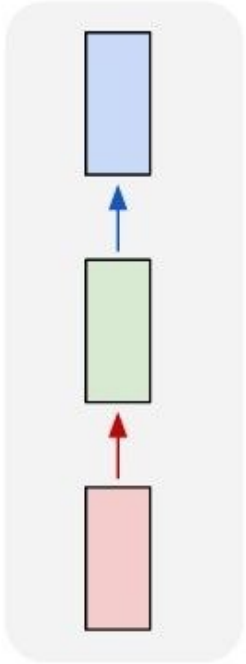
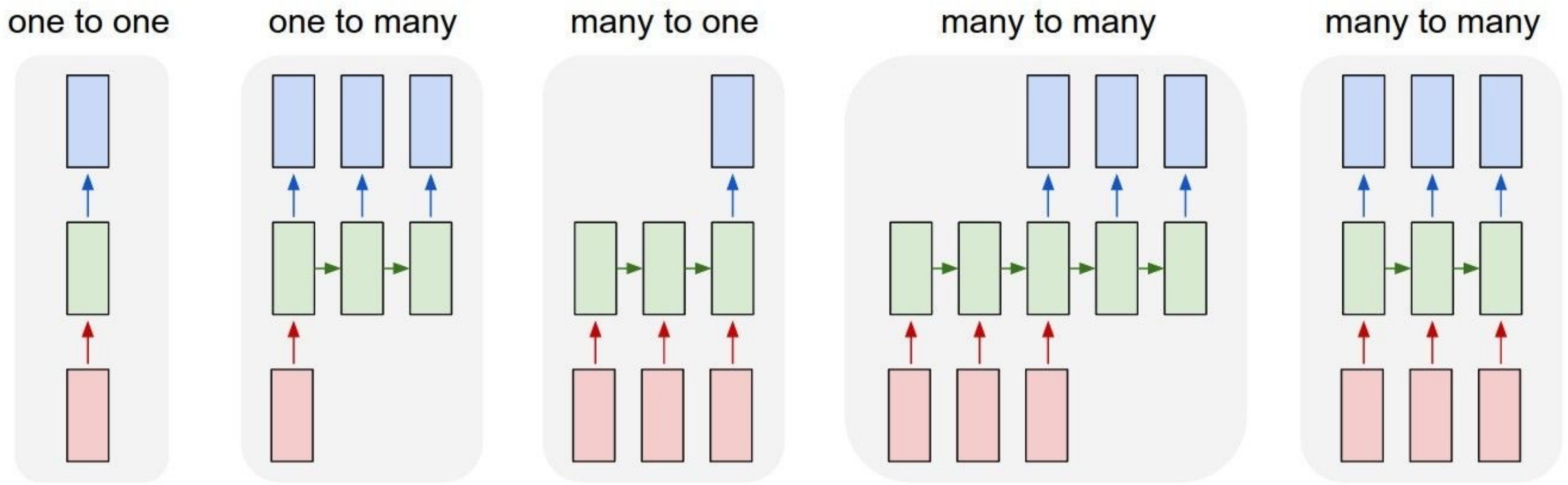
Sequence Modelling and Forecasting

