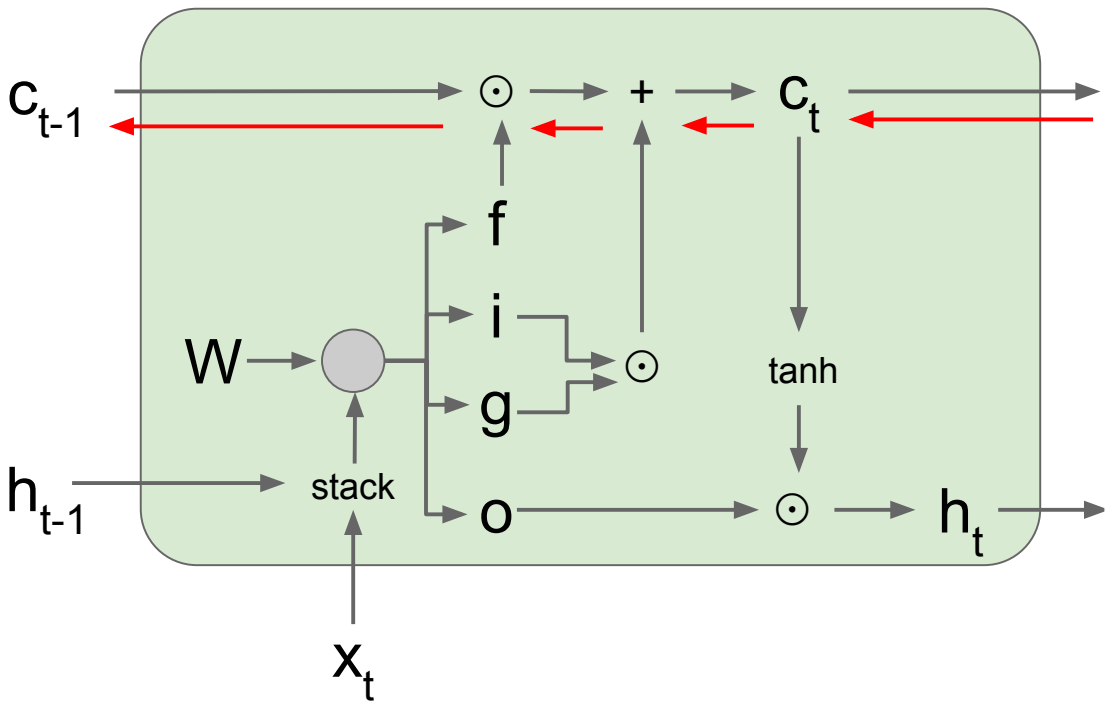
Recurrent Neural Networks

Changyou Chen

Department of Computer Science and Engineering
Universitpy at Buffalo, SUNY
changyou@buffalo.edu

March 28, 2019

Long Short Term Memory: Gradient Flow



Backpropagation from $\mathbf{c}_t$ to $\mathbf{c}_{t-1}$ only elementwise multiplication by $\mathbf{f}$, no direct matrix multiply by $\mathbf{W} \Rightarrow$ no gradient vanishing.

$$\begin{pmatrix} \mathbf{i} \\ \mathbf{f} \\ \mathbf{o} \\ \mathbf{g} \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} \left[\mathbf{W} \begin{pmatrix} \mathbf{h}_{t-1} \\ \mathbf{x}_t \end{pmatrix} \right]$$

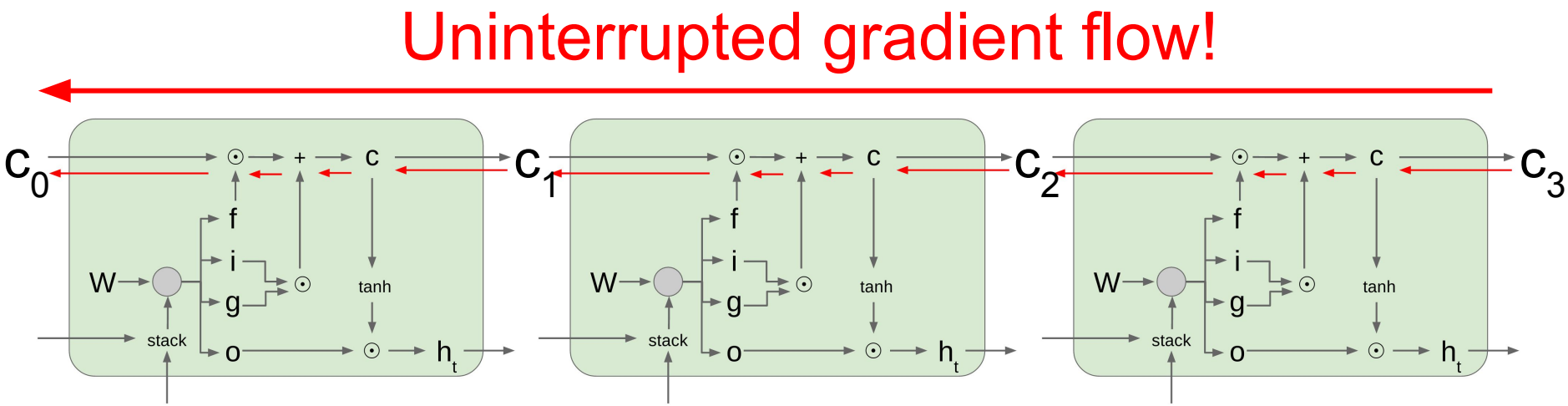
$$\mathbf{c}_t = \mathbf{f} \odot \mathbf{c}_{t-1} + \mathbf{i} \odot \mathbf{g}$$

$$\mathbf{h}_t = \mathbf{o} \odot \tanh(\mathbf{c}_t)$$

Hochreiter & Schmidhuber, Neural Computation 1997

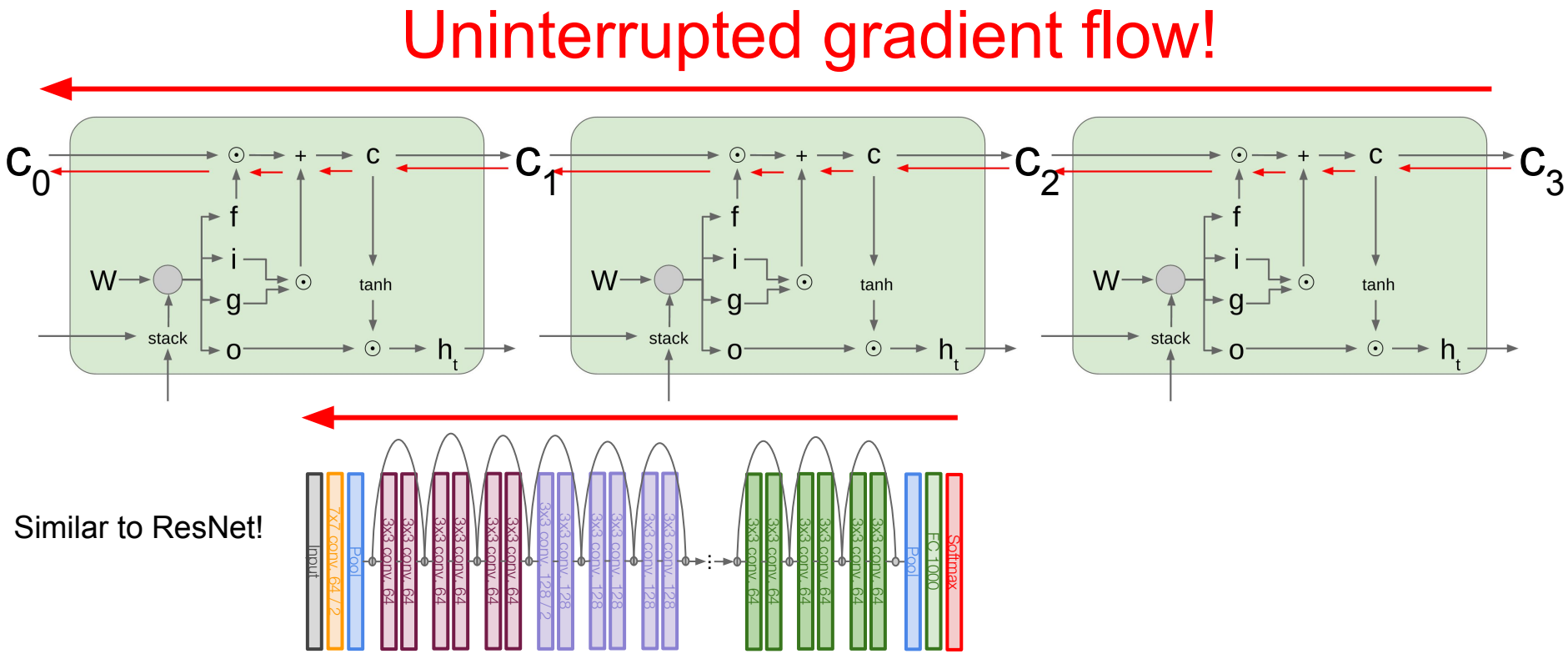
Long Short Term Memory: Gradient Flow

Gradients directly backpropagate through time.



Long Short Term Memory: Gradient Flow

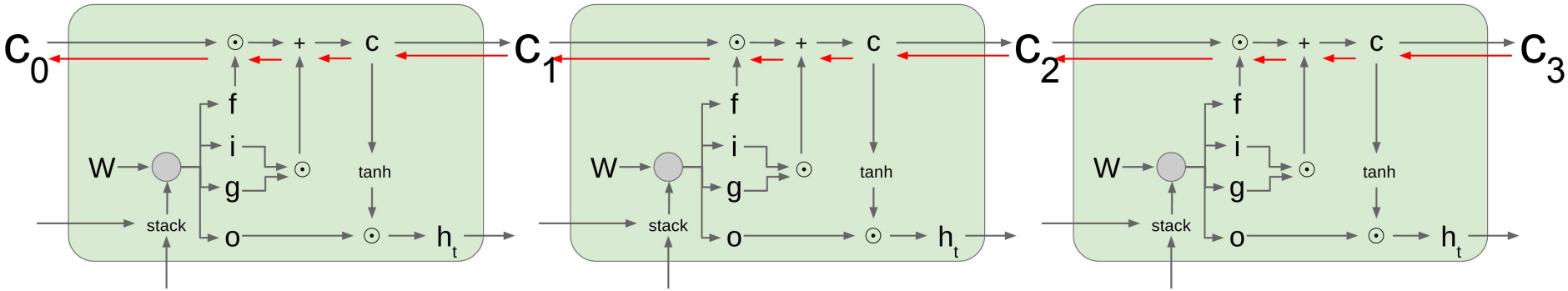
Gradients directly backpropagate through time.



