1 - |x_i^s - m|) \max(0, 1 - |y_i^s - n|)$$

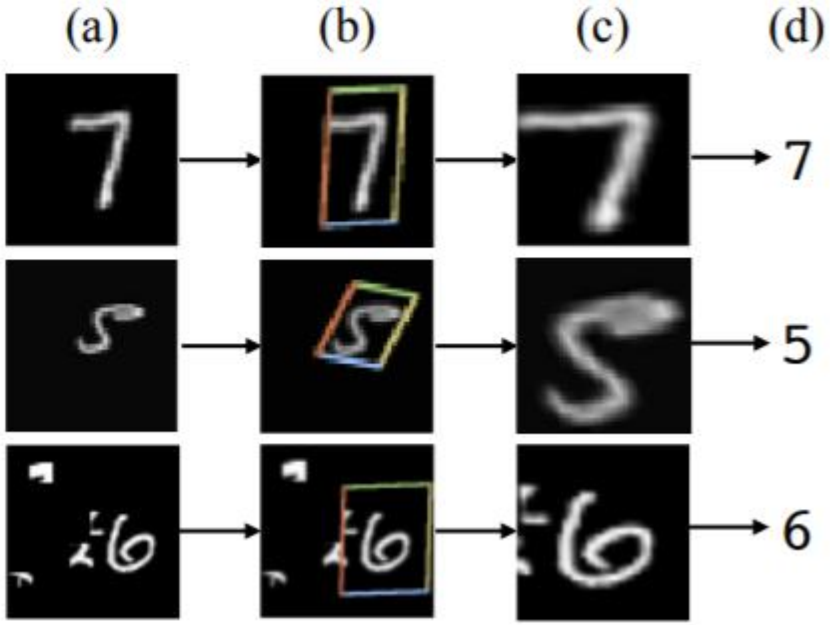
Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks

Differentiable “attention / transformation” module



Insert spatial transformers into a classification network and it learns to attend and transform the input



Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Attention Takeaways

Performance:

- Attention models can ***improve accuracy*** and ***reduce computation*** at the same time.

Complexity:

- There are many design choices.
- Those choices have a big effect on performance.
- Ensembling has unusually large benefits.
- Simplify where possible!

Attention Takeaways

Explainability:

- Attention models encode explanations.
- Both locus and trajectory help understand what’s going on.

Hard vs. Soft:

- Soft models are easier to train, hard models require reinforcement learning.
- They can be combined, as in Luong et al.