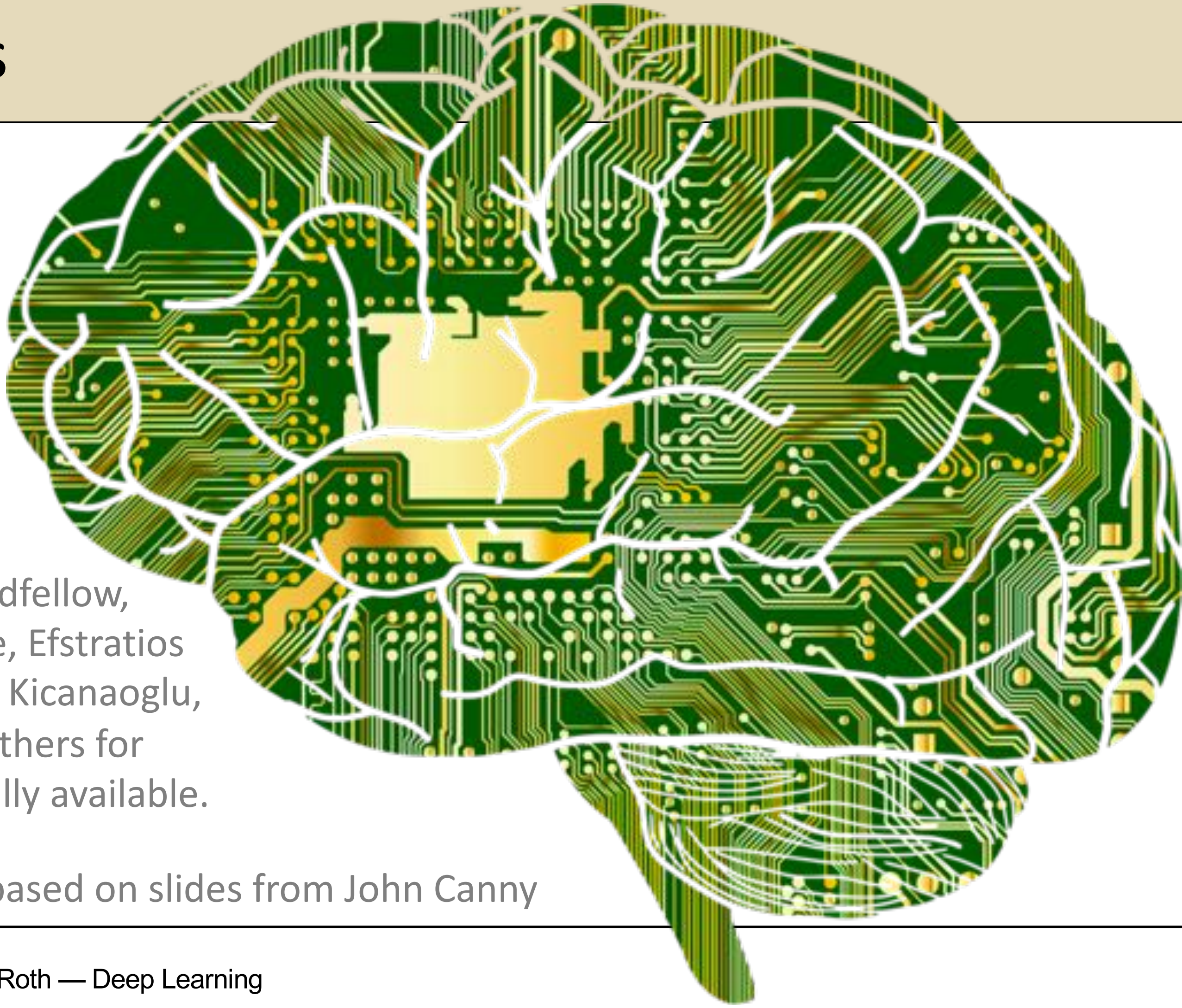
Deep Learning

Architectures and Methods:

Attention Models



Thanks to John Canny, Ian Goodfellow, Yoshua Bengio, Aaron Courville, Efstratios Gavves, Kirill Gavrilyuk, Berkay Kicanaoglu, and Patrick Putzky and many others for making their materials publically available.

The present slides are mainly based on slides from John Canny

Early attention models

Larochelle and Hinton, 2010, “Learning to combine foveal glimpses with a third-order Boltzmann machine”

Misha Denil et al, 2011, “Learning where to Attend with Deep Architectures for Image Tracking”

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.
- Can’t use gradient descent.
- Need **reinforcement learning.**

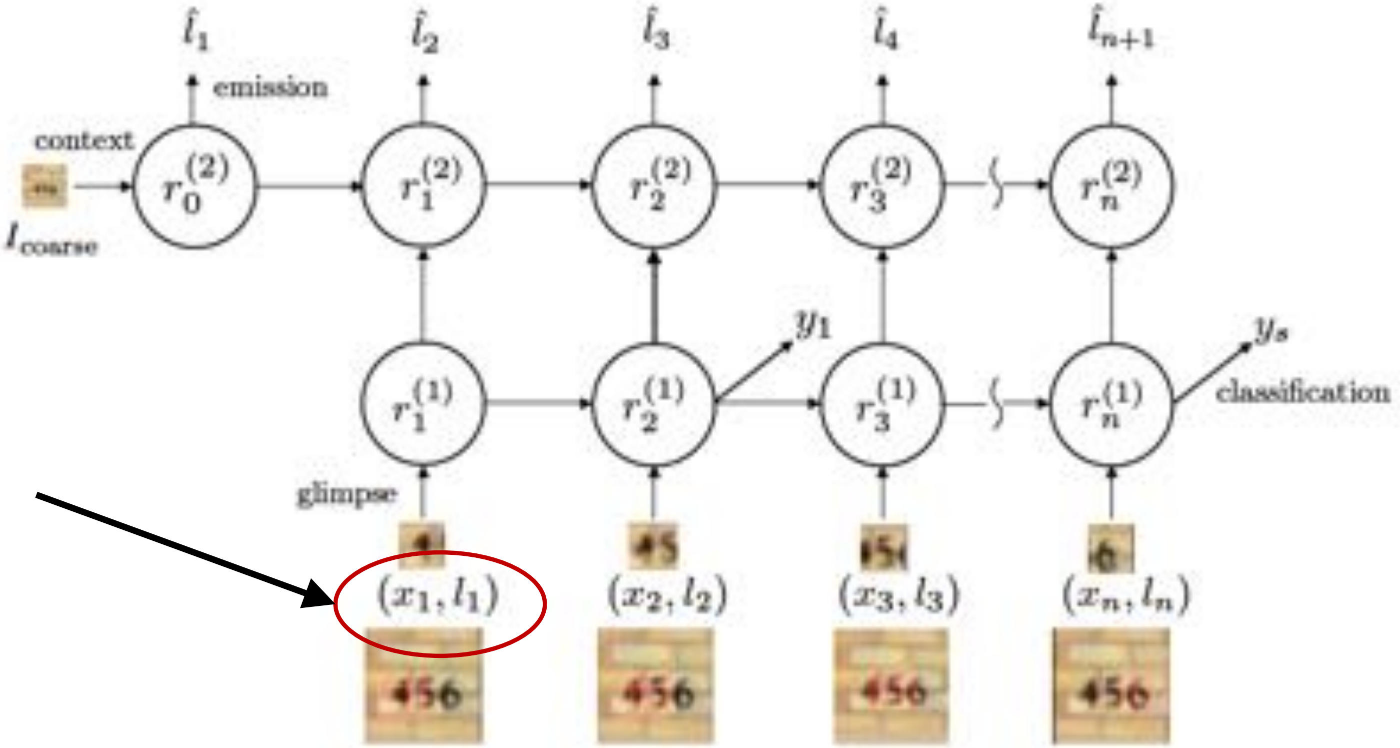
Soft attention:

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

Attention for Recognition (Ba et al 2014)

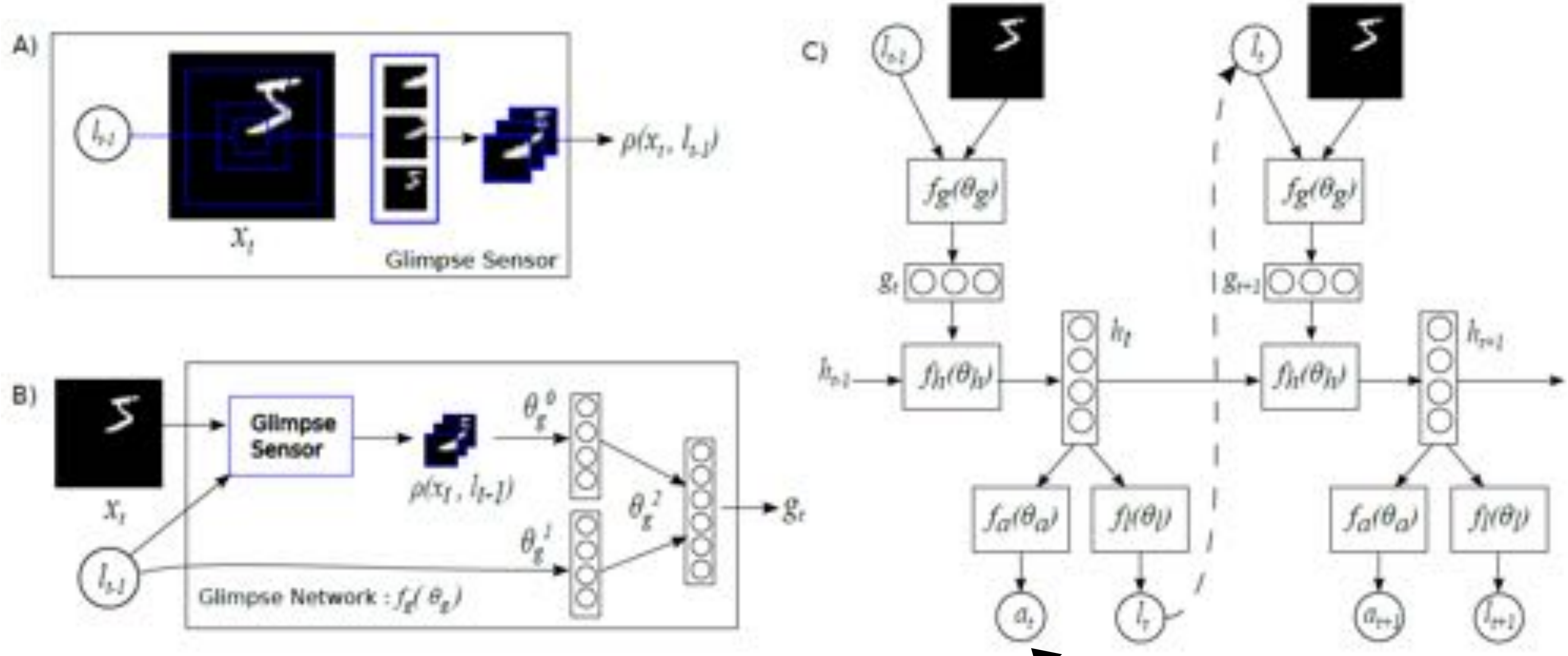
- RNN-based model.
- Hard attention.
- Required reinforcement learning.

l_i = location
x_i = image



Attention for Recognition (Mnih et al 2014)

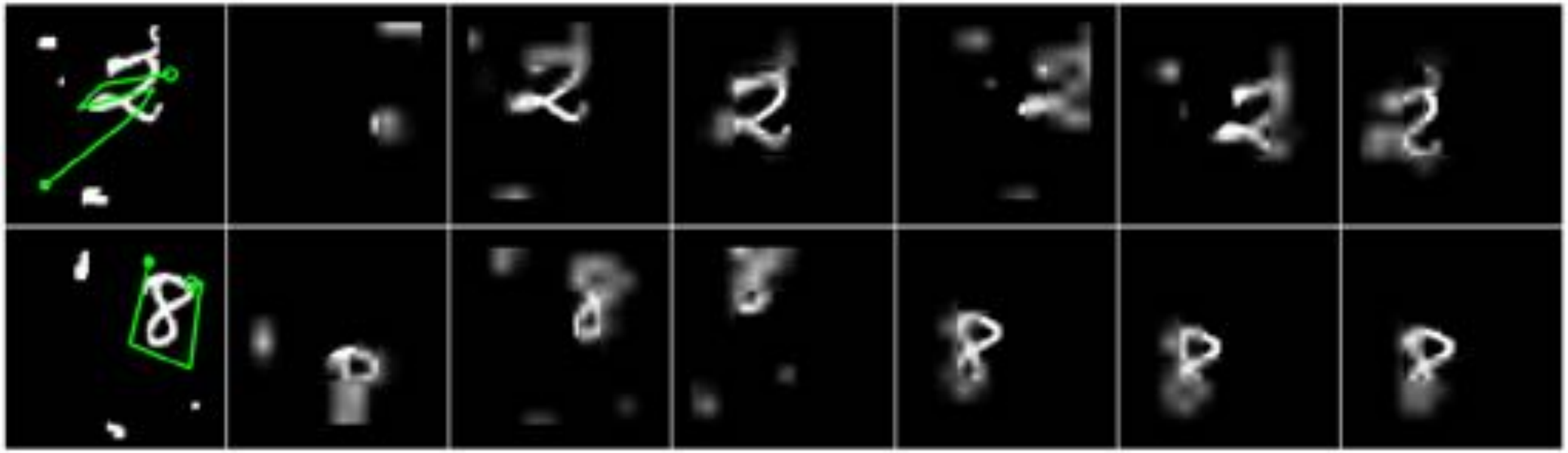
- Glimpses are retinal (graded resolution) images