Seen so far: “Vanilla” Neural Network

one to one



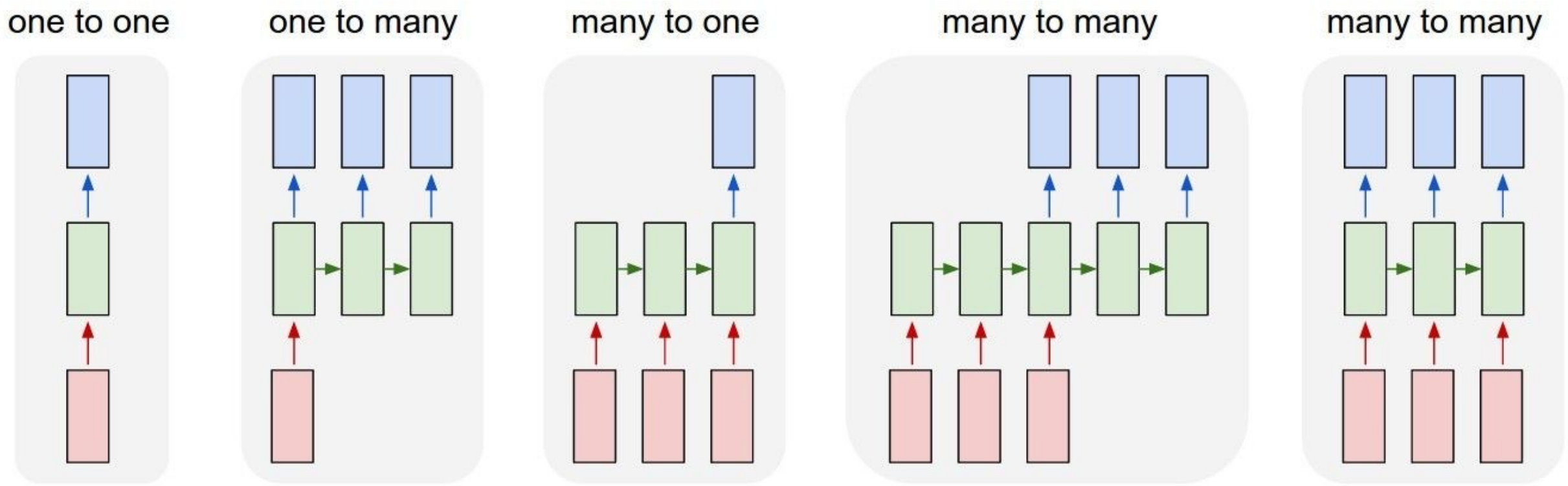
Process Sequences



Vanilla Neural Networks

Sequence tasks
captioning, video classification, forecasting, …

Process Sequences

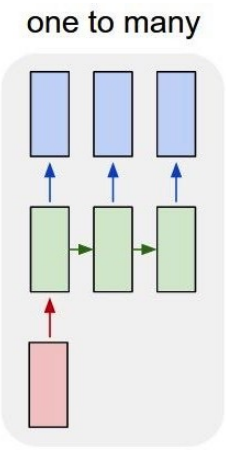


e.g. **Image Captioning**
image -> sequence of words

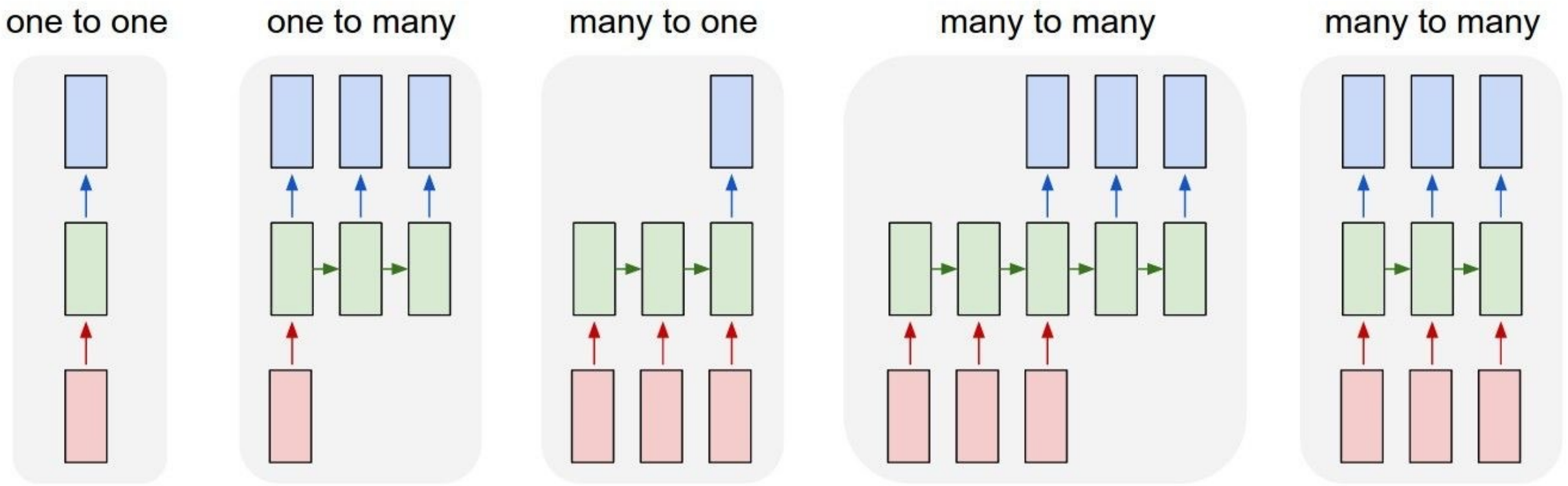
Process Sequences

Sequences in the output…
(one-to-many)

A happy brown dog.