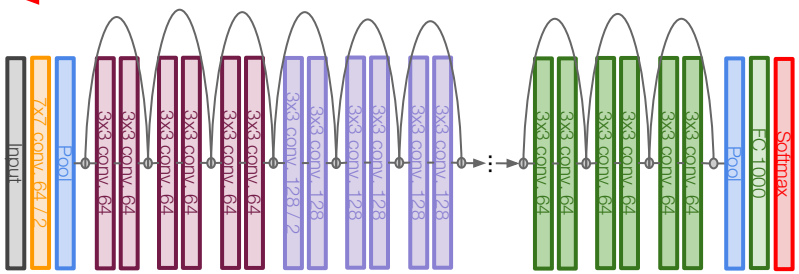
Long Short Term Memory: Gradient Flow

Gradients directly backpropagate through time.

Uninterrupted gradient flow!



Similar to ResNet!



In between:
Highway Networks

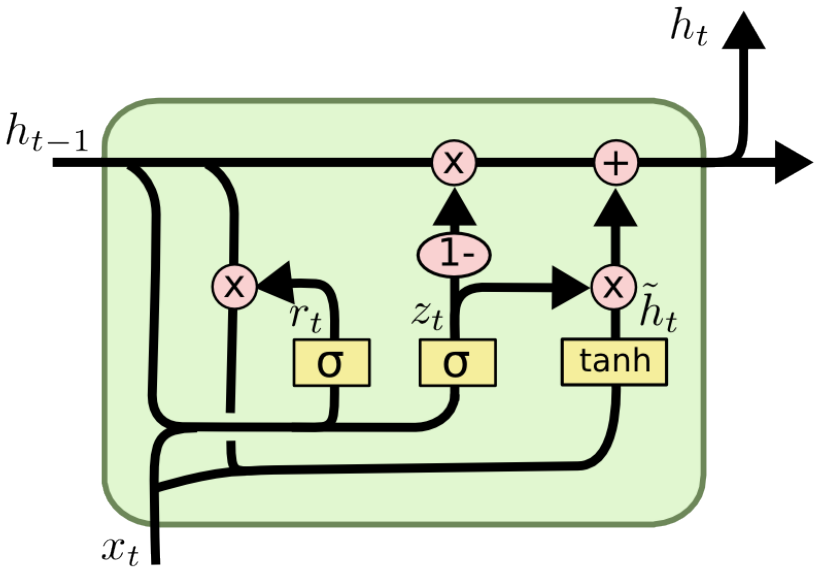
$$g = T(x, W_T)$$
$$y = g \odot H(x, W_H) + (1 - g) \odot x$$

Srivastava et al, "Highway Networks", ICML DL Workshop 2015

Other RNN Variants

Guideline

Need uninterrupted gradient flows to avoid gradient vanishing!



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

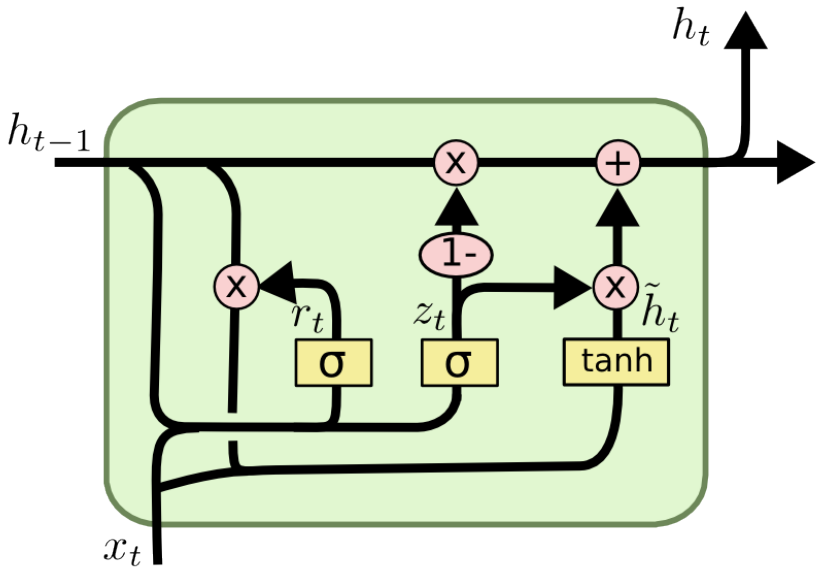
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Other RNN Variants

Gated Recurent Unit (GRU) [Cho et al., 2014]

$$\mathbf{r}_t = \sigma\left(\mathbf{W}_{xr}\,\mathbf{x}_t + \mathbf{W}_{hr}\,\mathbf{h}_{t-1} + \mathbf{b}_r\right)$$
$$\mathbf{z}_t = \sigma\left(\mathbf{W}_{xz}\,\mathbf{x}_t + \mathbf{W}_{hz}\,\mathbf{h}_{t-1} + \mathbf{b}_z\right)$$
$$\tilde{\mathbf{h}}_t = \tanh\left(\mathbf{W}_{xh}\,\mathbf{x}_t + \mathbf{W}_{hh}\left(\mathbf{r}_t \odot \mathbf{h}_{t-1}\right) + \mathbf{b}_h\right)$$
$$\mathbf{h}_t = \mathbf{z}_t \odot \tilde{\mathbf{h}}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1}$$



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Other RNN Variants [Jozefowicz et al., 2015]

$$
\begin{aligned}
\mathbf{z}_t &= \sigma\left(\mathbf{W}_{xz}\,\mathbf{x}_t + \mathbf{b}_z\right) \\
\mathbf{r}_t &= \sigma\left(\mathbf{W}_{xr}\,\mathbf{x}_t + \mathbf{W}_{hr}\,\mathbf{h}_t + \mathbf{b}_r\right) \\
\mathbf{h}_{t+1} &= \tanh\left(\mathbf{W}_{hh}\left(\mathbf{r}_t \odot \mathbf{h}_t\right) + \tanh(\mathbf{x}_t)\right. \\
&\qquad \left. + \mathbf{b}_h\right)\mathbf{z}_t + \mathbf{h}_t \odot (1 - \mathbf{z}_t)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{z}_t &= \sigma\left(\mathbf{W}_{xz}\,\mathbf{x}_t + \mathbf{W}_{hz}\,\mathbf{h}_t + \mathbf{b}_z\right) \\
\mathbf{r}_t &= \sigma\left(\mathbf{x}_t + \mathbf{b}_r\right) \\
\mathbf{h}_{t+1} &= \tanh\left(\mathbf{W}_{hh}\left(\mathbf{r}_t \odot \mathbf{h}_t\right) + \mathbf{W}_{xh}\,\mathbf{x}_t\right. \\
&\qquad \left. + \mathbf{b}_h\right)\mathbf{z}_t + \mathbf{h}_t \odot (1 - \mathbf{z}_t)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{z}_t &= \sigma\left(\mathbf{W}_{xz}\,\mathbf{x}_t + \mathbf{W}_{hz}\tanh(\mathbf{h}_t) + \mathbf{b}_z\right) \\
\mathbf{r}_t &= \sigma\left(\mathbf{W}_{xr}\,\mathbf{x}_t + \mathbf{W}_{hr}\,\mathbf{h}_t + \mathbf{b}_r\right) \\
\mathbf{h}_{t+1} &= \tanh\left(\mathbf{W}_{hh}\left(\mathbf{r}_t \odot \mathbf{h}_t\right) + \mathbf{W}_{xh}\,\mathbf{x}_t\right. \\
&\qquad \left. + \mathbf{b}_h\right)\mathbf{z}_t + \mathbf{h}_t \odot (1 - \mathbf{z}_t)
\end{aligned}
$$

Summary

1. RNNs allow a lot of flexibility in architecture design.
2. Vanilla RNNs are simple but don't work very well.
3. Common to use LSTM or GRU: their additive interactions improve gradient flow.
4. Backward flow of gradients in RNN can explode or vanish:
   - exploding is controlled with gradient clipping.
   - vanishing is controlled with additive interactions (LSTM).
5. Better/simpler architectures are a hot topic of current research.
6. Better understanding (both theoretical and empirical) is needed.

Applications[1]

[1] Partially adapted from http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf

Recap: Long Short Term Memory

Gradients directly backpropagate through time.

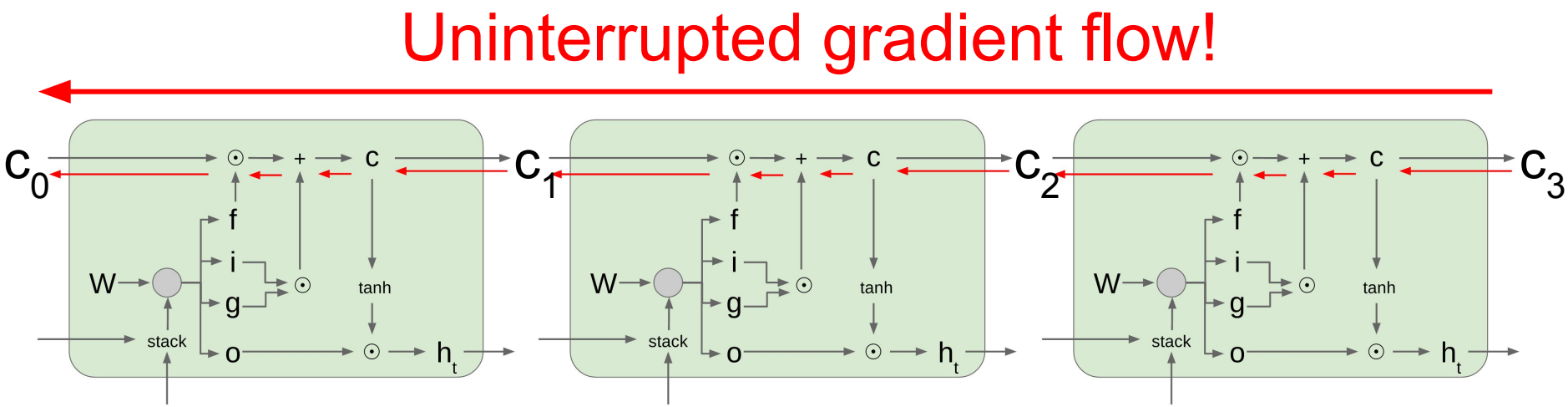


Image Captioning

How to predict a sequence of text (caption) from an image input?