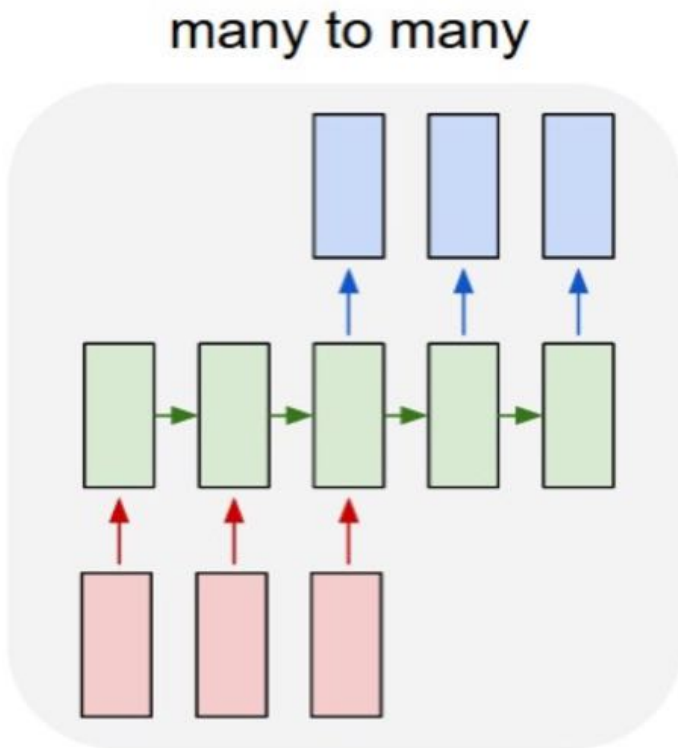
l_i = location
a_i = action (classification)

Attention for Recognition (Mnih et al 2014)

- Glimpse trace on some digit images:
- Green line shows trajectory, other images are the glimpses themselves.



Soft Attention for Translation

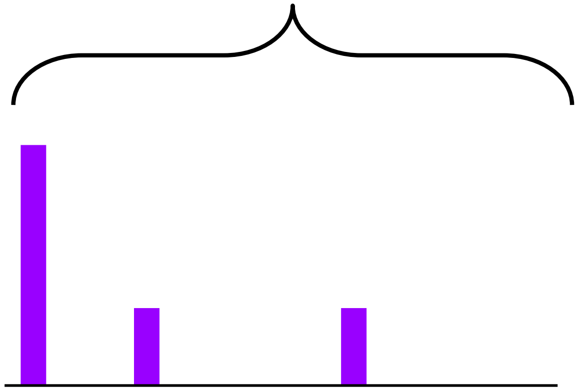


“I love coffee” -> “Me gusta el café”

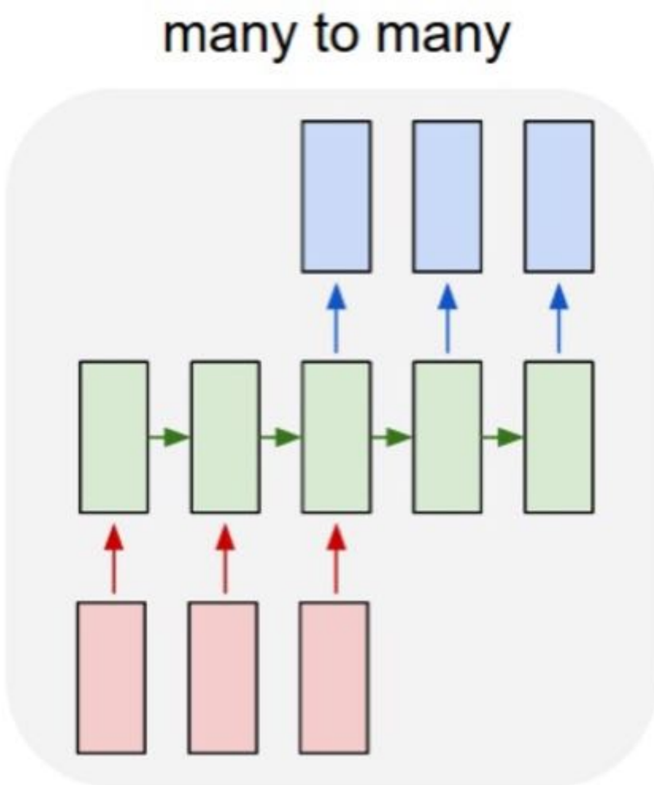
Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation

Distribution over
input words



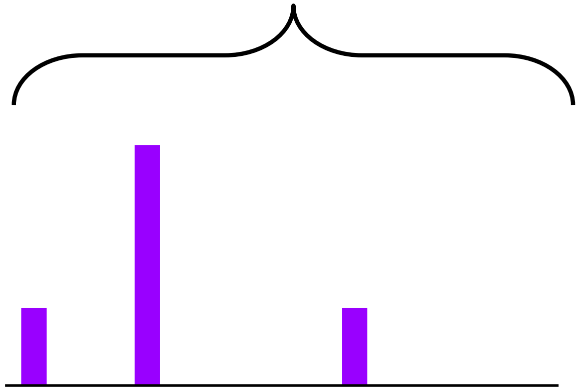
“I love coffee” -> “Me gusta el café”



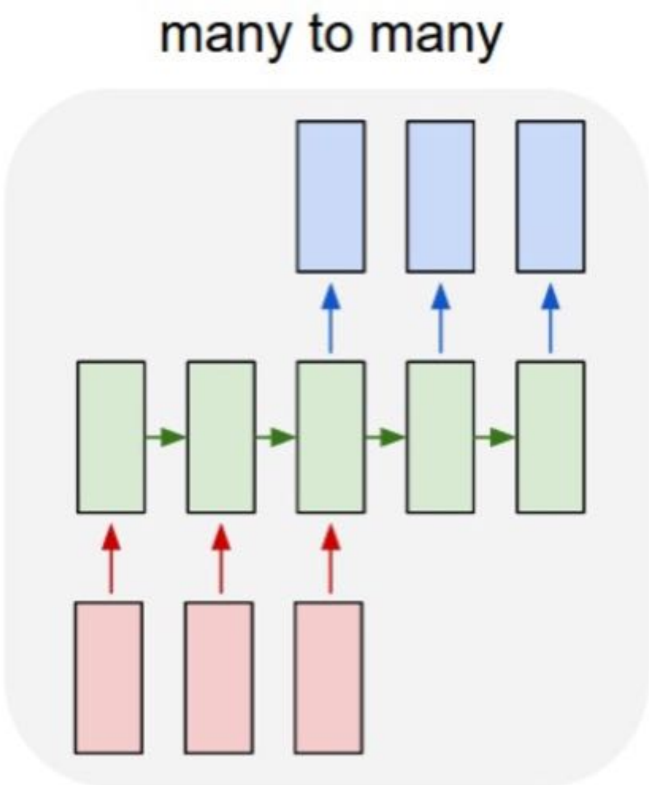
Bahdanau et al, “Neural Machine Translation by Jointly
Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation

Distribution over
input words



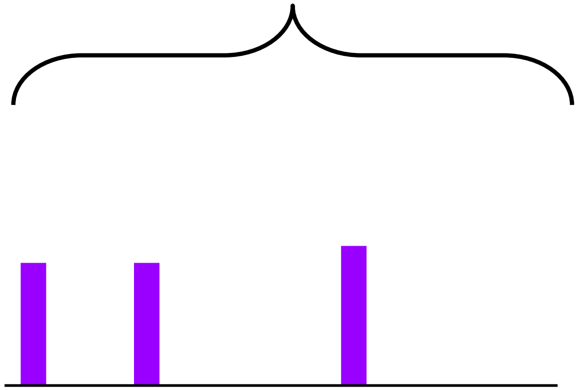
“I love coffee” -> “Me gusta el café”



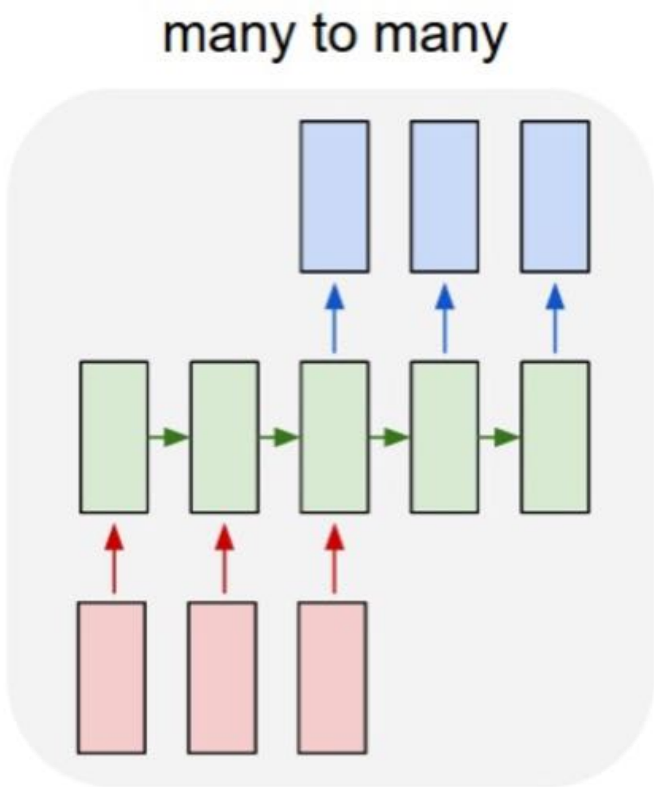
Bahdanau et al, “Neural Machine Translation by Jointly
Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation

Distribution over
input words

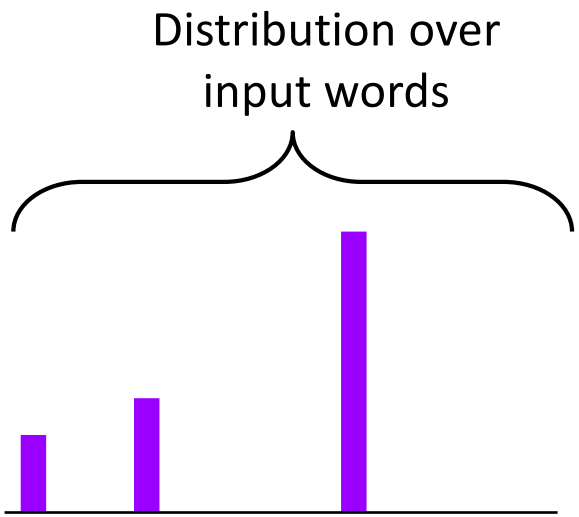


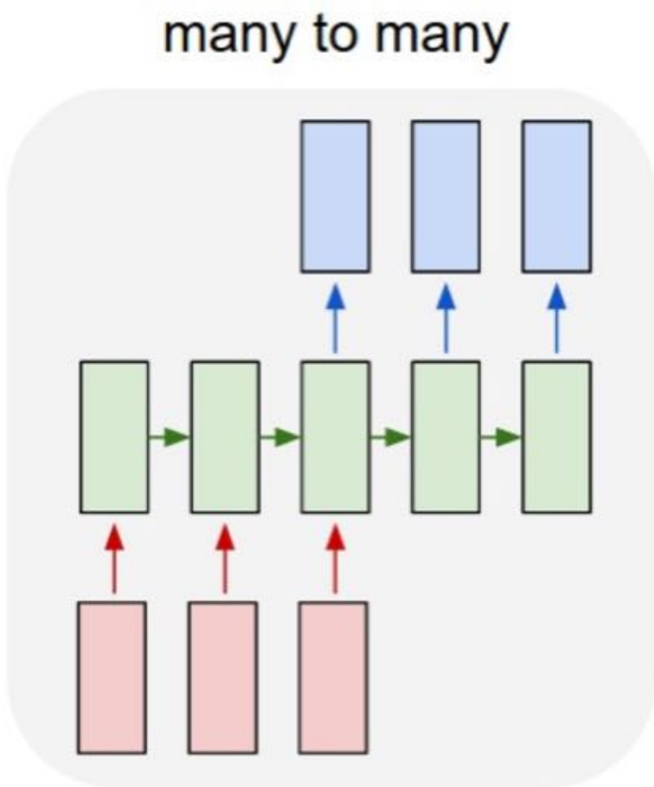
“I love coffee” -> “Me gusta el café”