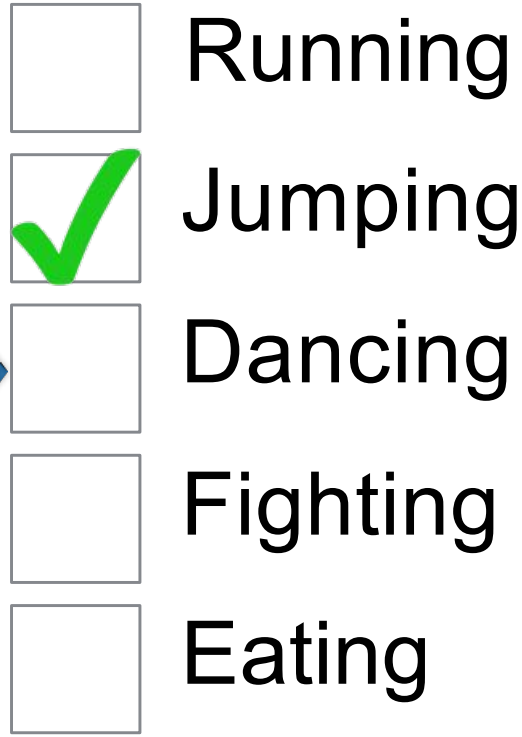
Process Sequences

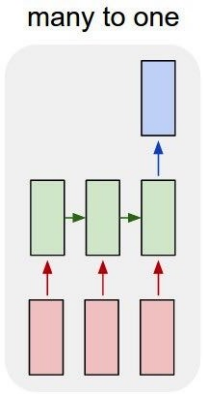


e.g. **Sentiment Classification**
sequence of words -> sentiment

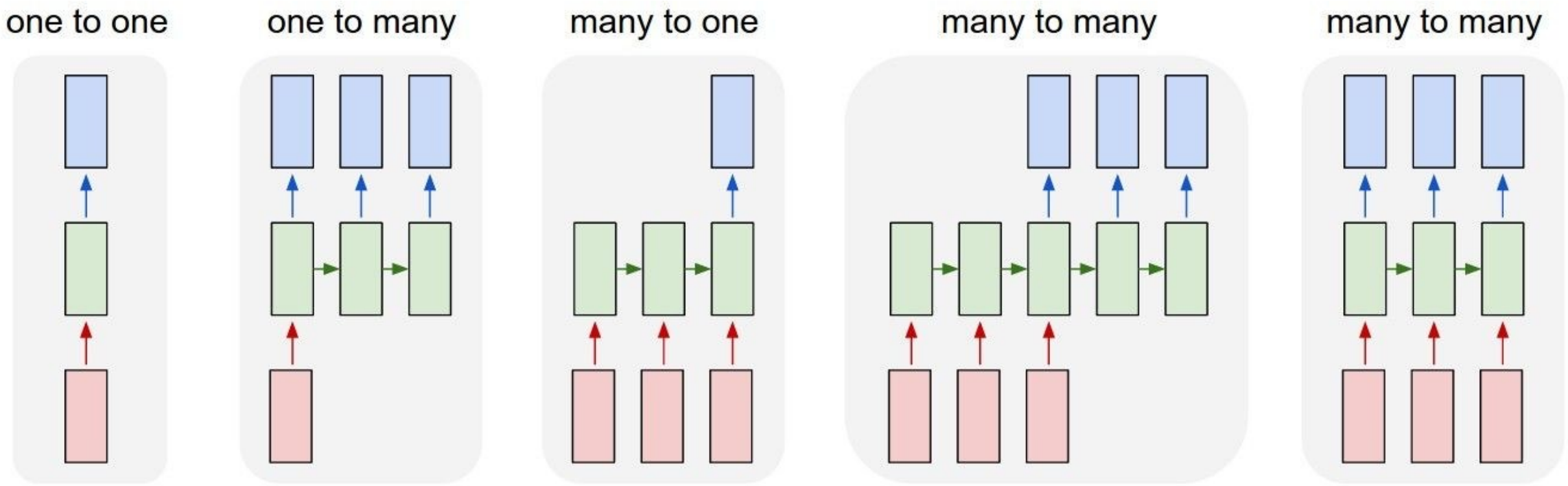
Process Sequences

Sequences in the input…
(many-to-one)





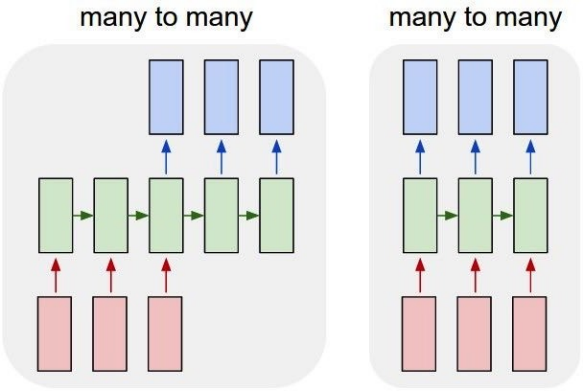
Process Sequences



e.g. **Machine Translation**
seq of words -> seq of words

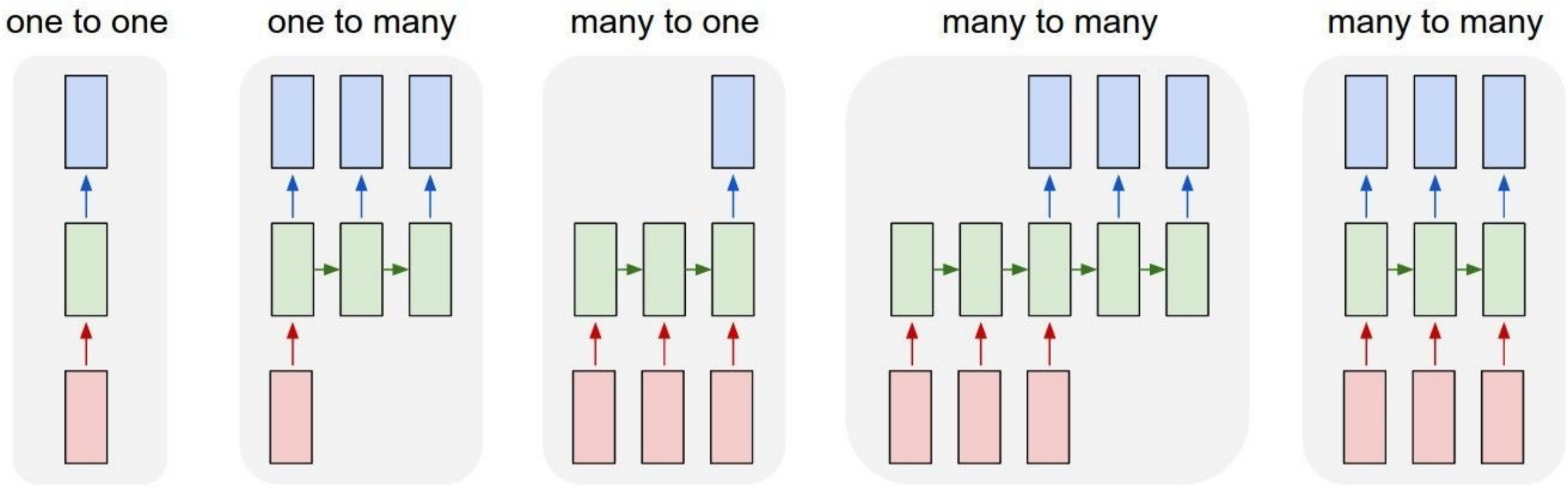
Process Sequences

Sequences everywhere!
(many-to-many)