Image Captioning

CNN for feature extraction; Many-to-many RNN for caption generation.

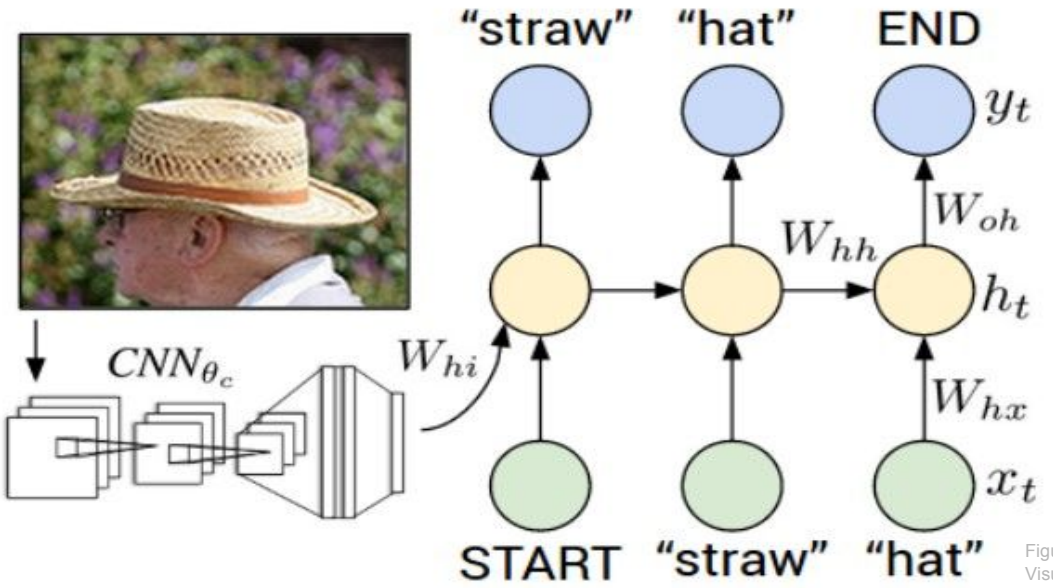


Figure from Karpathy et a, "Deep Visual-Semantic Alignments for Generating Image Descriptions", CVPR 2015; figure copyright IEEE, 2015. Reproduced for educational purposes.

Explain Images with Multimodal Recurrent Neural Networks, Mao et al.
Deep Visual-Semantic Alignments for Generating Image Descriptions, Karpathy and Fei-Fei
Show and Tell: A Neural Image Caption Generator, Vinyals et al.
Long-term Recurrent Convolutional Networks for Visual Recognition and Description, Donahue et al.
Learning a Recurrent Visual Representation for Image Caption Generation, Chen and Zitnick

Image Captioning

CNN for feature extraction; Many-to-many RNN for caption generation.

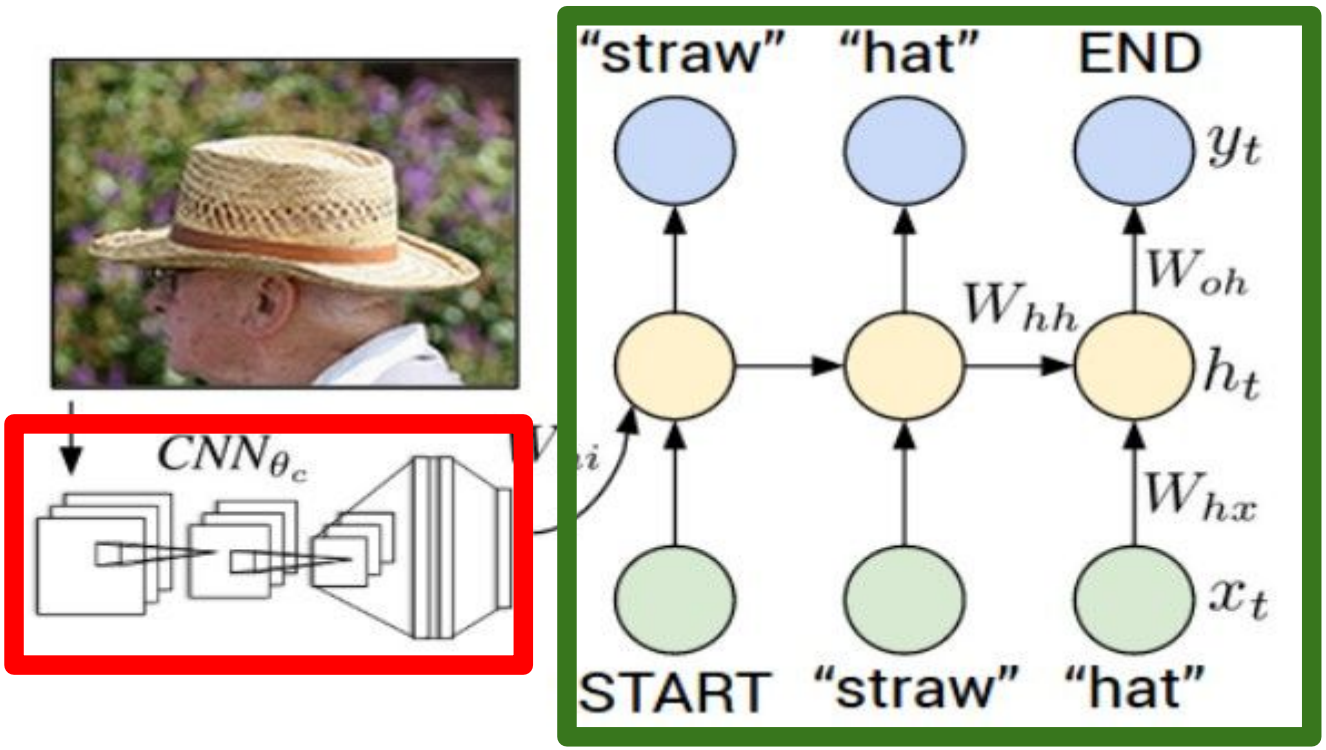


Image Captioning

This image is CC0 public domain

test image

Image Captioning

test image

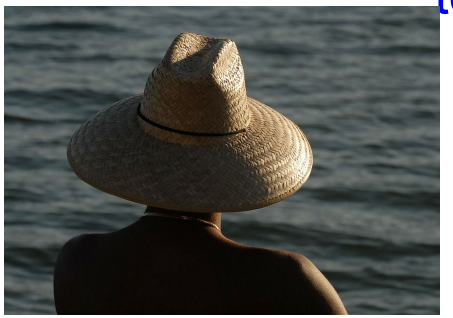
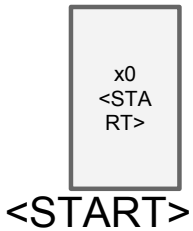


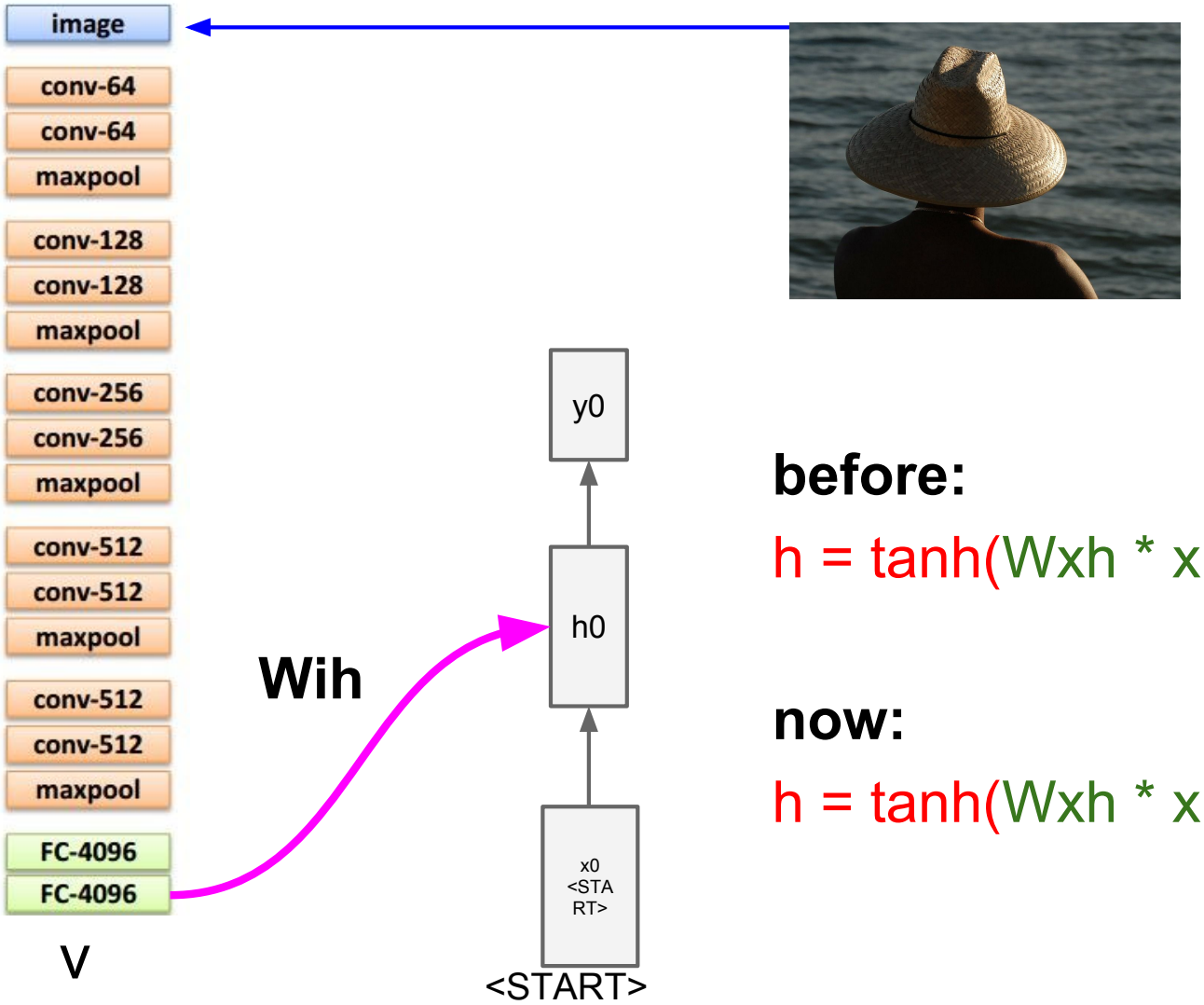
Image Captioning

test image



<START>

Image Captioning



before:

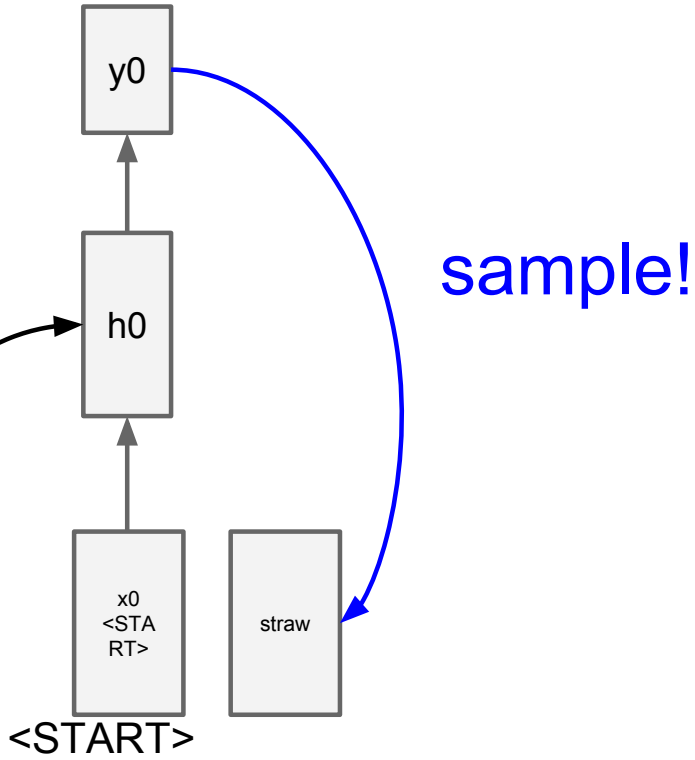
h = tanh(Wxh * x + Whh * h)

now:

h = tanh(Wxh * x + Whh * h **+ Wih * v**)

Image Captioning

test image



sample!