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

Soft Attention for Translation

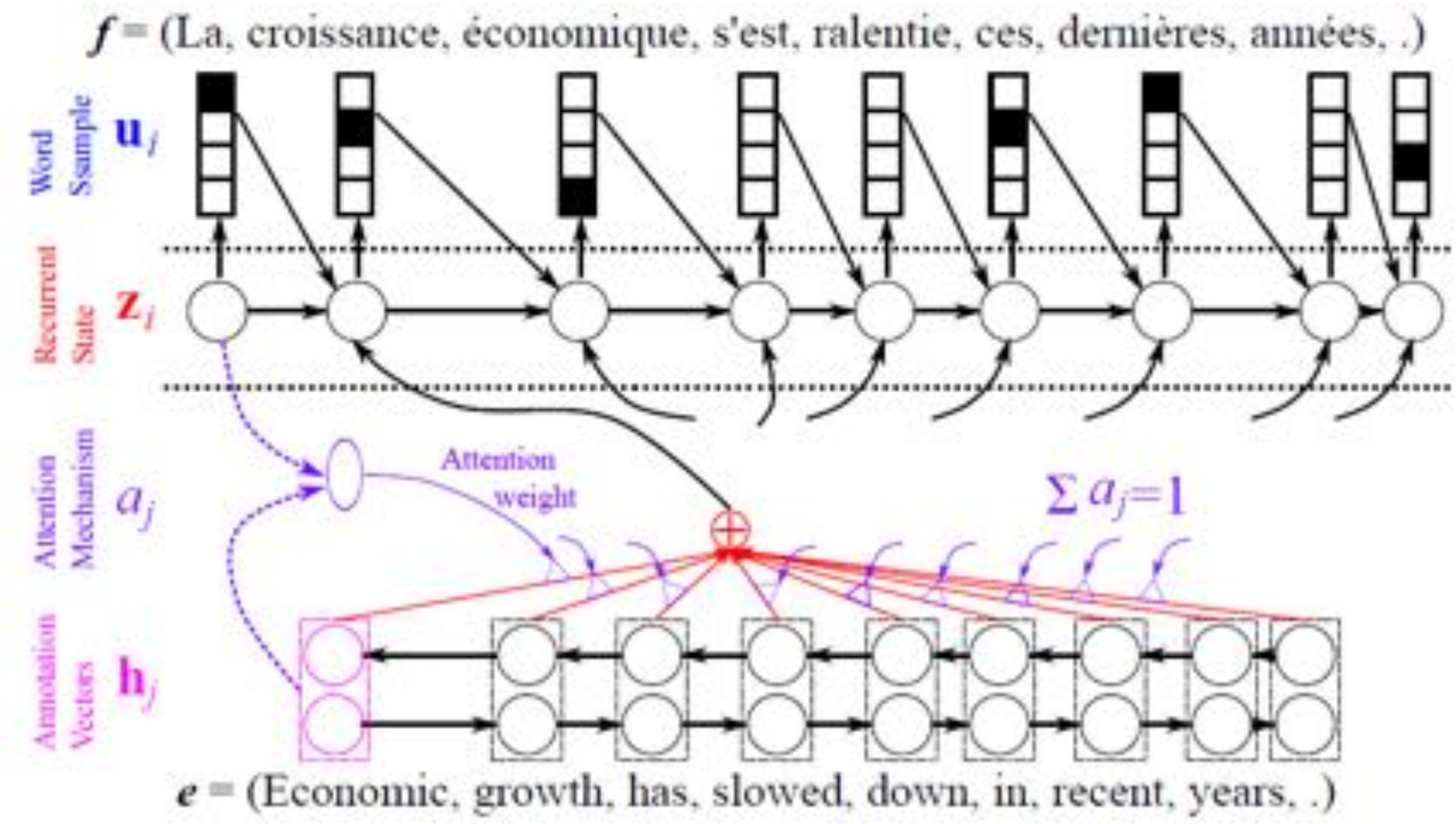




"I love coffee" -> "Me gusta el café"

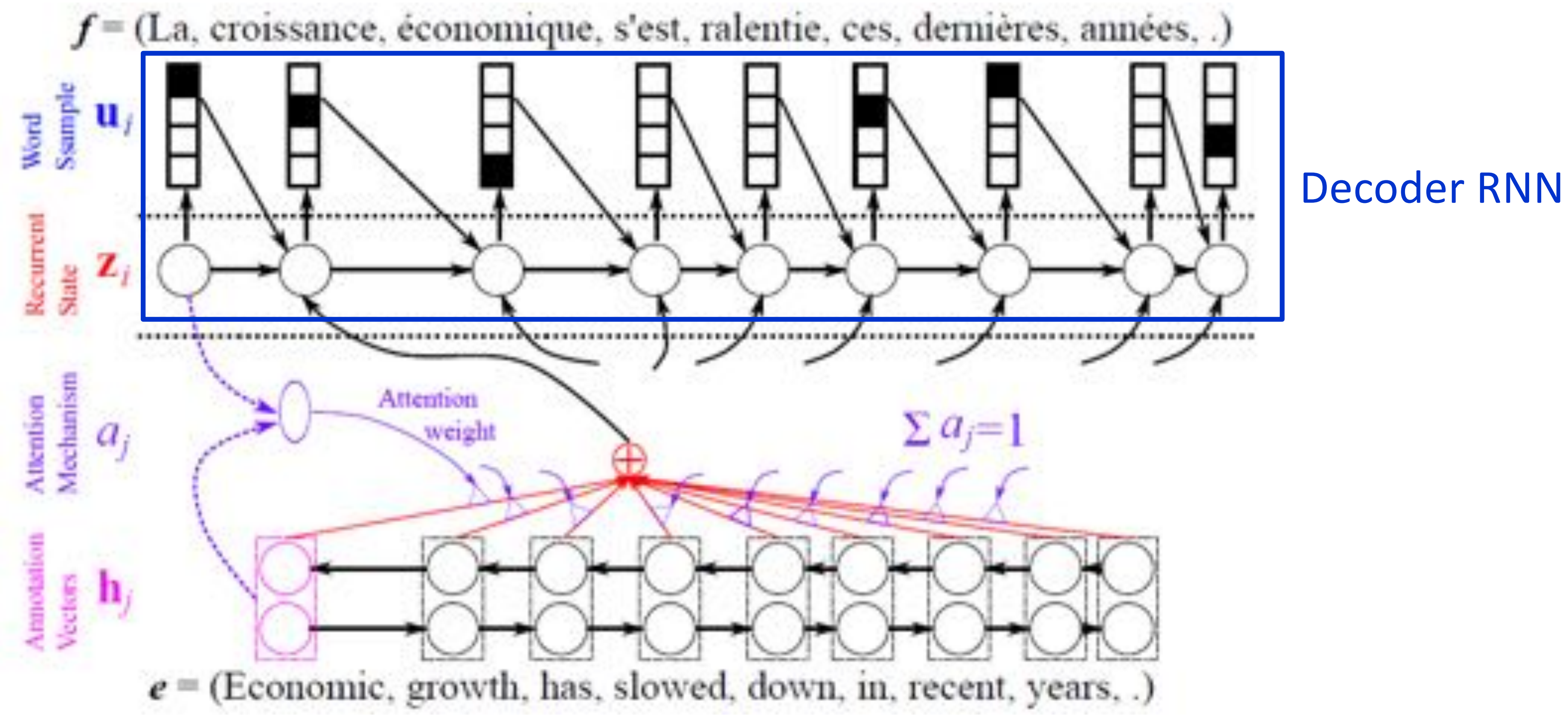
Bahdanau et al, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR 2015