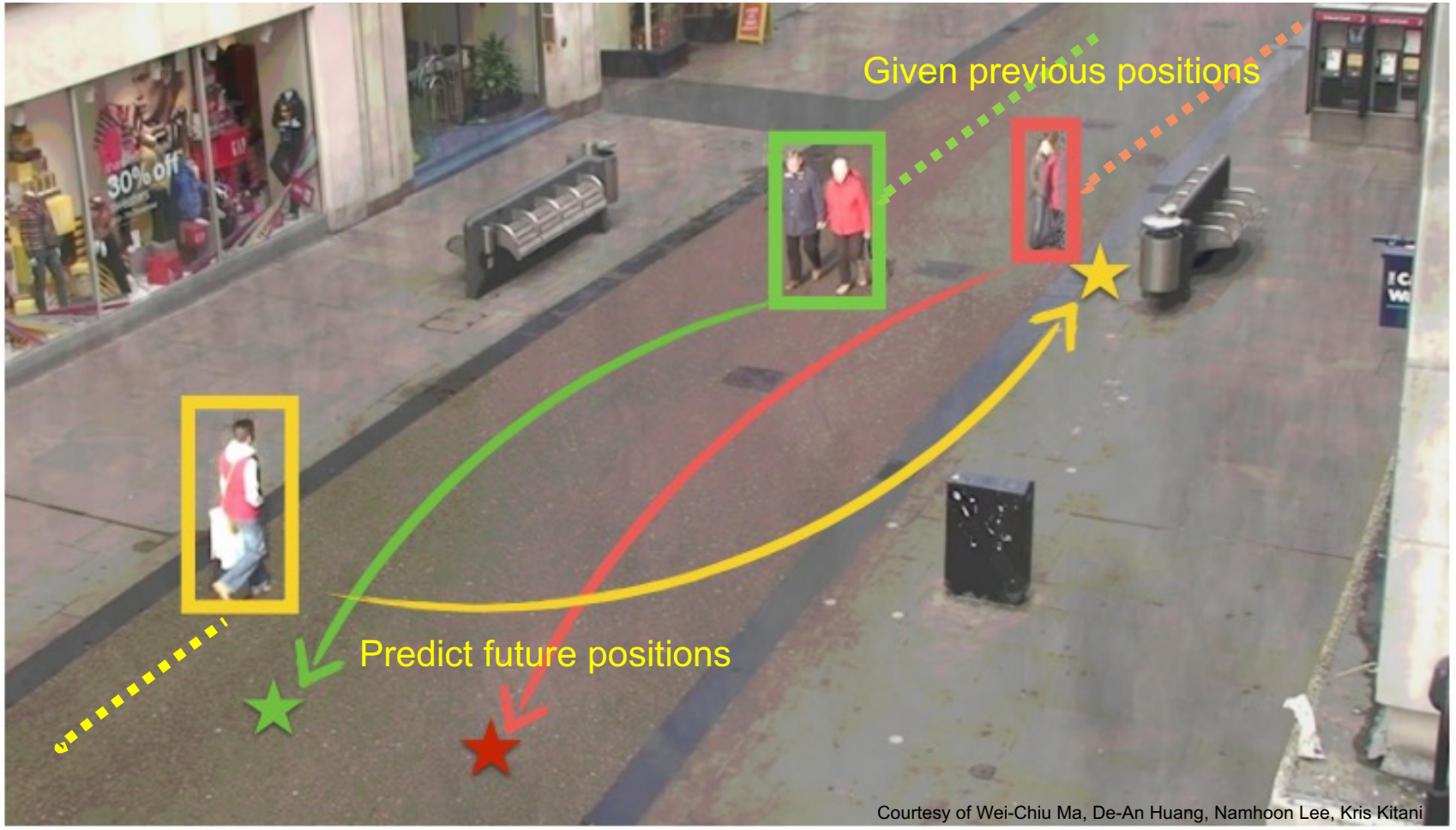
A dog jumps over a hurdle.