Image Captioning

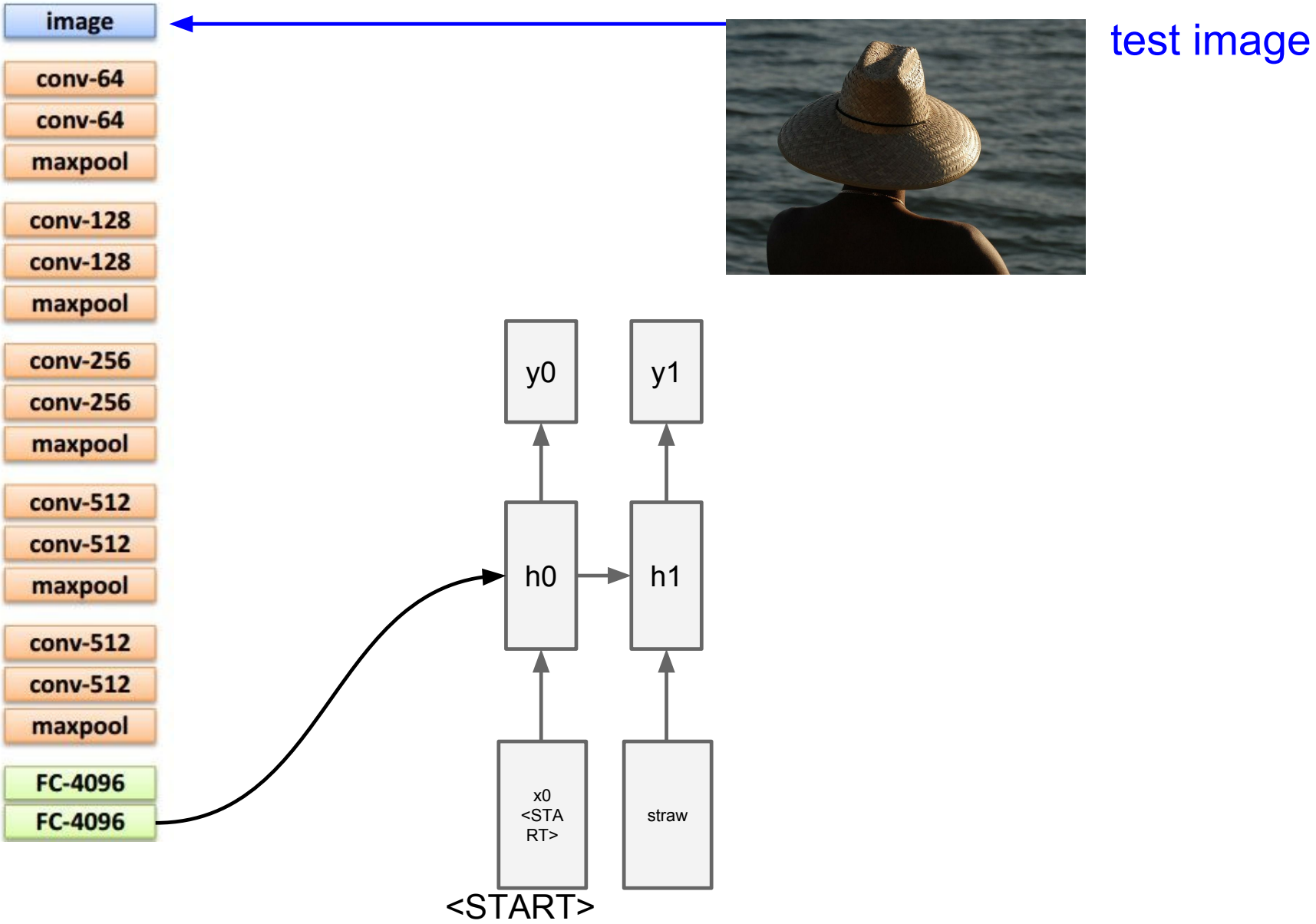


Image Captioning

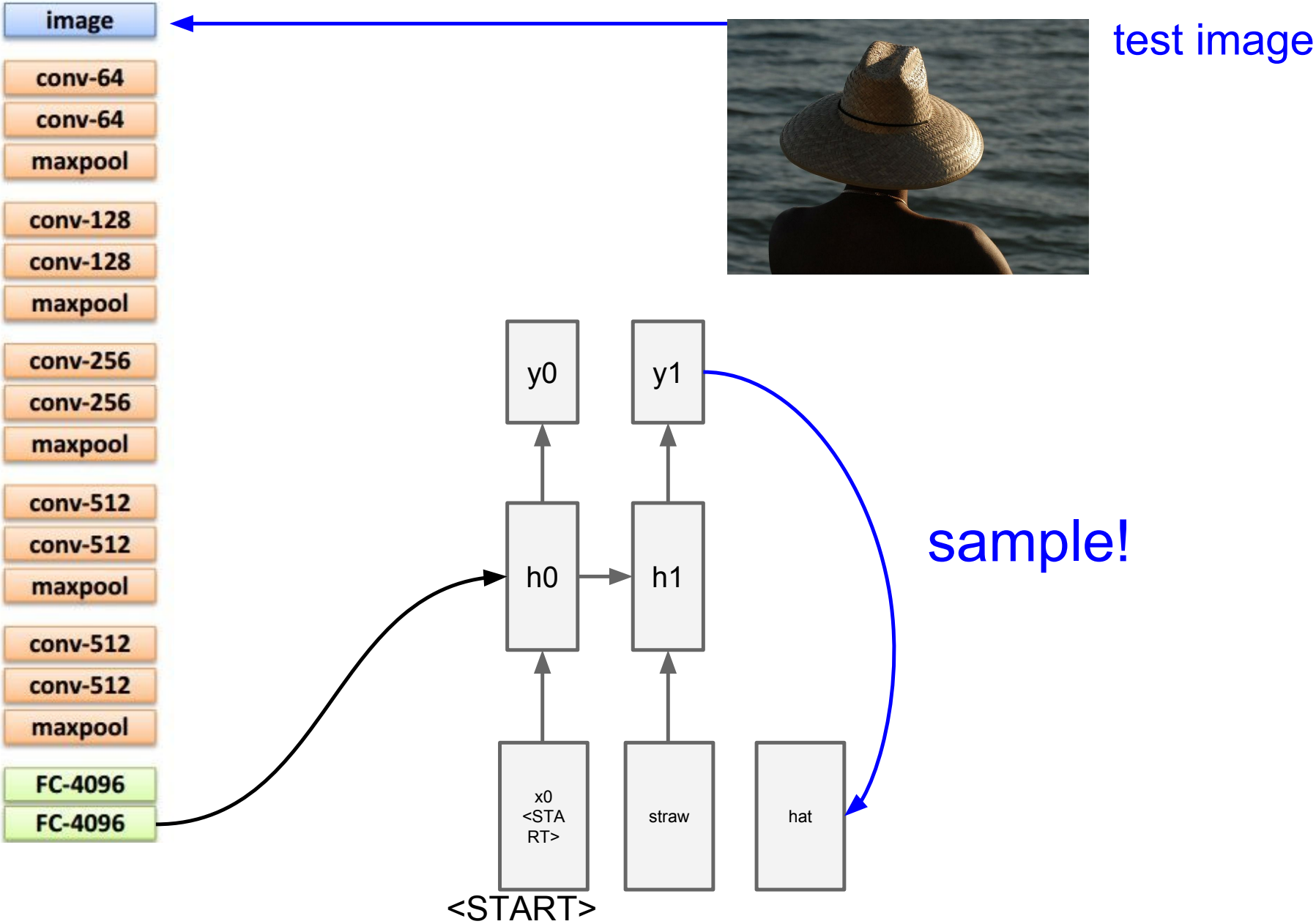


Image Captioning

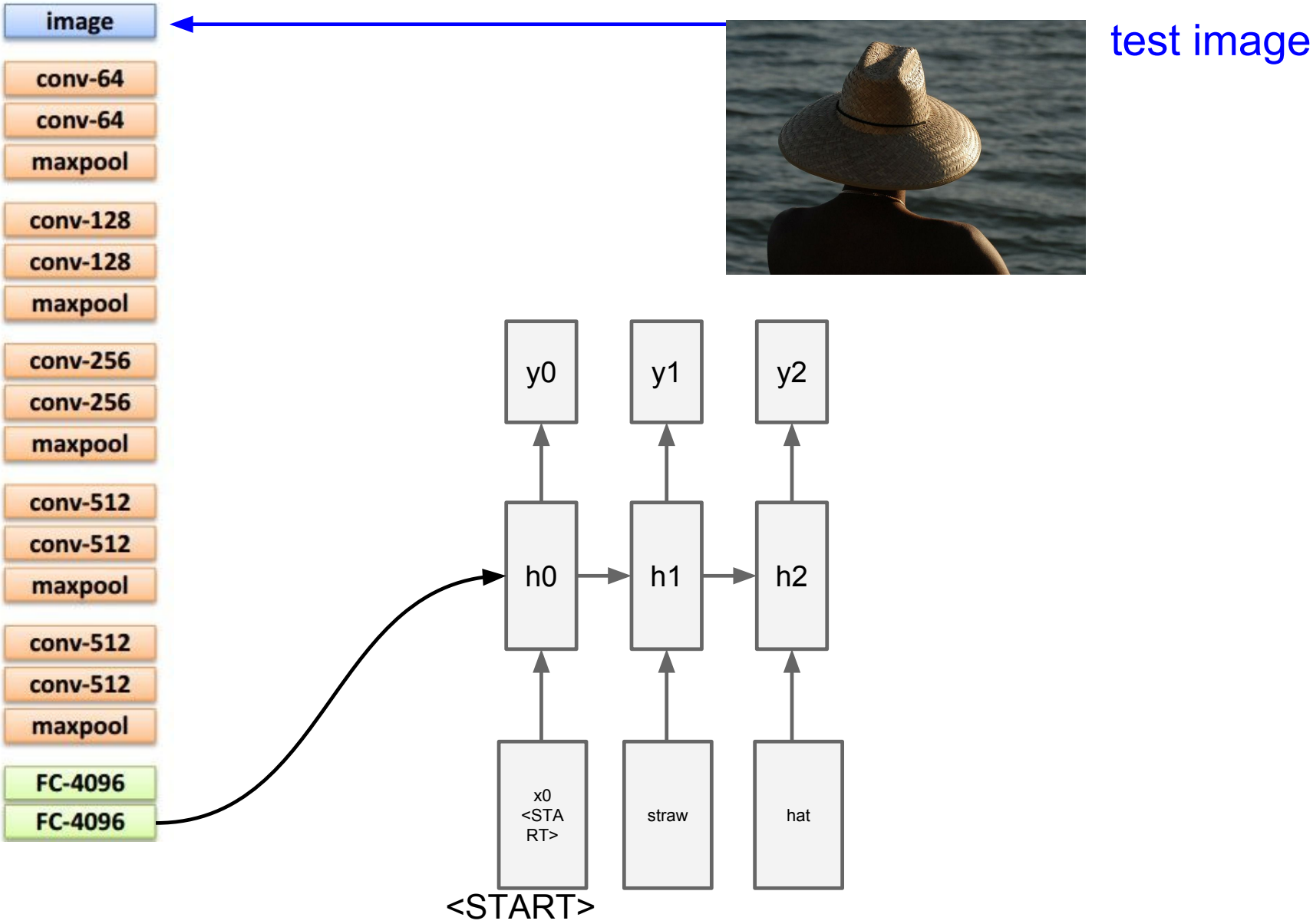


Image Captioning

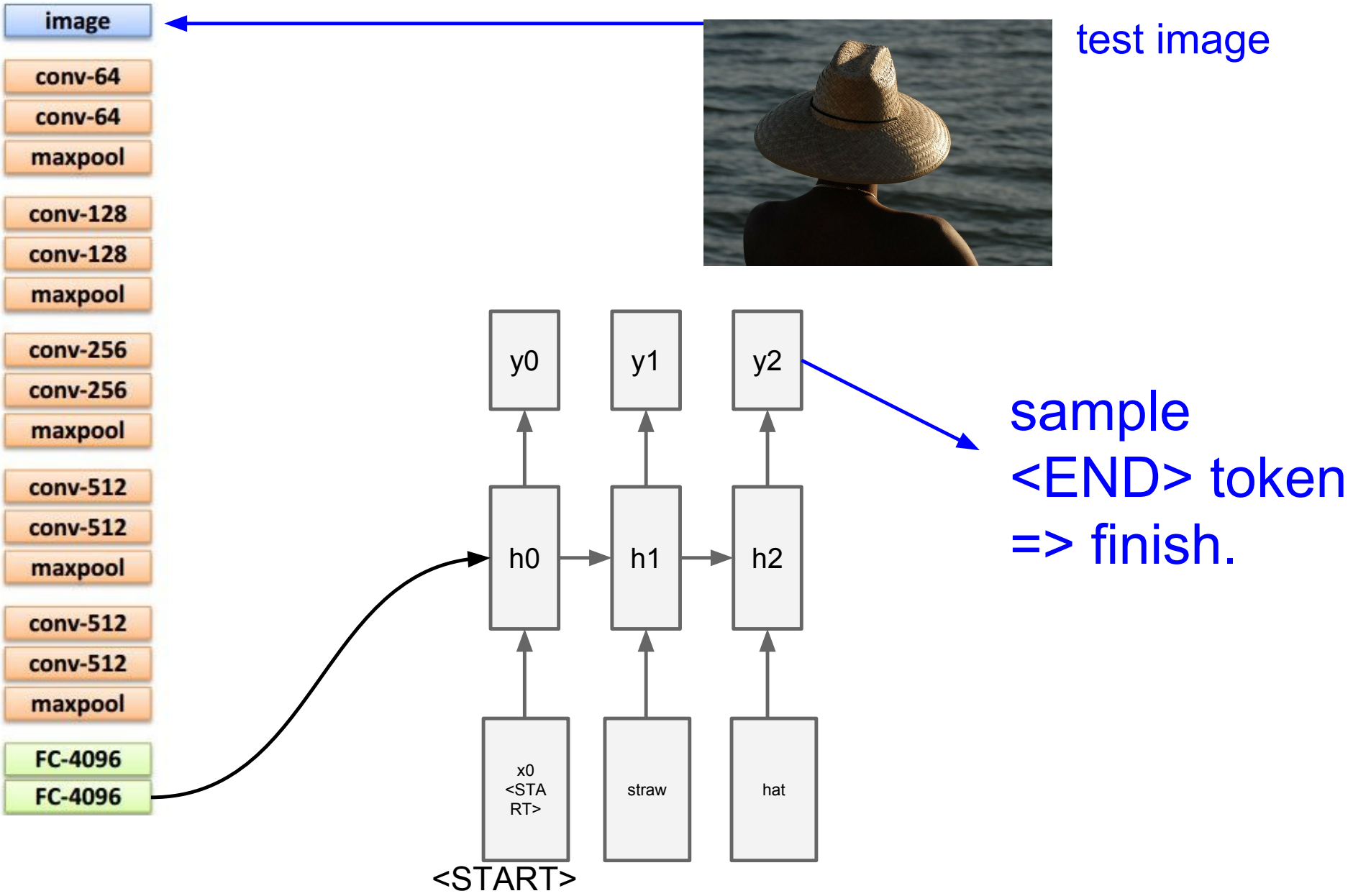
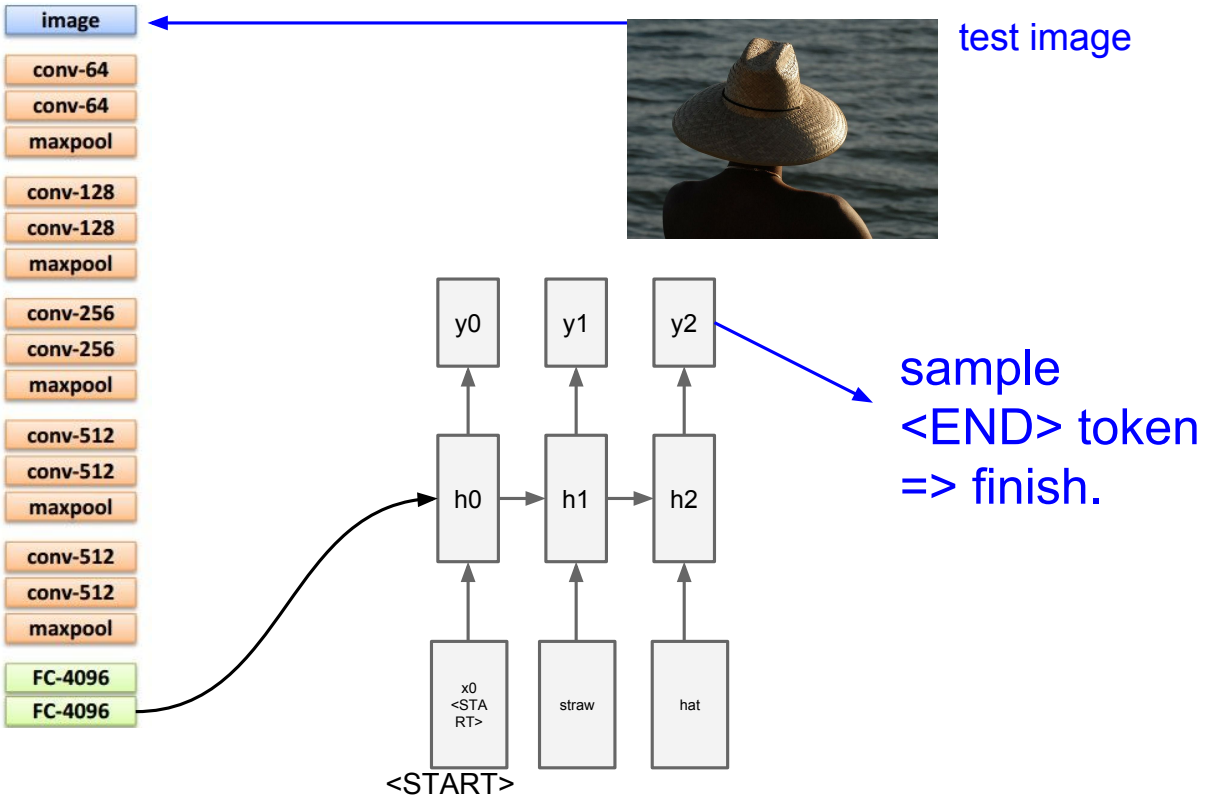