Soft Attention for Translation



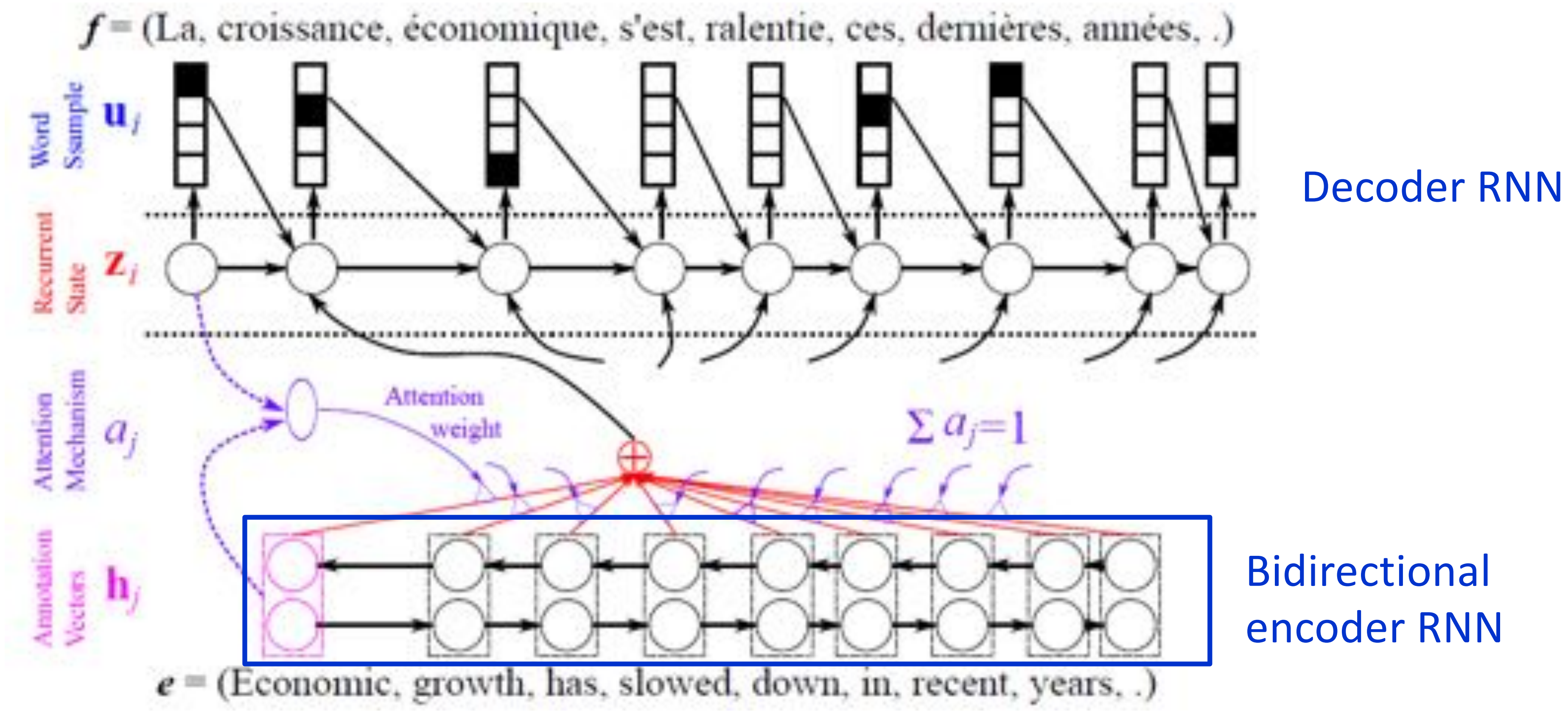
From Y. Bengio CVPR 2015 Tutorial

Soft Attention for Translation



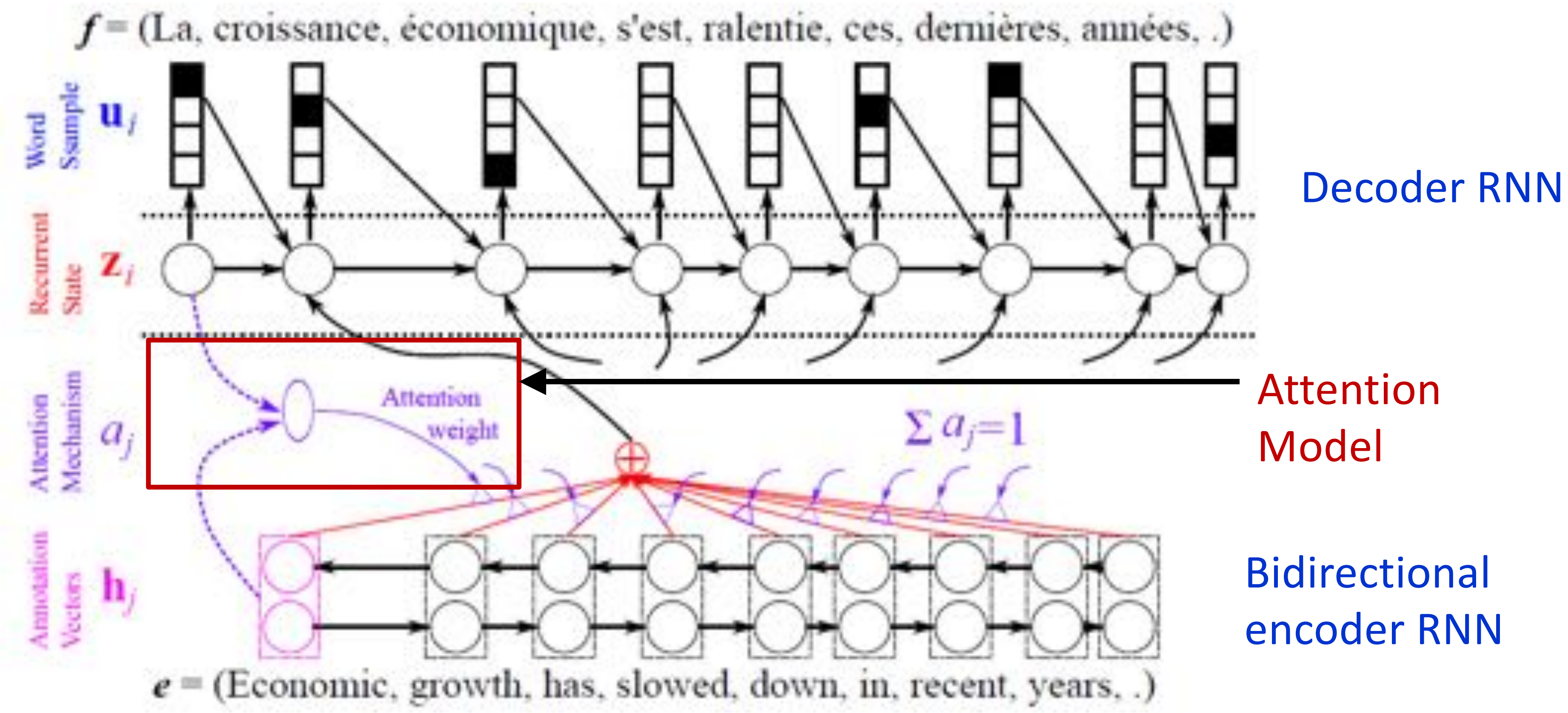
From Y. Bengio CVPR 2015 Tutorial

Soft Attention for Translation



From Y. Bengio CVPR 2015 Tutorial

Soft Attention for Translation



From Y. Bengio CVPR 2015 Tutorial

Soft Attention for Translation

Context vector (input to decoder):

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

Mixture weights:

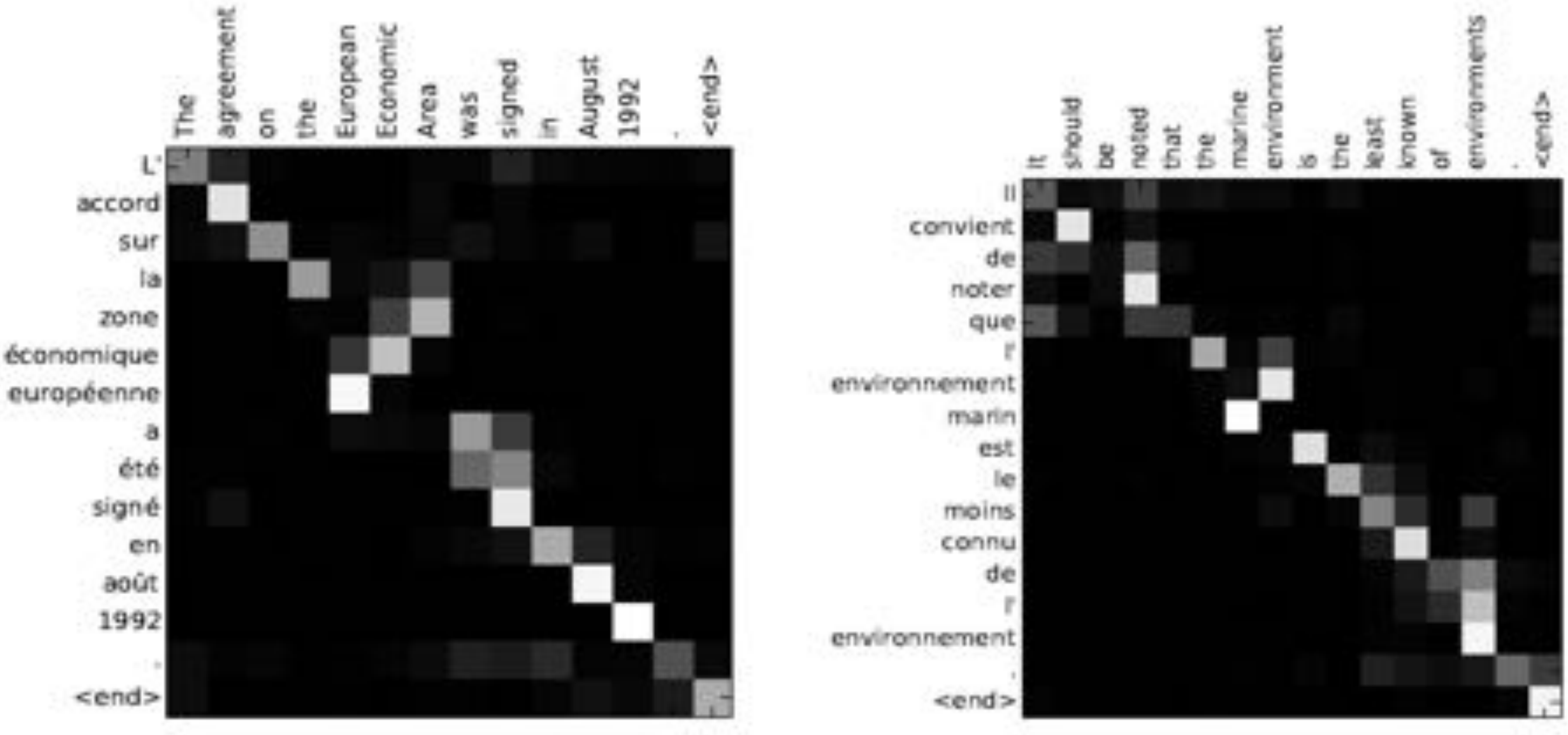
$$\alpha_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{k=1}^{T_x} \exp\left(e_{ik}\right)}$$

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

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

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

Soft Attention for Translation



(a) (b)

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

Soft Attention for Translation

Reached State of the art in one year:

(a) **English→French (WMT-14)**

	NMT(A)	Google	P-SMT
NMT	32.68	30.6*	
+Cand	33.28	–	**37.03•**
+UNK	33.99	32.7°	
+Ens	36.71	**36.9°**	

(b) **English→German (WMT-15)**

Model	Note
24.8	Neural MT
24.0	U.Edinburgh, Syntactic SMT
23.6	LIMSI/KIT
22.8	U.Edinburgh, Phrase SMT
22.7	KIT, Phrase SMT

(c) **English→Czech (WMT-15)**

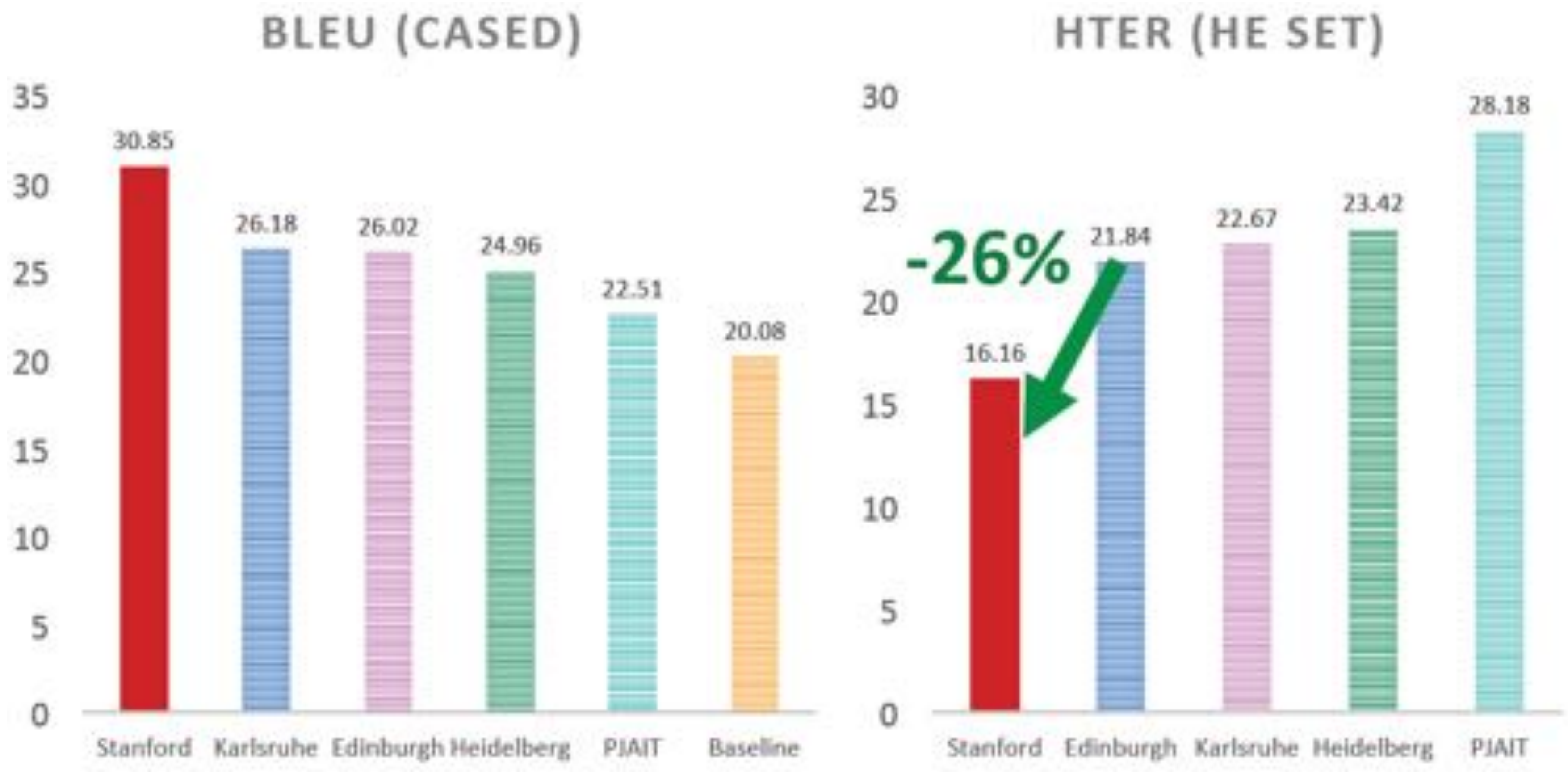
Model	Note
18.3	Neural MT
18.2	JHU, SMT+LM+OSM+Sparse
17.6	CU, Phrase SMT
17.4	U.Edinburgh, Phrase SMT
16.1	U.Edinburgh, Syntactic SMT

Yoshua Bengio, NIPS RAM workshop 2015

Soft Attention for Translation

Luong, Pham and Manning's Translation System (2015):