Process Sequences



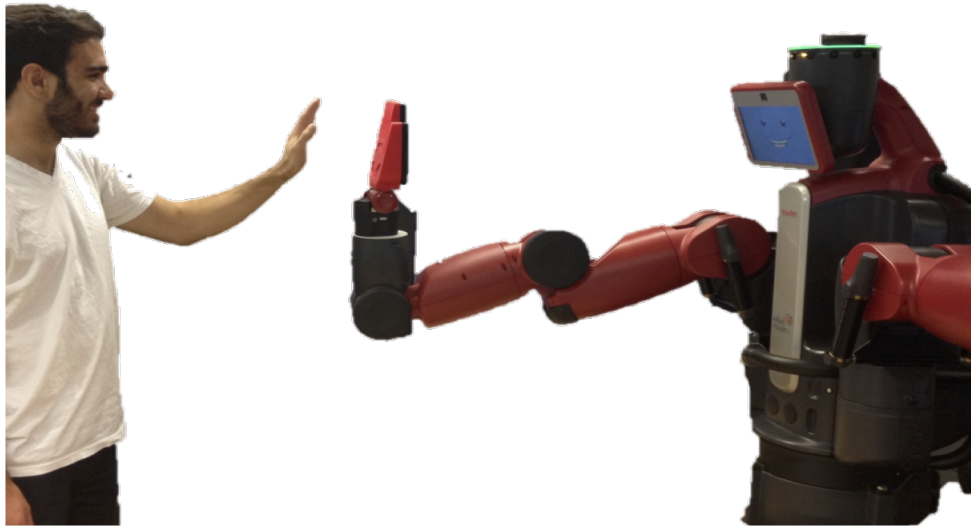
e.g. **Video classification on frame level**

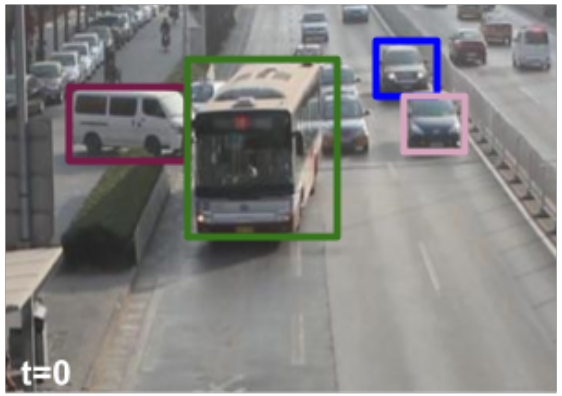
Trajectory forecasting

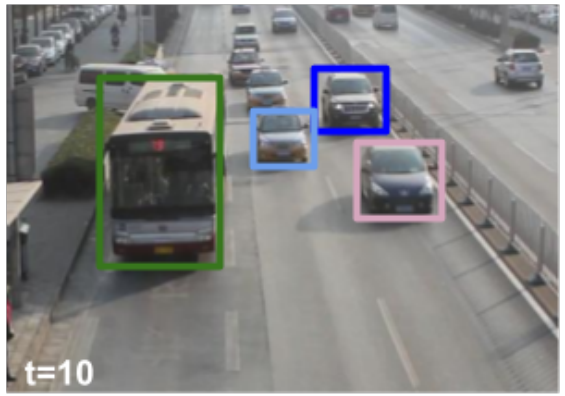


Forecasting

- Human-robot interaction
- Long-term tracking across occlusions
- Safety of autonomous vehicles







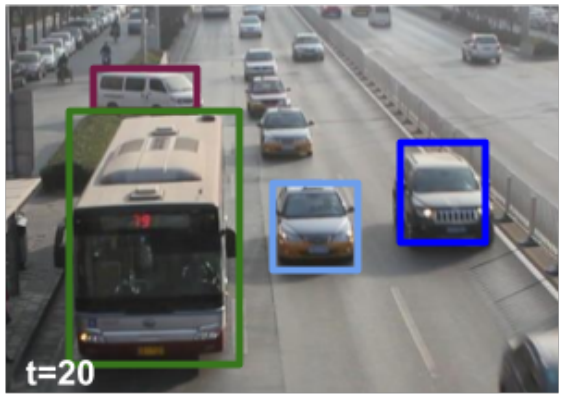
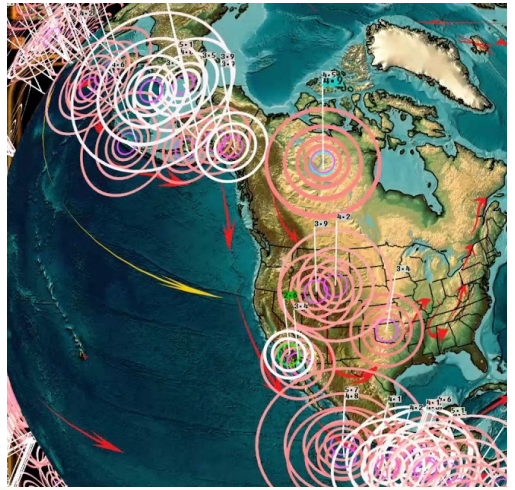


Image sources: https://www.grasp.upenn.edu/projects/haptics-human-robot-interaction
http://www.woostercollective.com/post/3d-optical-illusion-painted-on-street-to-make-drivers-slow-down

Also with Forecasting

- Weather and natural disasters, e.g. earthquakes
- Market trends
- Critical pressure of a power plant
- Gas supply and demand
- …