Image Captioning



Some modifications from Vinyals *et al.* 2014

- Only input image features in the first time step of an RNN.
- Use word embedding.
- Learn a feature embedding for CNN features.

- Typically fix the CNN parameters when training.

Image Captioning

Training

- Cross entropy loss for all time steps of all training image-caption pairs (I_i, S_i):

$$L(I, S) = -\sum_i \sum_t \log p_t(S_{i,t})$$

Inference

- **Sampling**: sample the first word according to p_1, then provide the corresponding embedding as input and sample p_2, continuing like this until we sample the special end-of-sentence token or some maximum length.
- **Beam Search**: iteratively consider the set of the k best sentences up to time t as candidates to generate sentences of size $t + 1$, and keep only the resulting best k of them:

  Better approximates $S = \arg\max_{S'} p(S'|I)$.

Image Captioning: Example Results

A cat sitting on a suitcase on the floor

A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

Two people walking on the beach with surfboards

A tennis player in action on the court

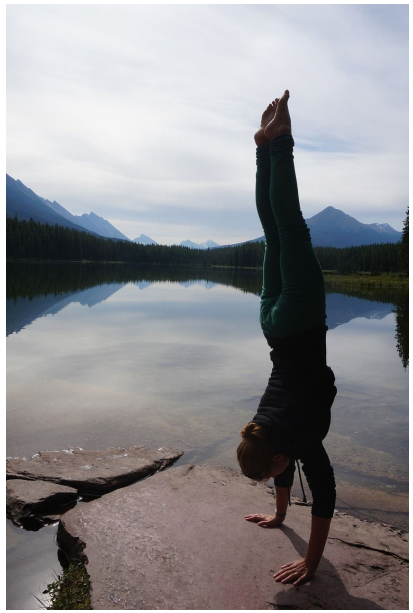
Two giraffes standing in a grassy field

A man riding a dirt bike on a dirt track

Image Captioning: Failure Cases

A woman is holding a cat in her hand

A person holding a computer mouse on a desk



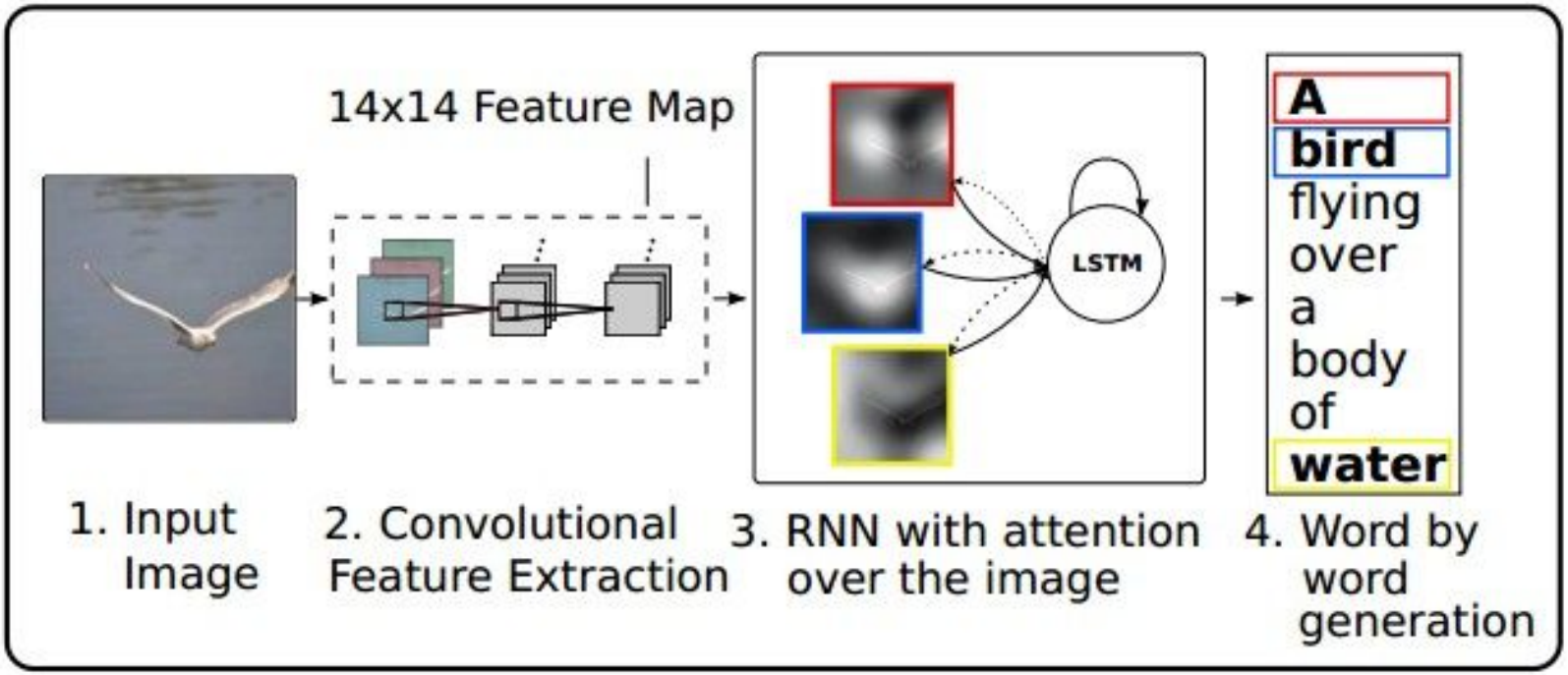
A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