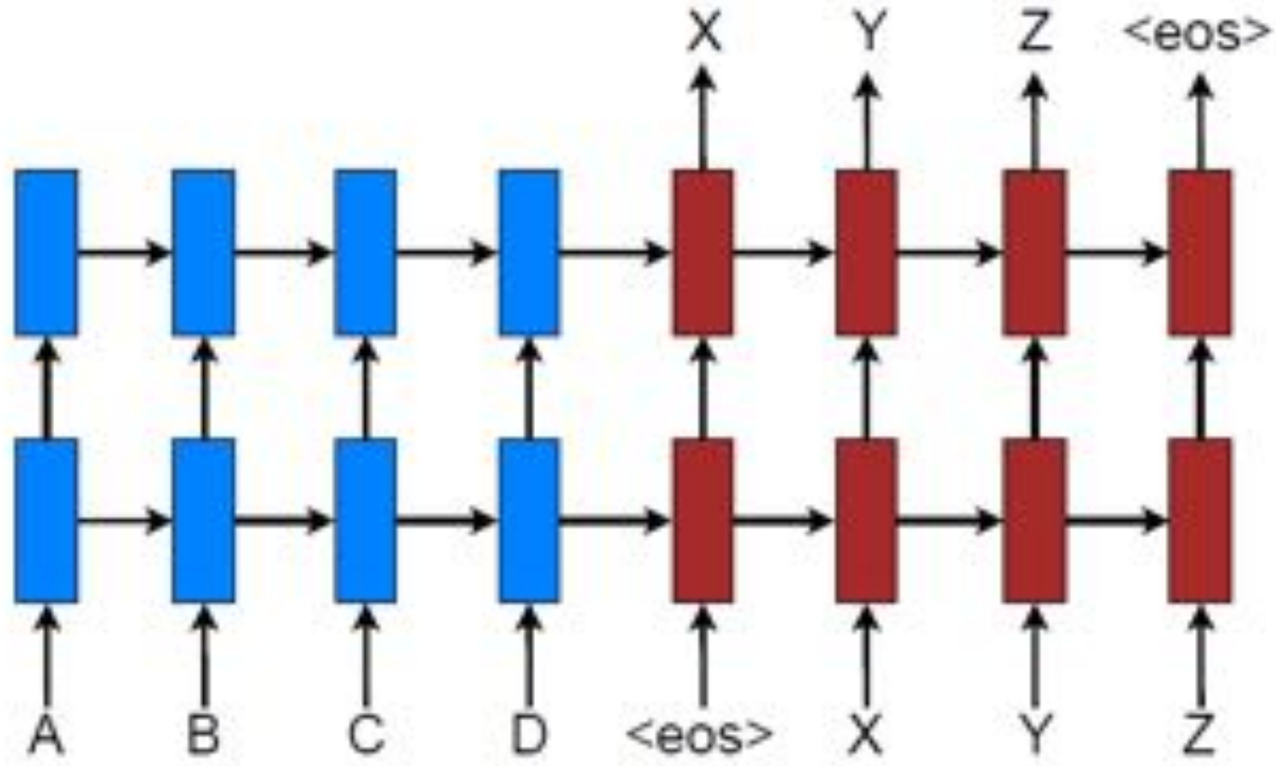
Translation Error Rate vs Human



Luong and Manning IWSLT 2015

Luong, Pham and Manning 2015

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

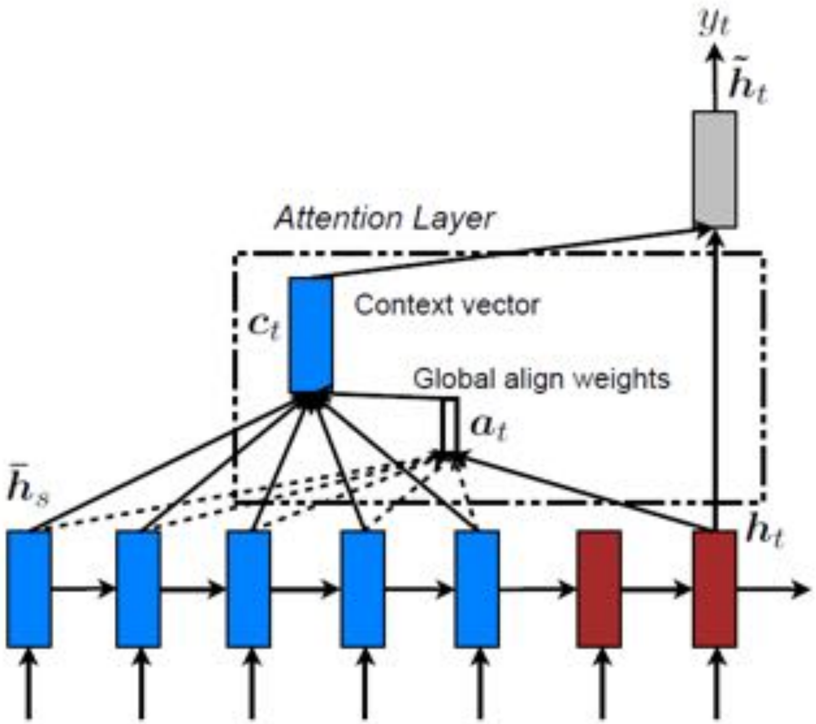


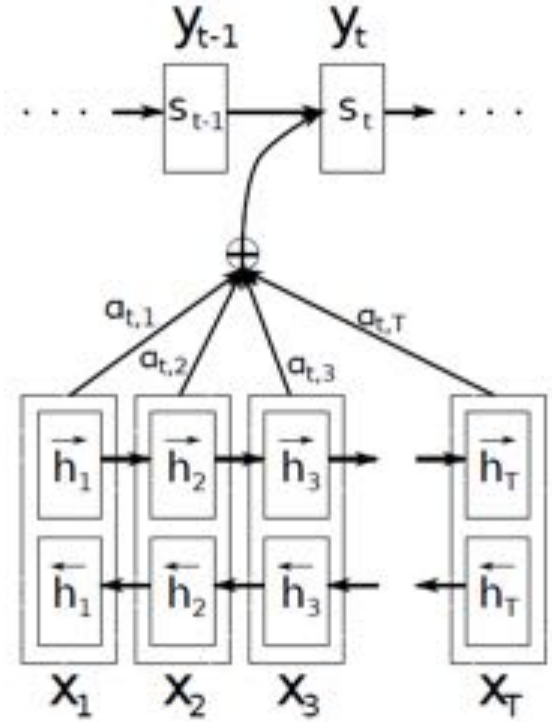
Effective Approaches to Attention-based Neural Machine Translation
Minh-Thang Luong, Hieu Pham, Christopher D. Manning, EMNLP 15

Global Attention Model

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

Different word matching functions were used

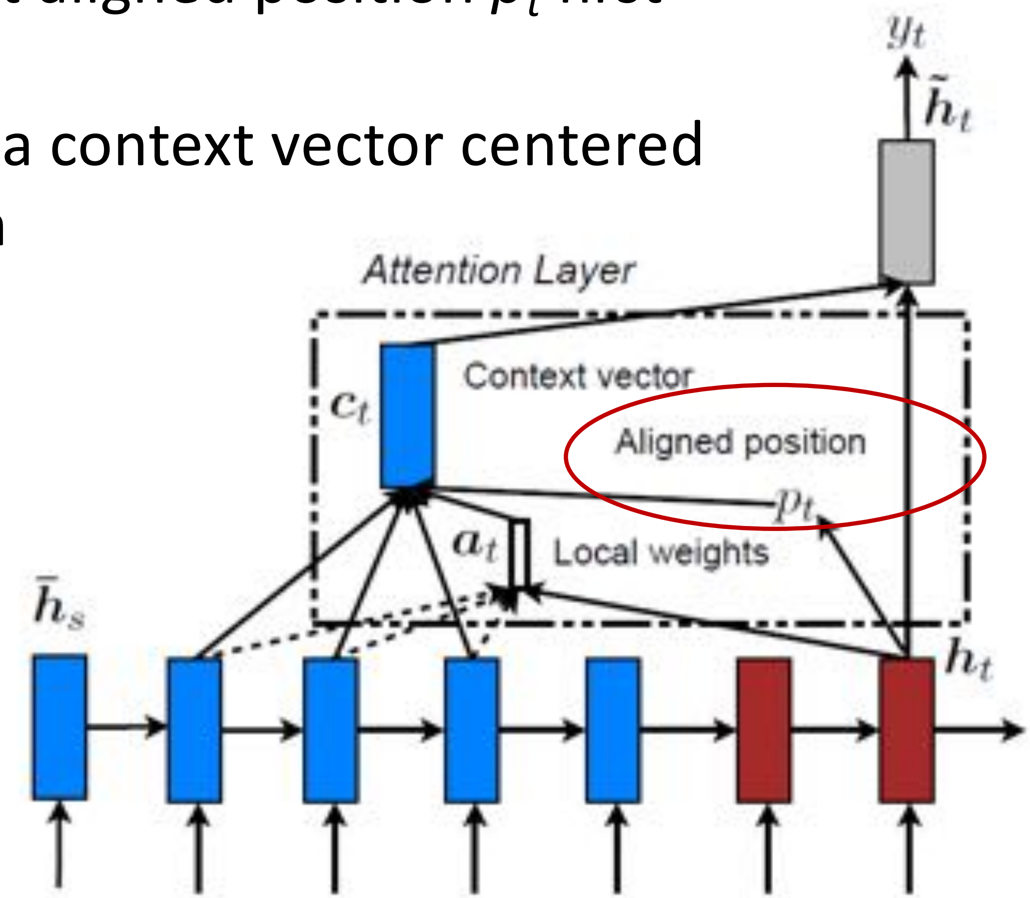




Effective Approaches to Attention-based Neural Machine Translation
Minh-Thang Luong Hieu Pham Christopher D. Manning, EMNLP 15

Local Attention Model

- Compute a best aligned position p_t first
- Then compute a context vector centered at that position



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

Results

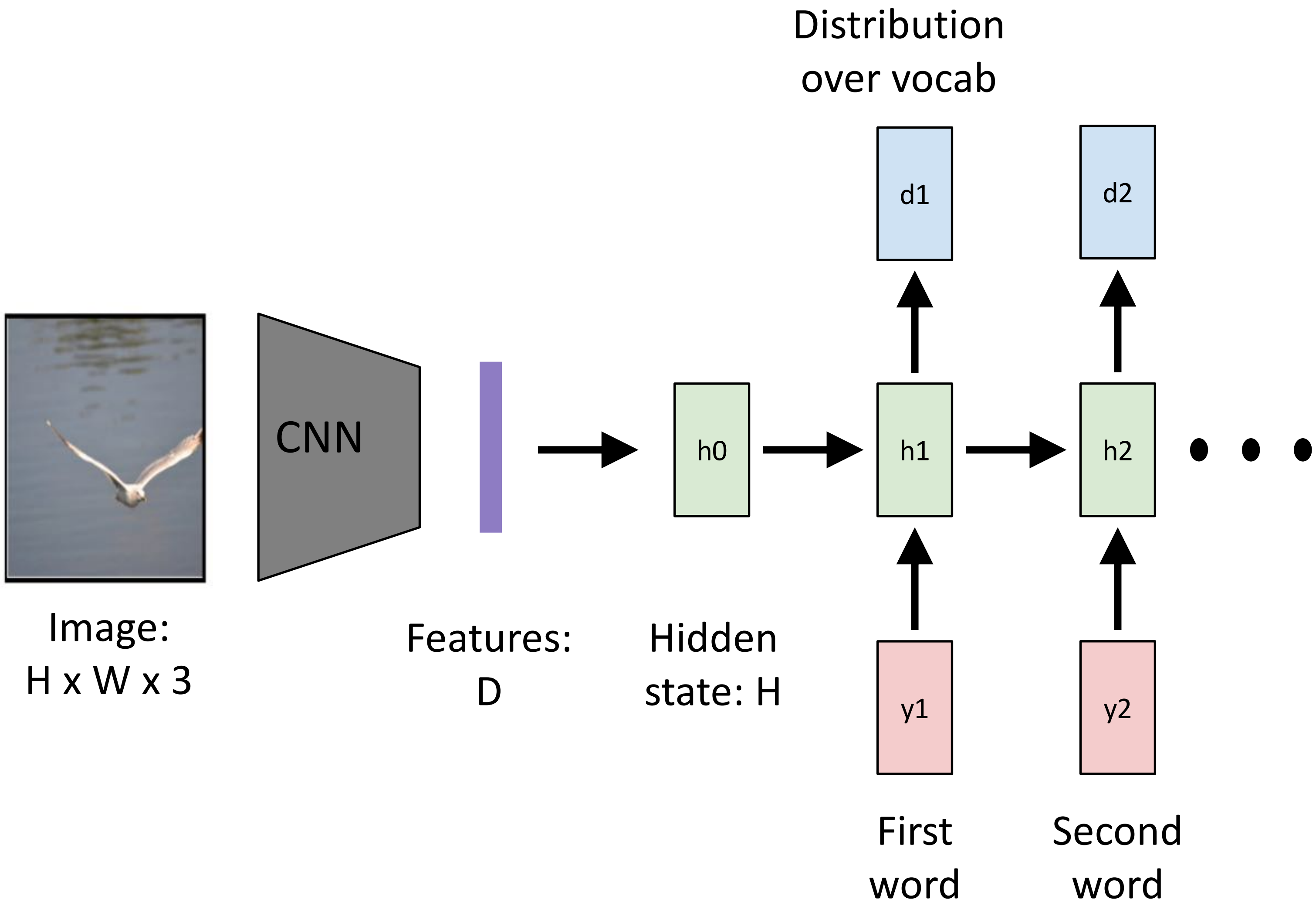
System	Ppl	BLEU
Winning WMT'14 system – *phrase-based + large LM* (Buck et al., 2014)		20.7
Existing NMT systems		
RNNsearch (Jean et al., 2015)		16.5
RNNsearch + unk replace (Jean et al., 2015)		19.0
RNNsearch + unk replace + large vocab + *ensemble* 8 models (Jean et al., 2015)		**21.6**
Our NMT systems		
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