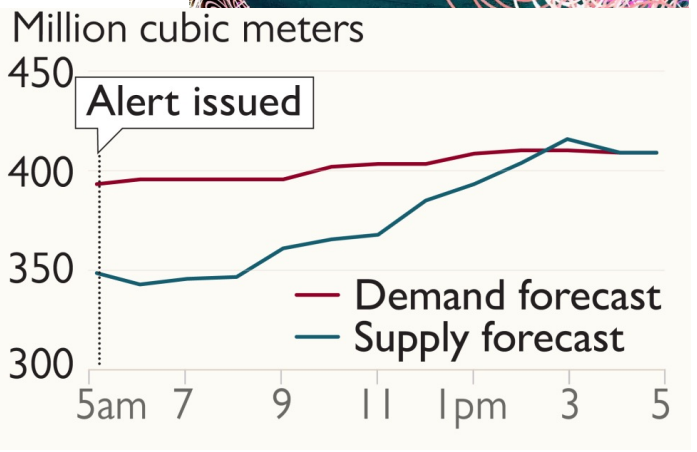
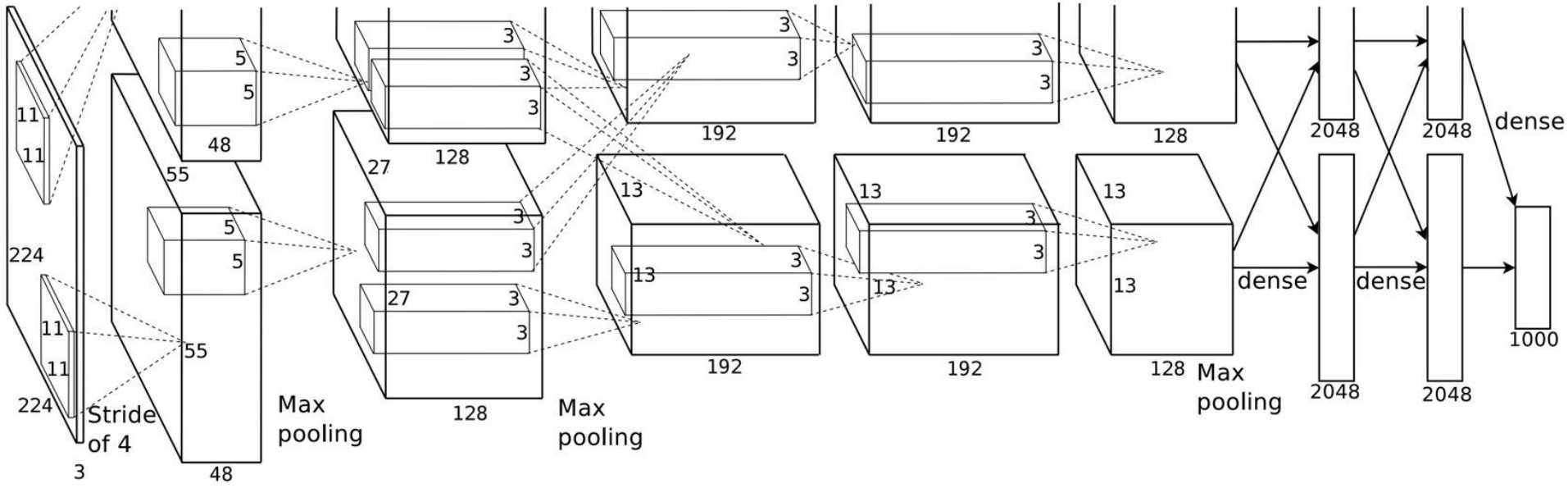


Image sources: http://www.vestinet.rs/pogledi/ozbiljno-upozorenje-naucnika-u-2018-ocekujte-razorne-zemljotrese-evo-gde-ce-biti-najgore-video
https://allstarcharts.com/monthlysjune2018/, https://www.thetimes.co.uk/article/national-gas-alert-as-prices-soar-08xjfqc8q

Temporal Convolution Networks

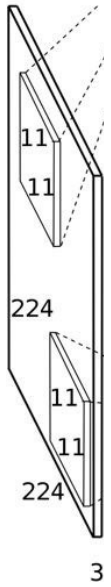
slide credit: Jeff Donahue

ConvNets

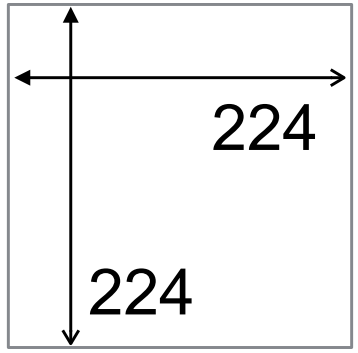


Krizhevsky et al., NIPS 2012

Problem #1

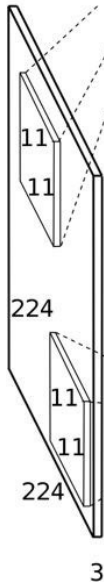


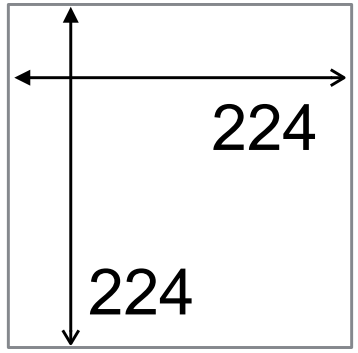
fixed-size, static input



Problem #1

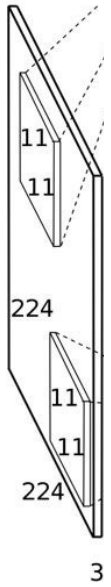
fixed-size, static input



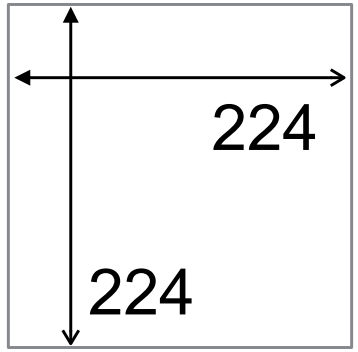


???