A man in a baseball uniform throwing a ball

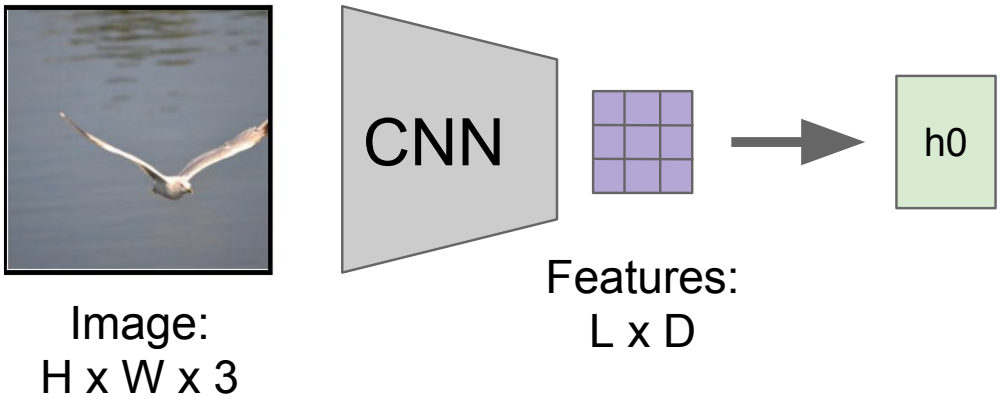
Image Captioning with Attention

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



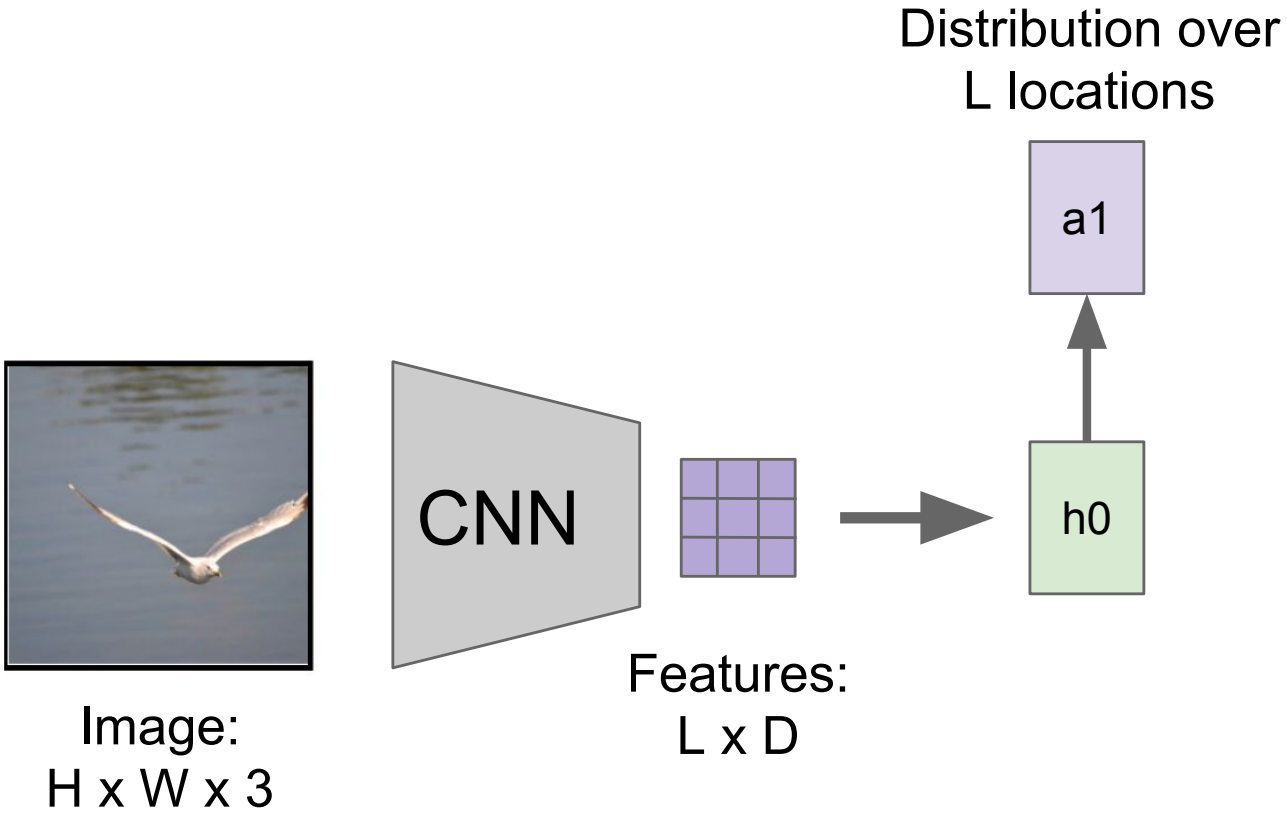
Xu et al, “Show, Attend, and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Image Captioning with Attention



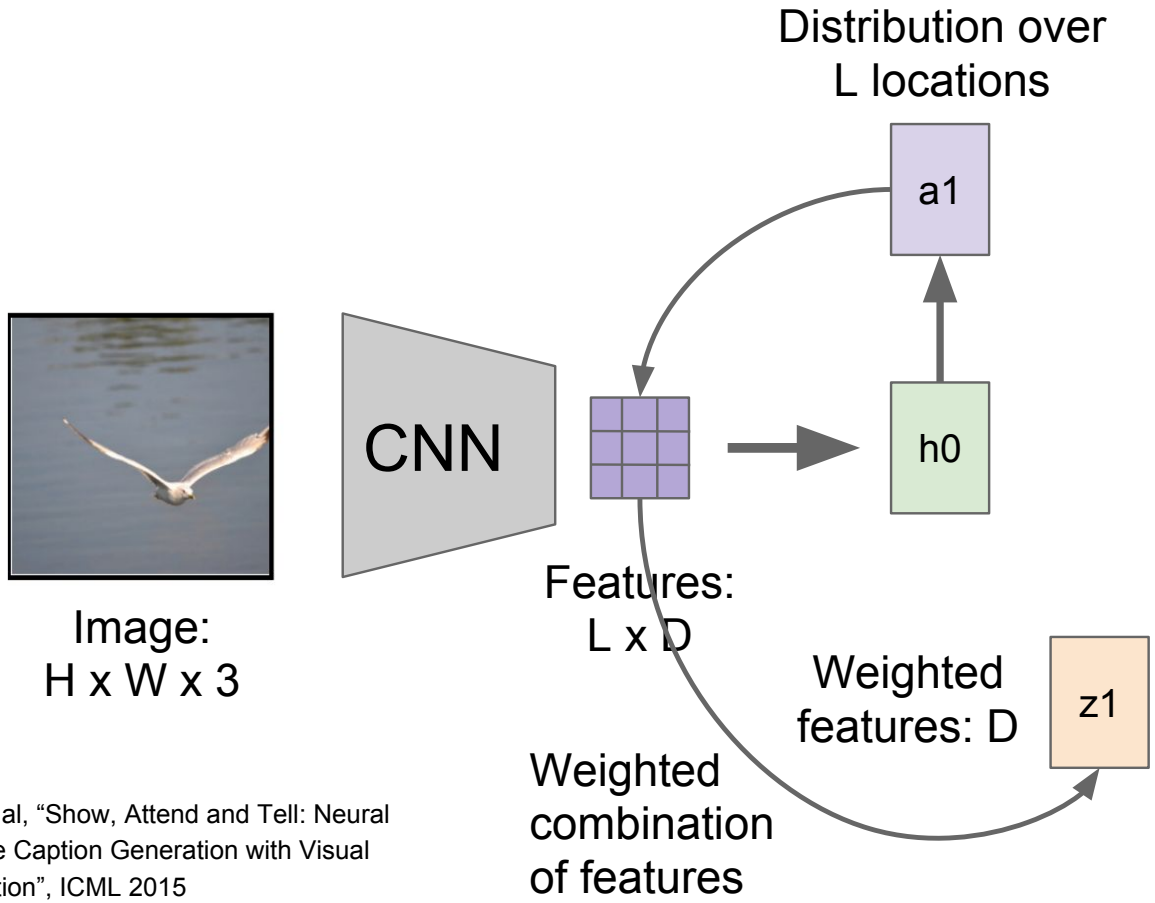
Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Image Captioning with Attention