Recall: RNN for Captioning

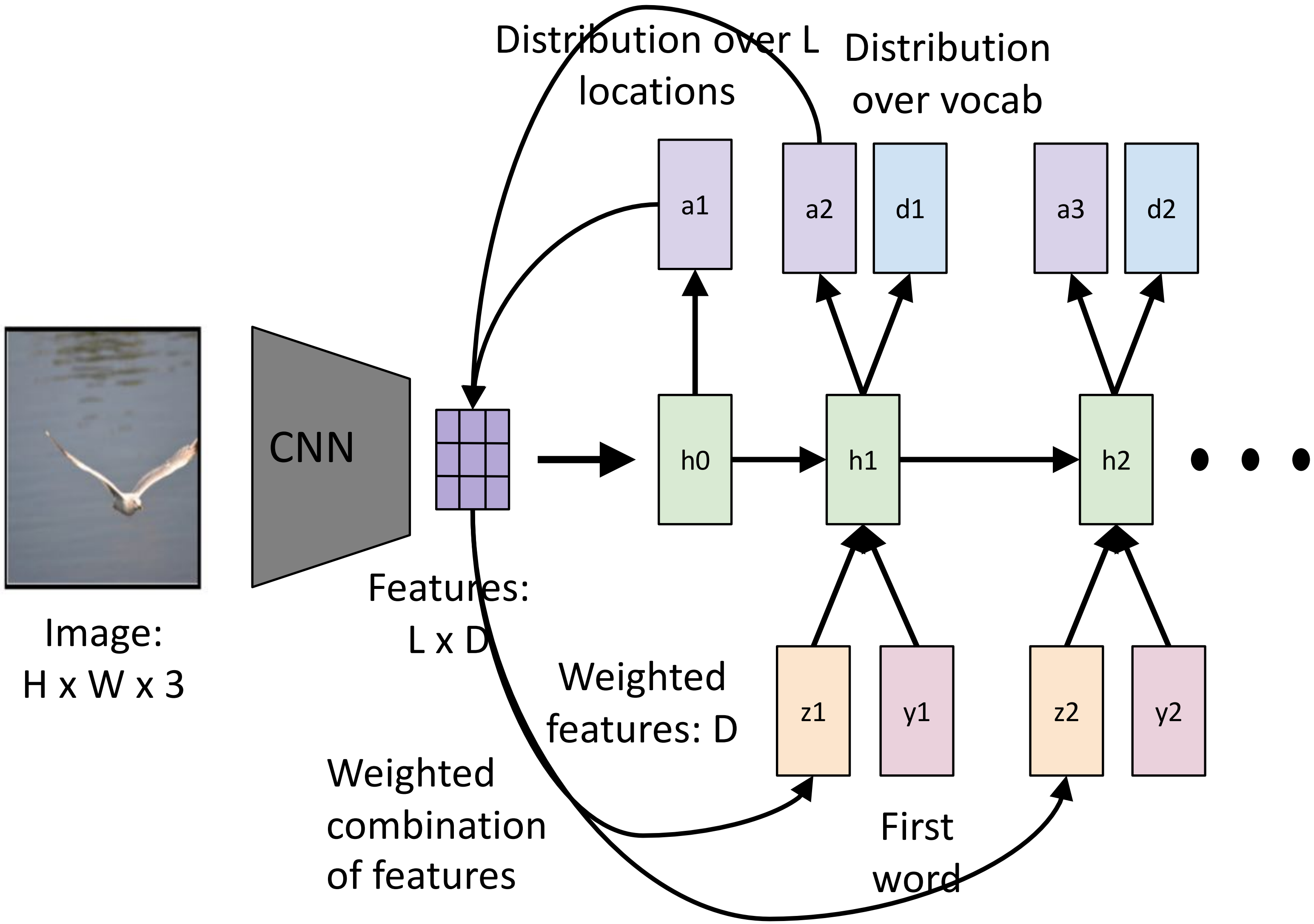
TECHNISCHE UNIVERSITÄT DARMSTADT



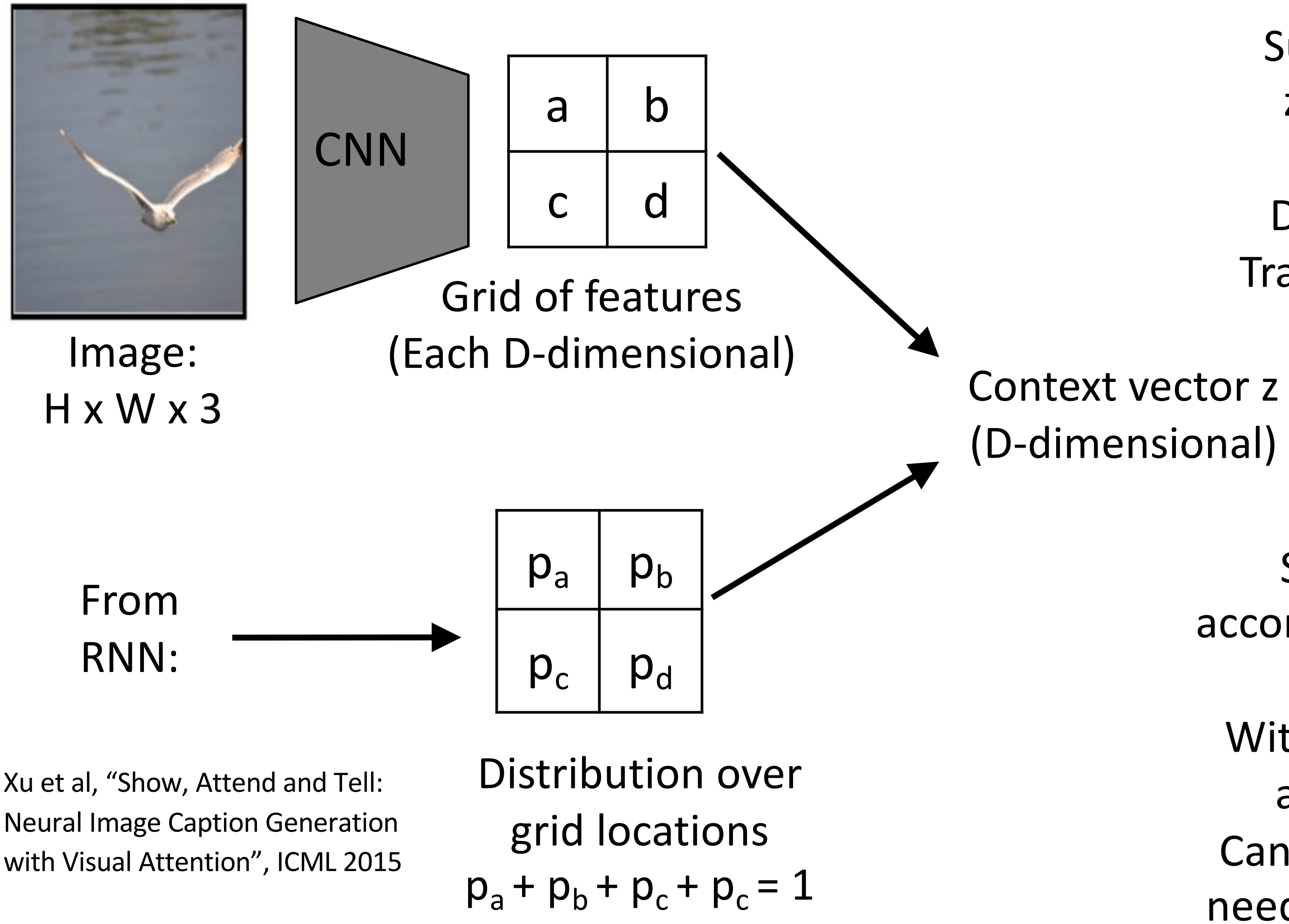
RNN only looks at whole image, once

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

Soft Attention for Captioning



Soft vs Hard Attention



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

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

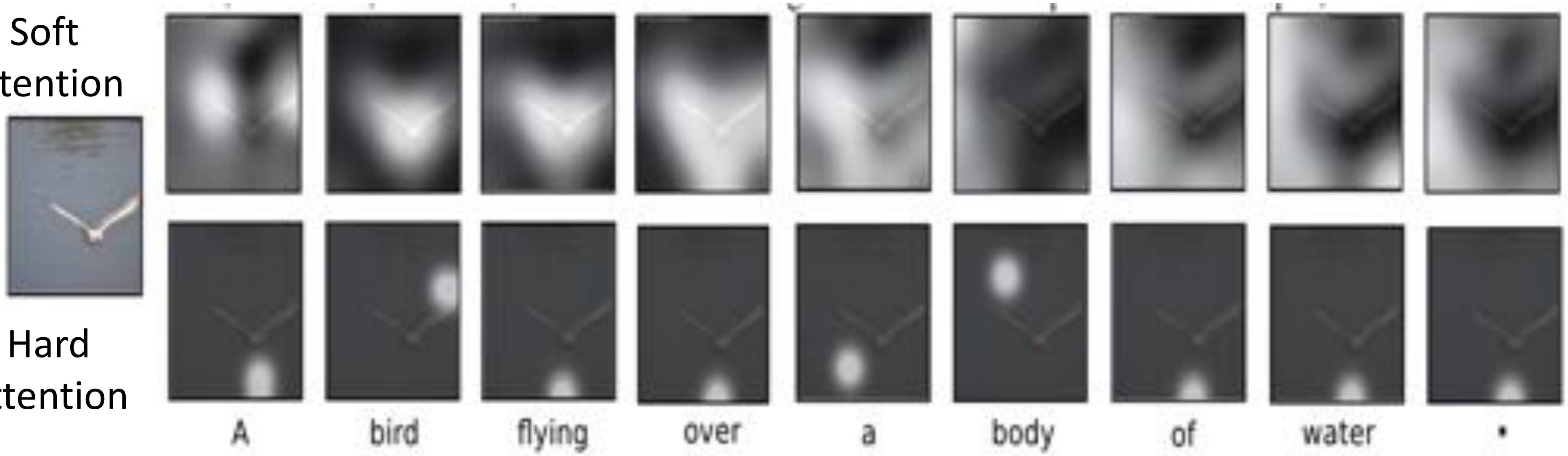
Hard attention:
Sample ONE location
according to p, z = that vector

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

Soft Attention for Captioning

Soft
attention

Hard
attention



Soft Attention for Captioning

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

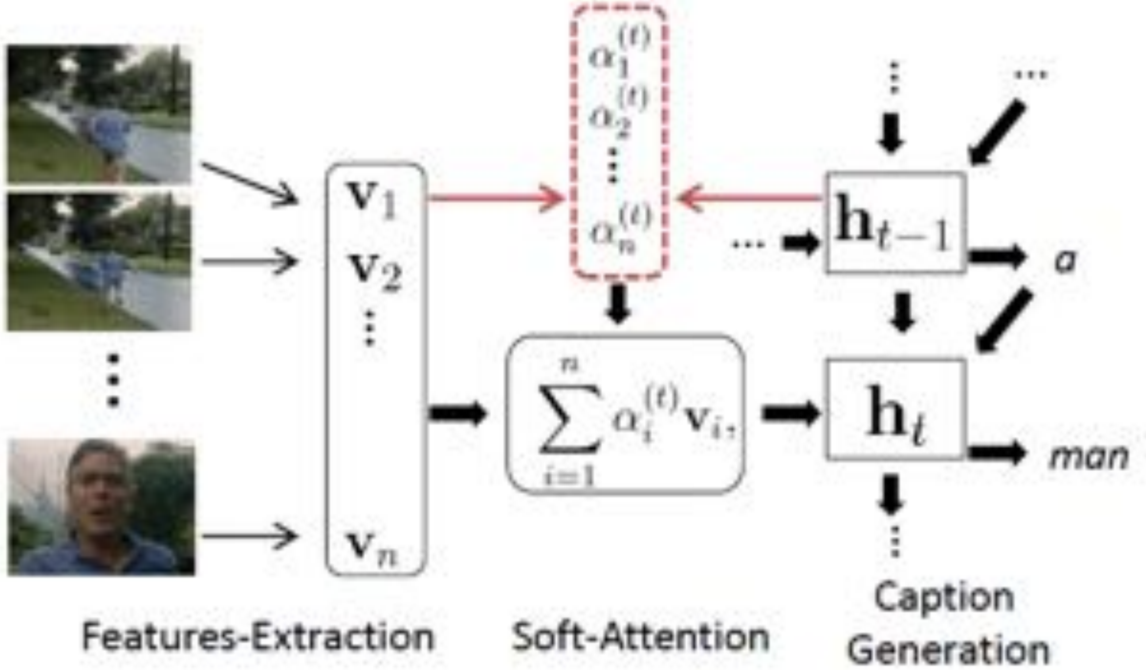
Soft Attention for Video

“Descr

.5.

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.