Problem #1



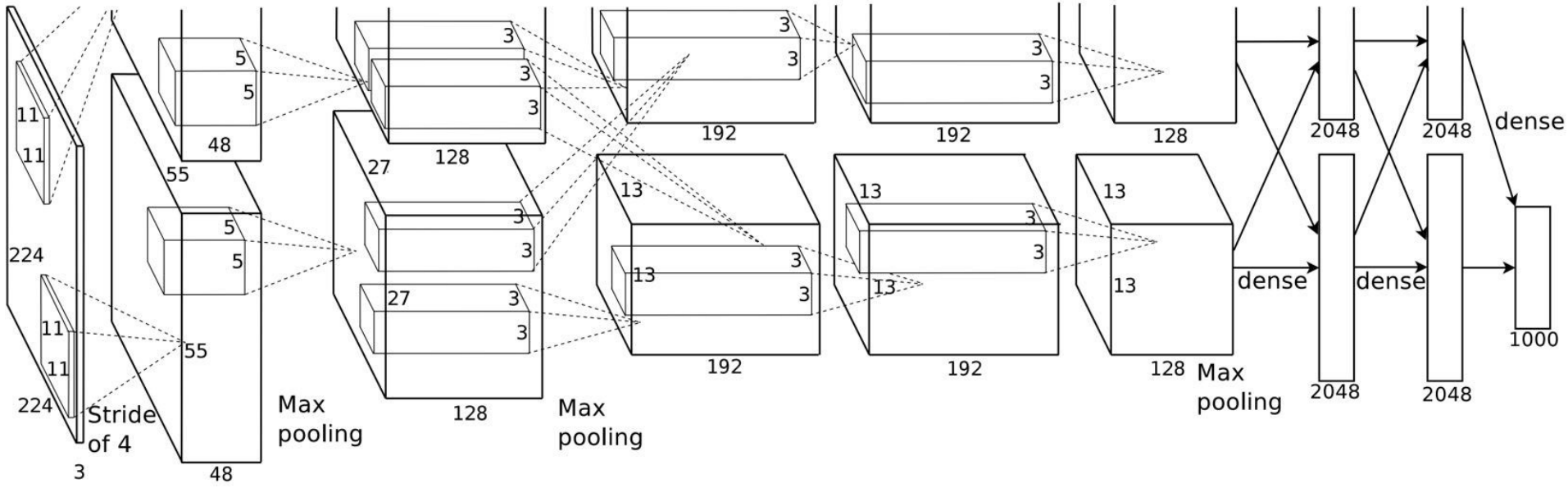
fixed-size, static input



???