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

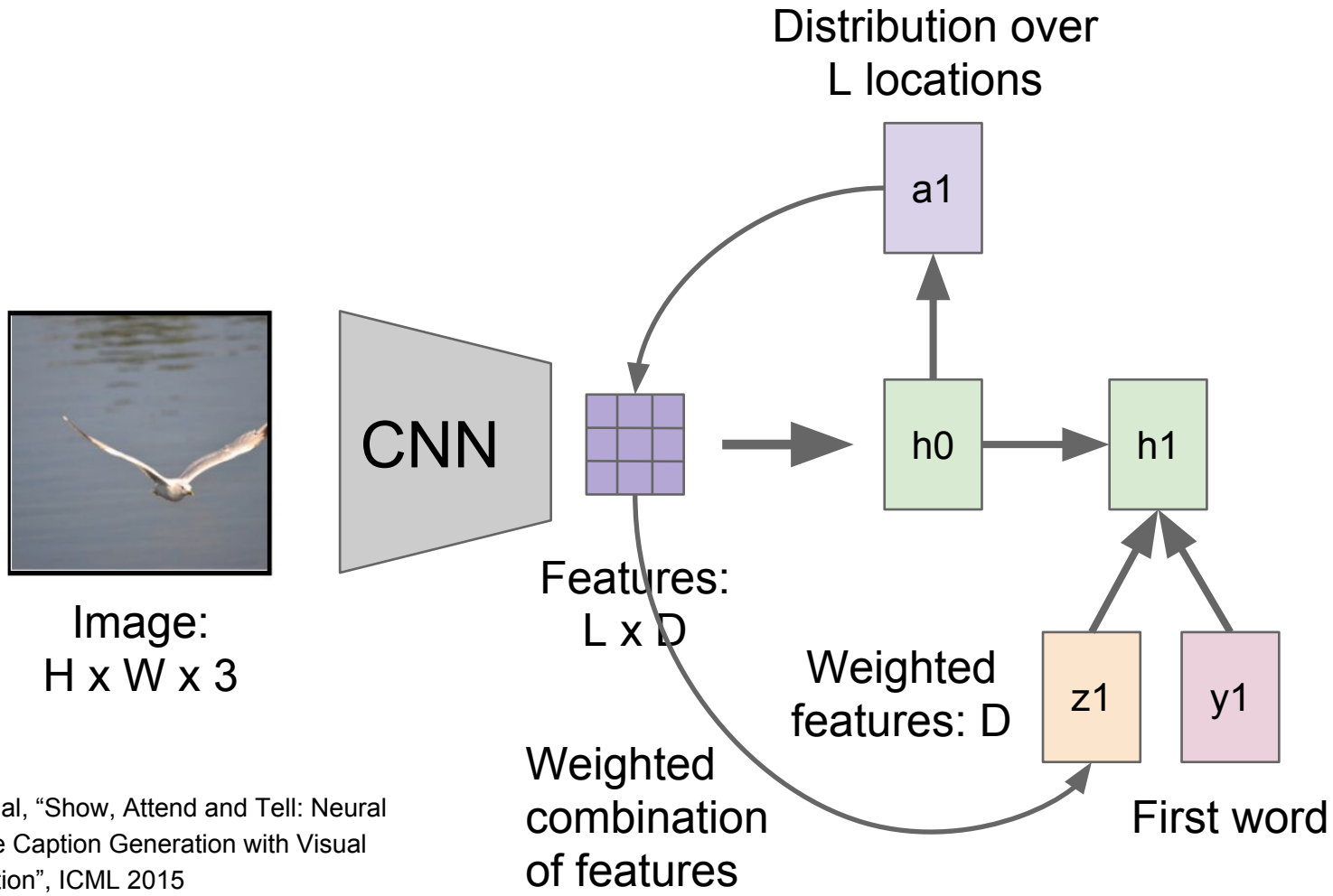
Image Captioning with Attention



$$z = \sum_{i=1}^{L} p_i v_i$$

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

Image Captioning with Attention



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

Image Captioning with Attention

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

Distribution
over vocab

a1

a2

d1

h0

h1

Weighted
features: D

z1

y1

Weighted
combination
of features

First word

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

Image Captioning with Attention

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

Distribution
over vocab

a1

a2

d1

h0

h1

h2

z1

y1

z2

y2

Weighted
features: D

Weighted
combination
of features

First word

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

Image Captioning with Attention

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

Distribution
over vocab

a1

a2

d1

a3

d2

h0

h1

h2

z1

y1

z2

y2

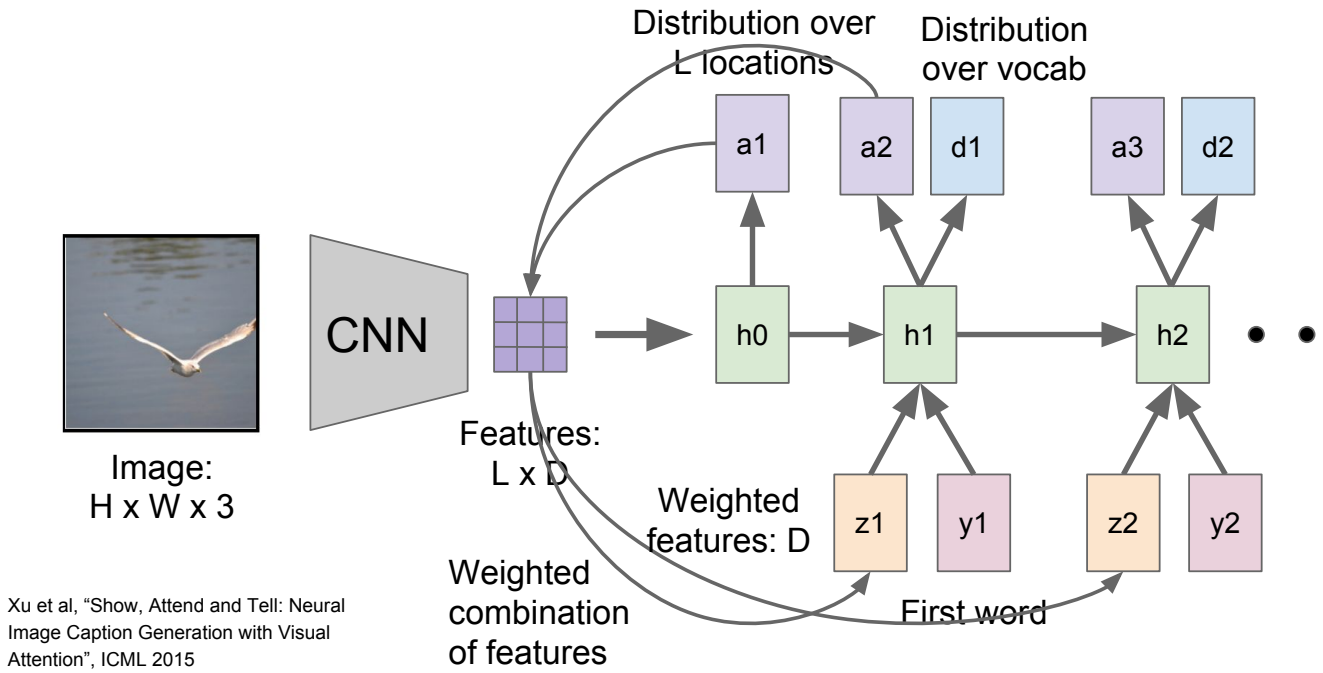
Weighted
features: D

Weighted
combination
of features

First word

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

Image Captioning with Attention



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

$$\begin{bmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} \mathbf{T} \begin{bmatrix} \mathbf{E}\,\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \mathbf{z}_t \end{bmatrix}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t)$$

Image Captioning with Attention

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.

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

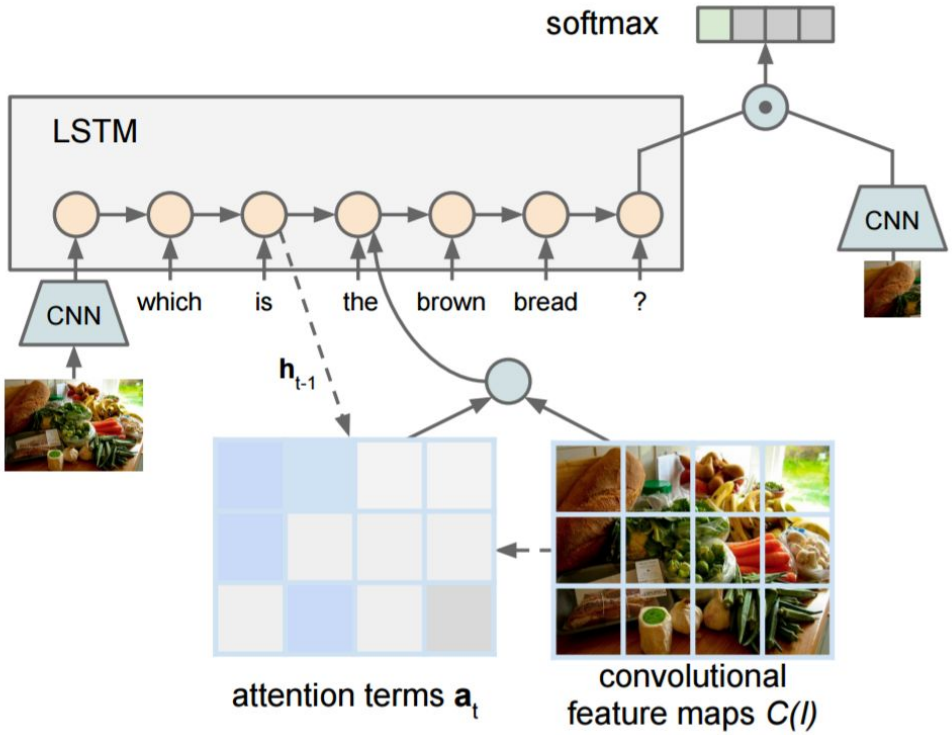
Q: Who is under the umbrella?

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

- The model first reads the image and all the question tokens until reaching the end token of the question sequence.
- In training, continue feeding the ground-truth answer tokens.
- Compute the log-likelihood of an candidate region by a dot product between its transformed visual feature from CNN and the last LSTM hidden state.



What kind of animal is in the photo?
A **cat**.

Why is the person holding a knife?
To cut the **cake** with.

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

Extending RNNs: Multilayer RNNs

$$\mathbf{h} \in \mathbb{R}^n$$

Vanilla RNN: $\mathbf{W}^{(\ell)} \in \mathbb{R}^{n \times 2n}$

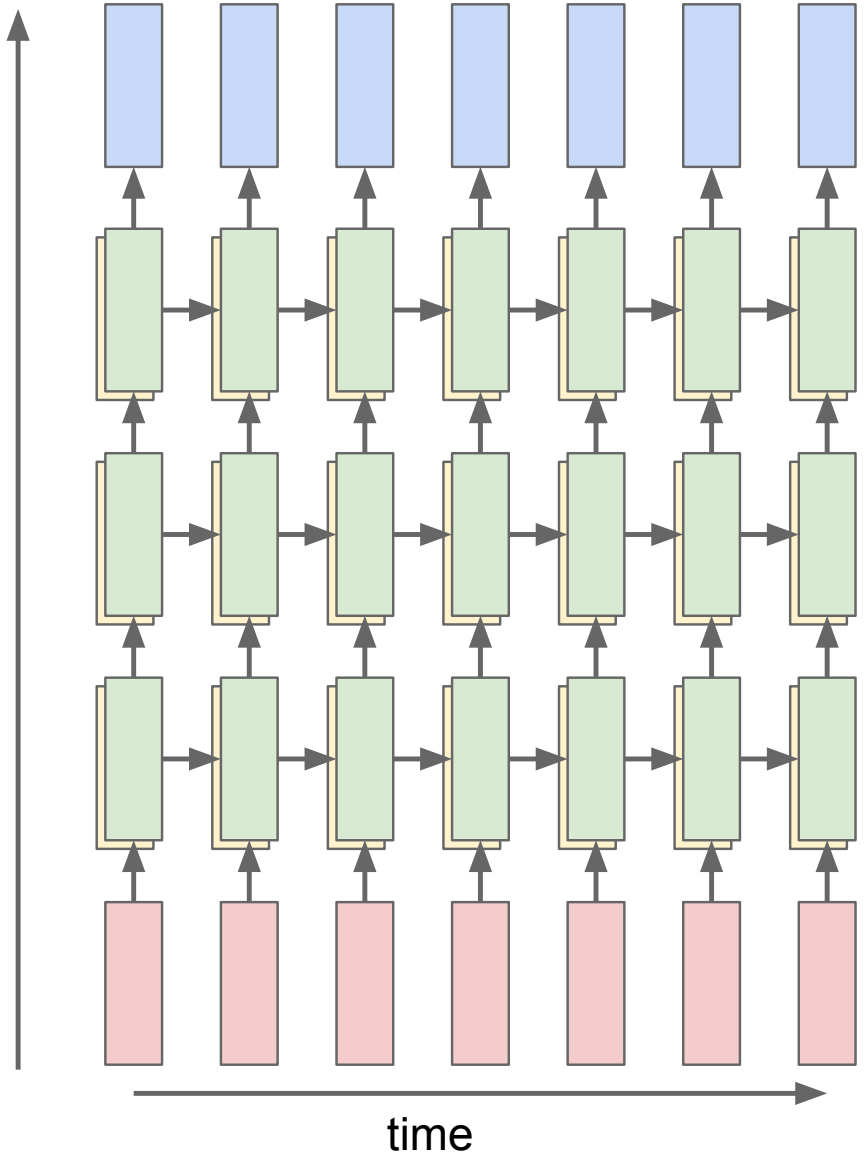
$$\mathbf{h}_t^{(\ell)} = \tanh \mathbf{W}^{(\ell)} \begin{bmatrix} \mathbf{h}_t^{(\ell-1)} \\ \mathbf{h}_{t-1}^{(\ell)} \end{bmatrix}$$

LSTM: $\mathbf{W}^{(\ell)} \in \mathbb{R}^{4n \times 2n}$

$$\begin{pmatrix} \mathbf{i}^{(\ell)} \\ \mathbf{f}^{(\ell)} \\ \mathbf{o}^{(\ell)} \\ \mathbf{g}^{(\ell)} \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} \left[\mathbf{W}^{(\ell)} \begin{pmatrix} \mathbf{h}_t^{(\ell-1)} \\ \mathbf{h}_{t-1}^{(\ell)} \end{pmatrix} \right.$$

$$\mathbf{c}_t^{(\ell)} = \mathbf{f}^{(\ell)} \odot \mathbf{c}_{t-1}^{(\ell)} + \mathbf{i}^{(\ell)} \odot \mathbf{g}^{(\ell)}$$

$$\mathbf{h}_t^{(\ell)} = \mathbf{o}^{(\ell)} \odot \tanh(\mathbf{c}_t^{(\ell)})$$



Google Translation System