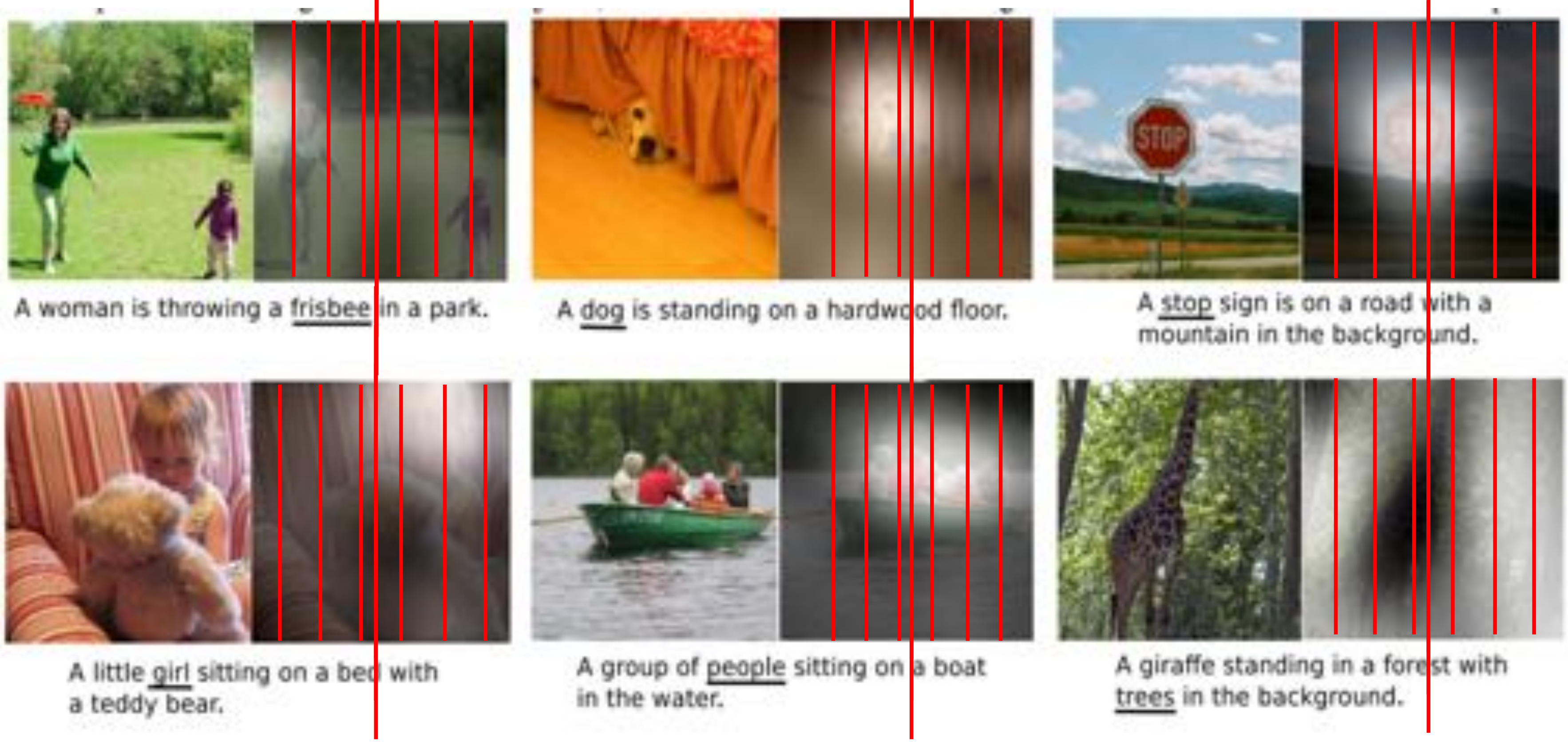
Model	Youtube2Text BLEU	METEOR	CIDEr	Perplexity	DVS BLEU	METEOR	CIDEr	Perplexity
Enc-Dec (Basic)	0.3869	0.2868	0.4478	33.09	0.003	0.044	0.044	88.28
+ Local (3-D CNN)	0.3875	0.2832	0.5087	33.42	0.004	0.051	0.050	84.41
+ Global (Temporal Attention)	0.4028	0.2900	0.4801	27.89	0.003	0.040	0.047	66.63
+ Local + 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.

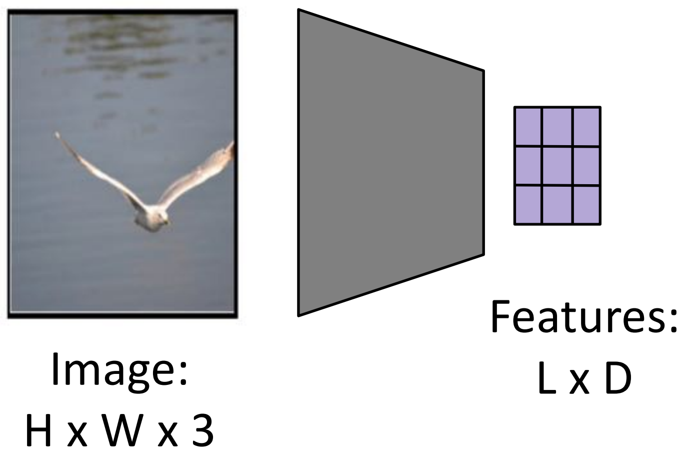
Soft Attention for Captioning

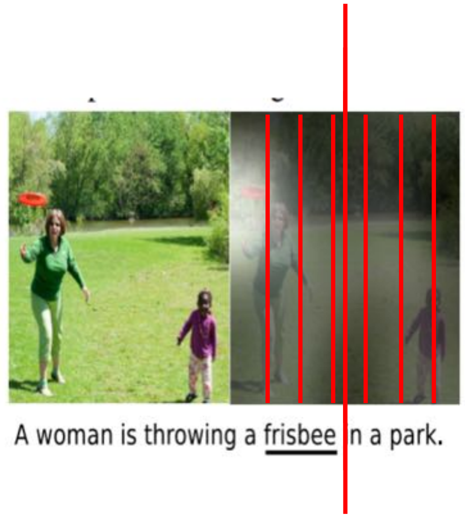
TECHNISCHE UNIVERSITÄT DARMSTADT

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



Attending to arbitrary regions?

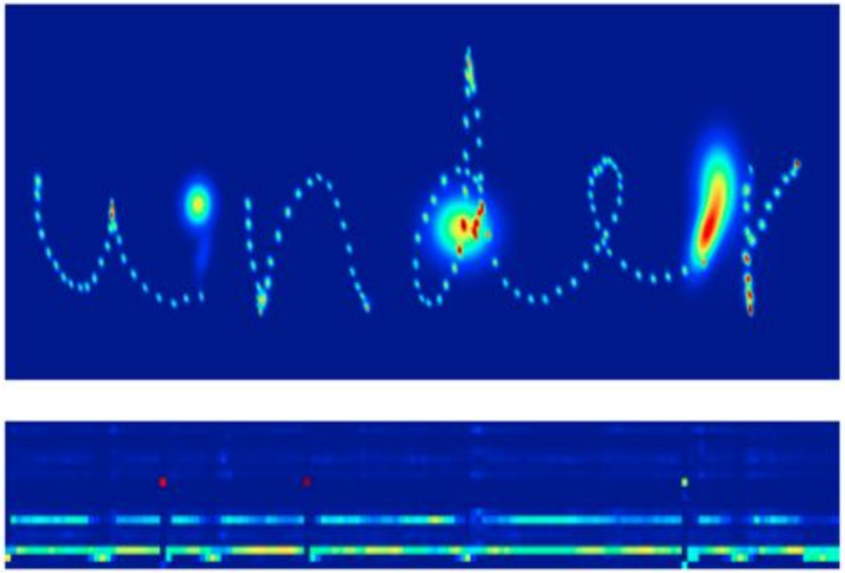




Attention mechanism from Show, Attend, and Tell only lets us softly attend to fixed grid positions … can we do better?

Attending to Arbitrary Regions

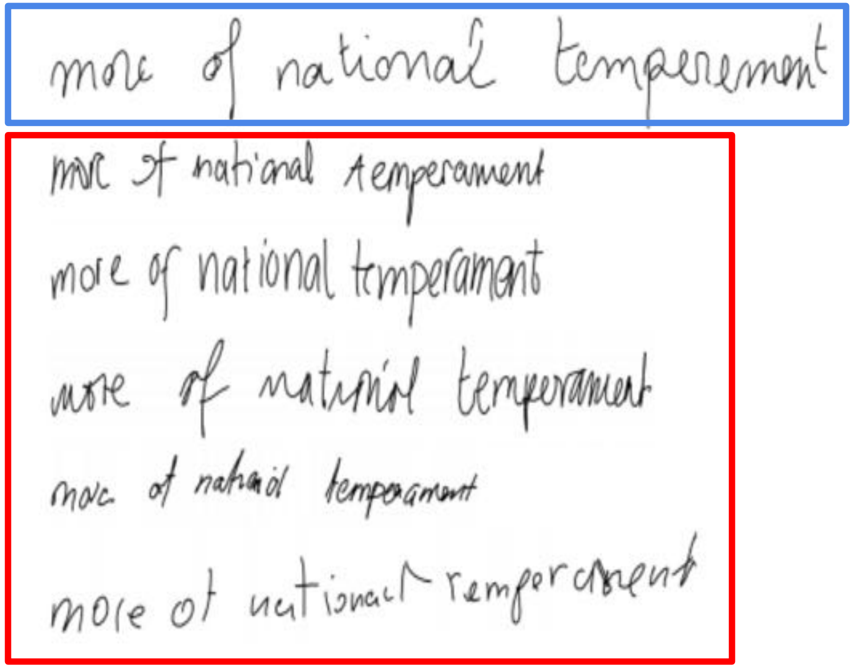
- Read text, generate handwriting using an RNN
- Attend to arbitrary regions of the **output** by predicting params of a mixture model



Graves, “Generating Sequences with Recurrent Neural Networks”, arXiv 2013

Which are real and which are generated?

REAL

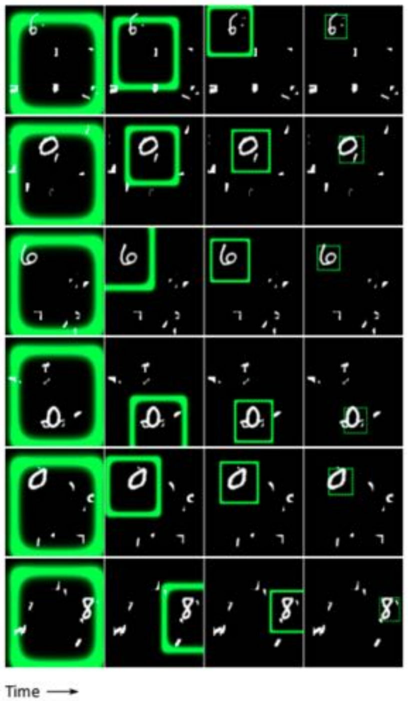


GENERATED

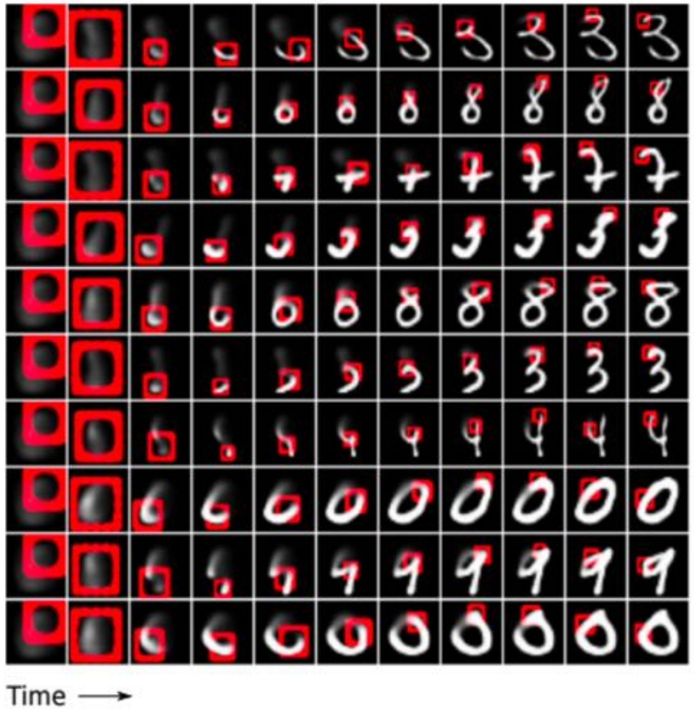
Attending to Arbitrary Regions: DRAW

TECHNISCHE UNIVERSITÄT DARMSTADT

Classify images by attending to arbitrary regions of the *input*

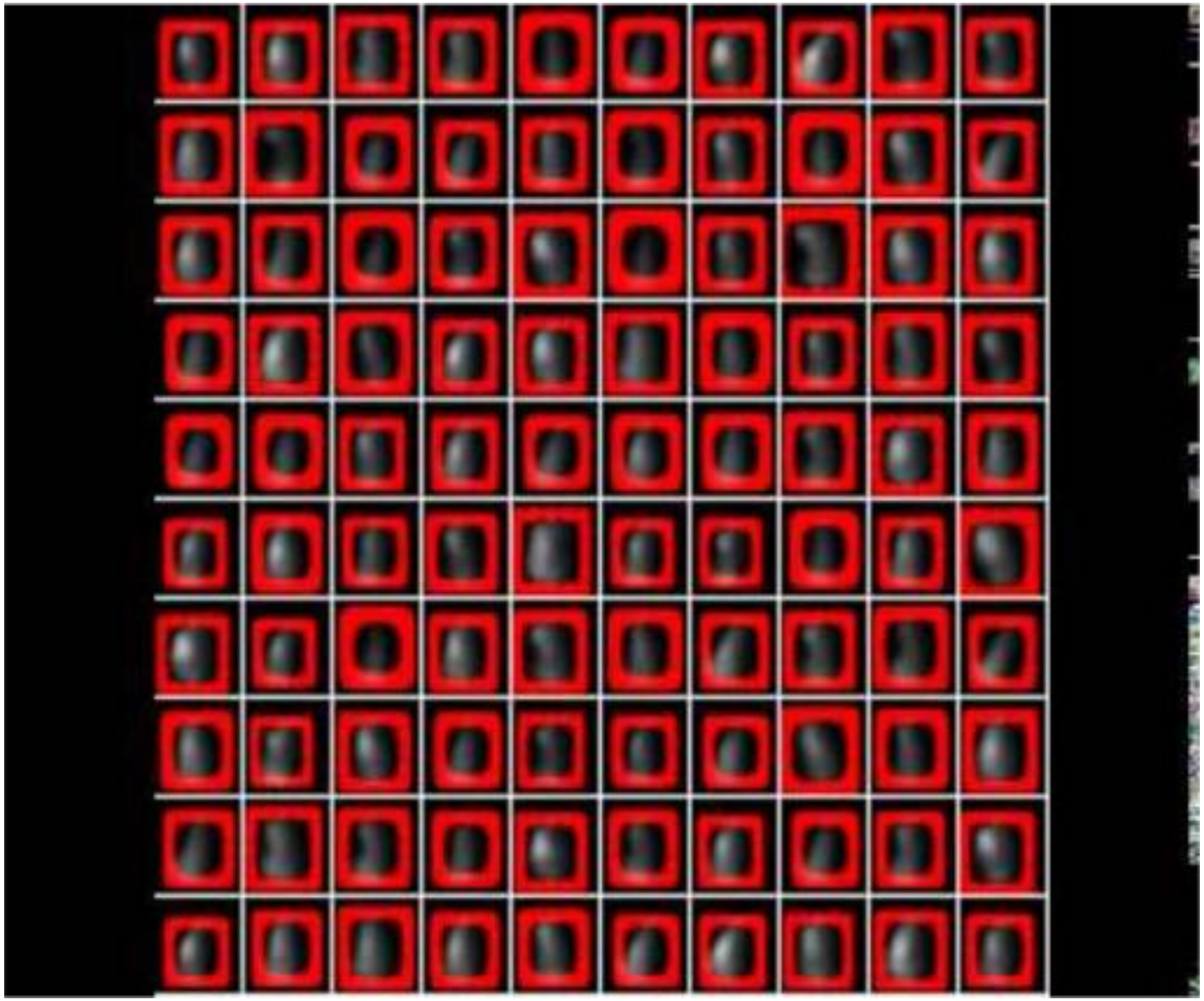


Generate images by attending to arbitrary regions of the *output*



Gregor et al, "DRAW: A Recurrent Neural Network For Image Generation", ICML 2015

TECHNISCHE UNIVERSITÄT DARMSTADT



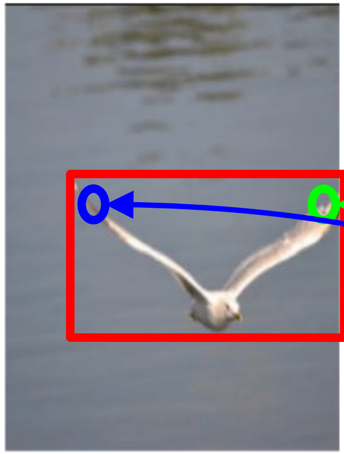
Attending to Arbitrary Regions: Spatial Transformer Networks

Attention mechanism similar to DRAW, but easier to explain

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

Spatial Transformer Networks

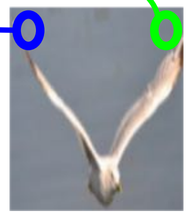
TECHNISCHE UNIVERSITÄT DARMSTADT



Input image: H x W x 3

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

Can we make this function differentiable?

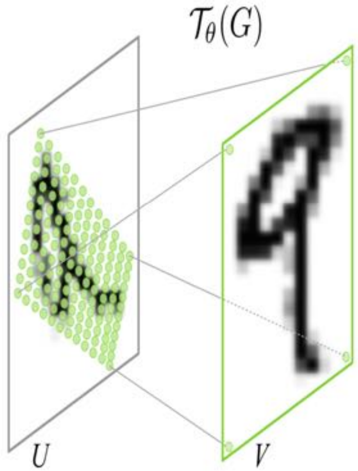


Cropped and rescaled image: X x Y x 3

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**

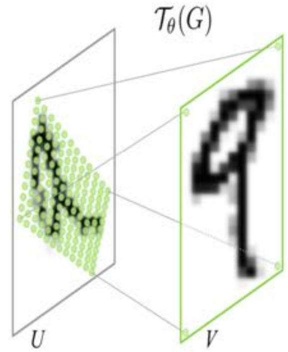
Then use **bilinear interpolation** to compute output

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

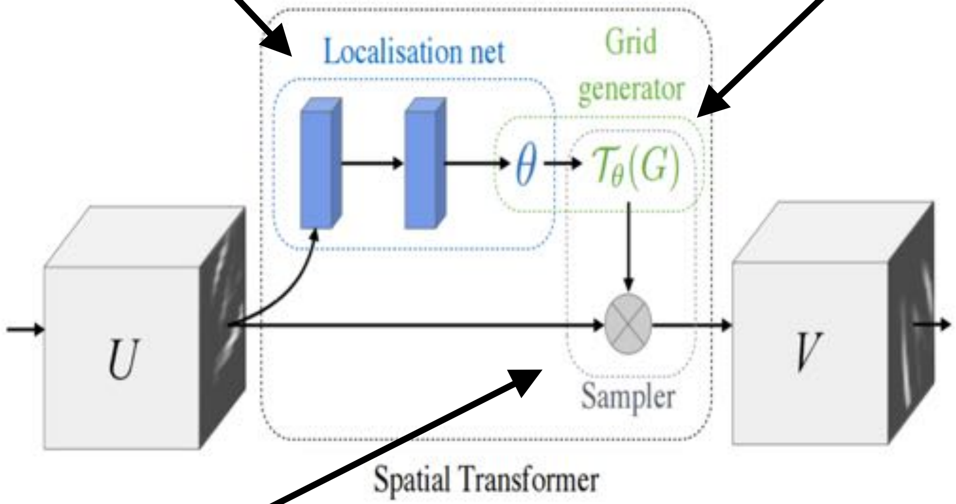
Spatial Transformer Networks

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

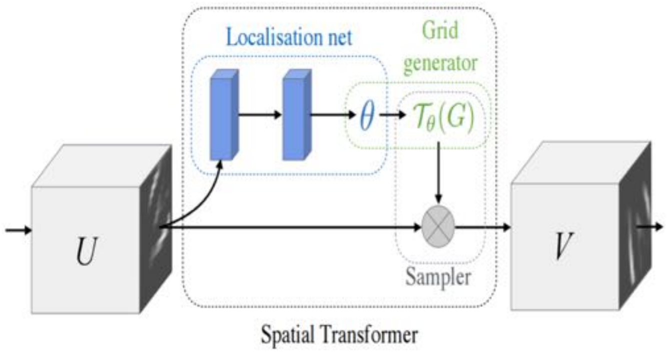
Output: Region of interest from input

Sampler uses bilinear interpolation to produce output

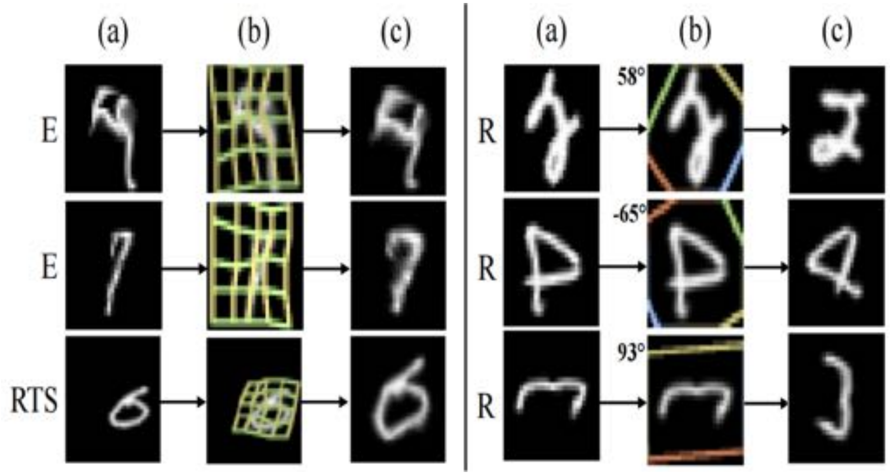
$$V_i^c = \sum_n^H \sum_m^W U_{nm}^c \max(0, 1 - |x_i^s - m|) \max(0, 1 - |y_i^s - n|)$$

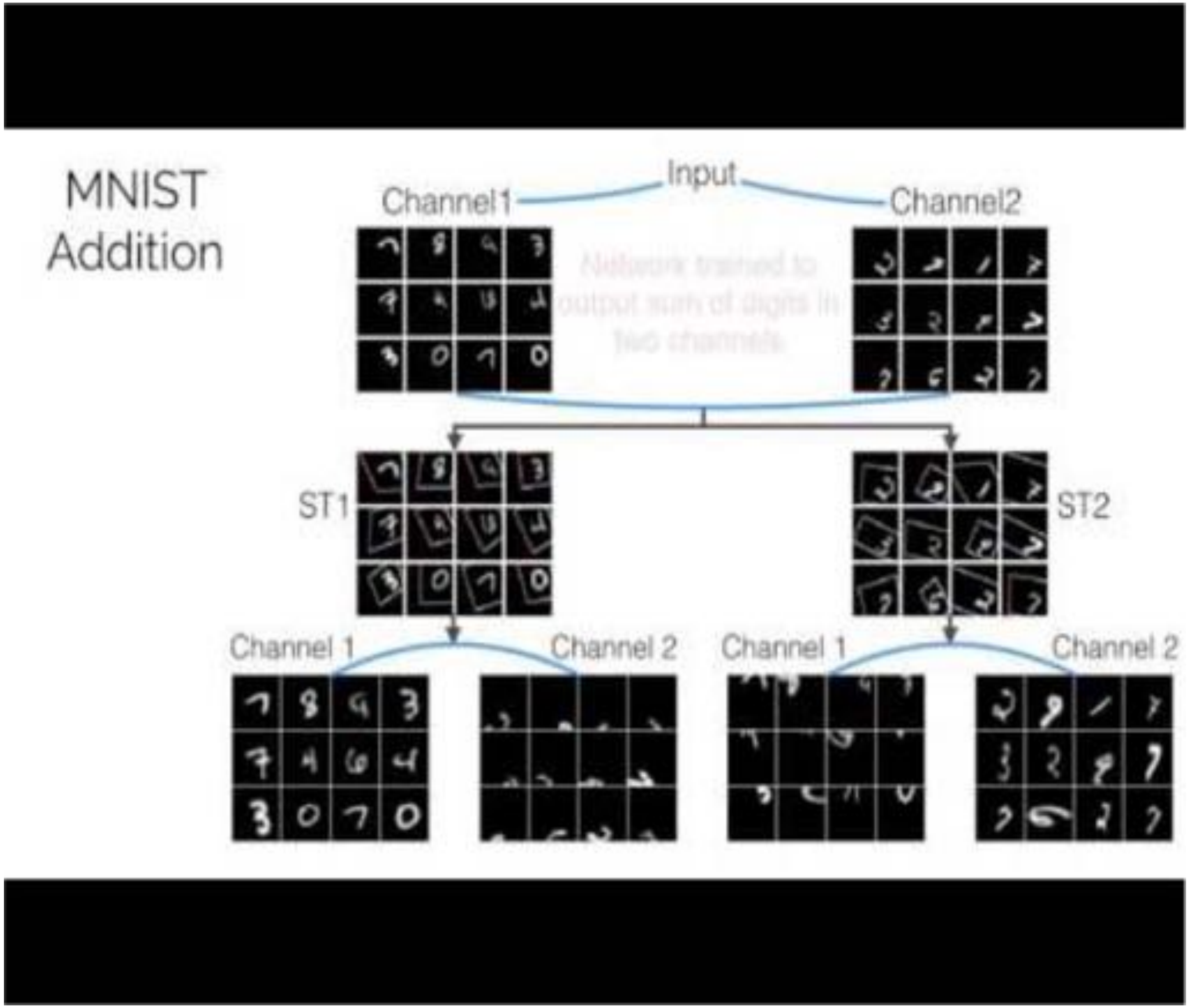
Spatial Transformer Networks

Differentiable “attention / transformation” module



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




MNIST Addition
Input
Channel1
Channel2
Network trained to output sum of digits in two channels
ST1
ST2
Channel 1
Channel 2
Channel 1
Channel 2

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 explanation
- 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.