However one may concatenate (a fixed number of) frames as input

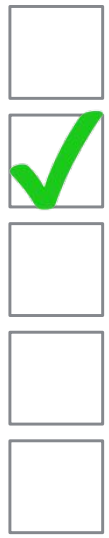
Problem #2



Krizhevsky et al., NIPS 2012

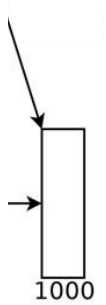
Problem #2

output is a single choice from a fixed list of options



cat dog horse

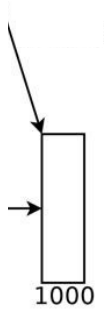
fish snake



Problem #2

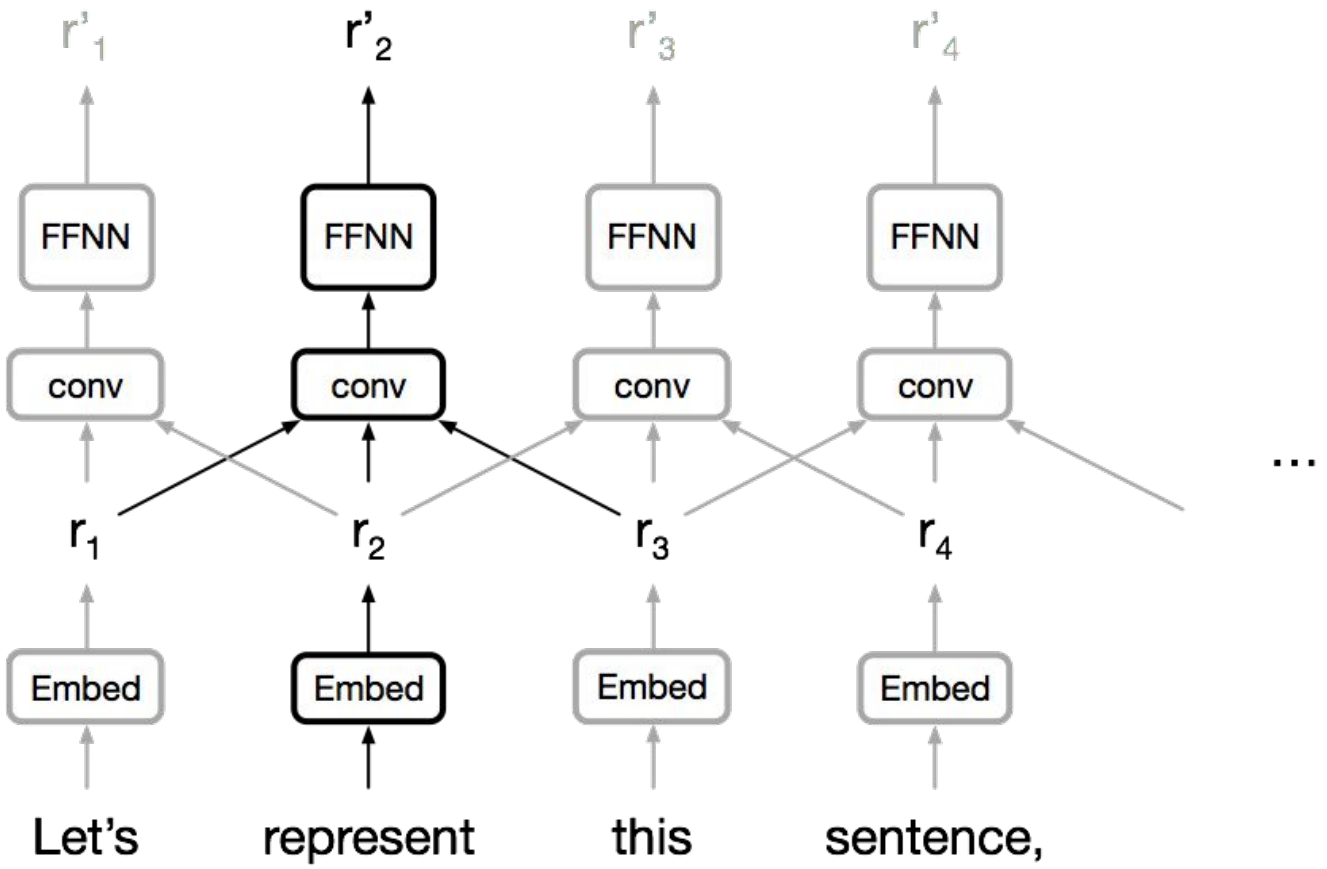
output is a single choice from a fixed list of options

- [ ] a happy brown dog a big
- [x] brown dog
- [ ] a happy red dog a big red
- [ ] dog
- [ ] …



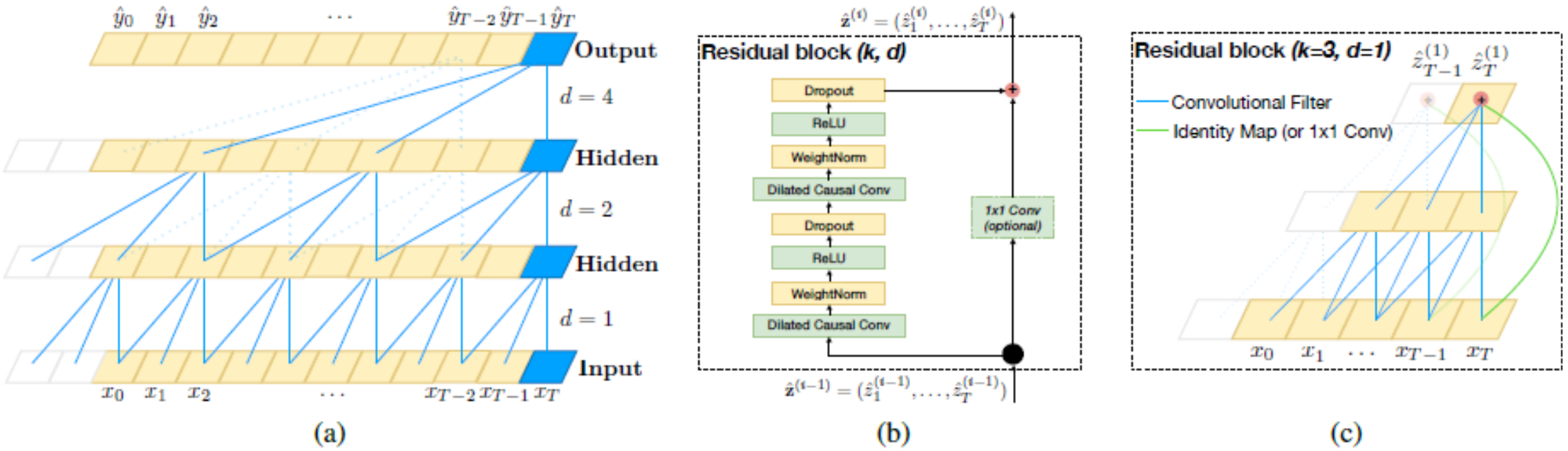
However one may predict one word at the time or a fixed number of output words

Convolutional Neural Networks for Sequences



Temporal Convolutional Networks (TCN)

- Bai et al. ArXiv'18 "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling"
  - Convolutions are causal
  - Dilation: 12 layers may process a history of size 2^12



Convolutional Neural Networks for Sequences

Trivial to parallelize (per layer).

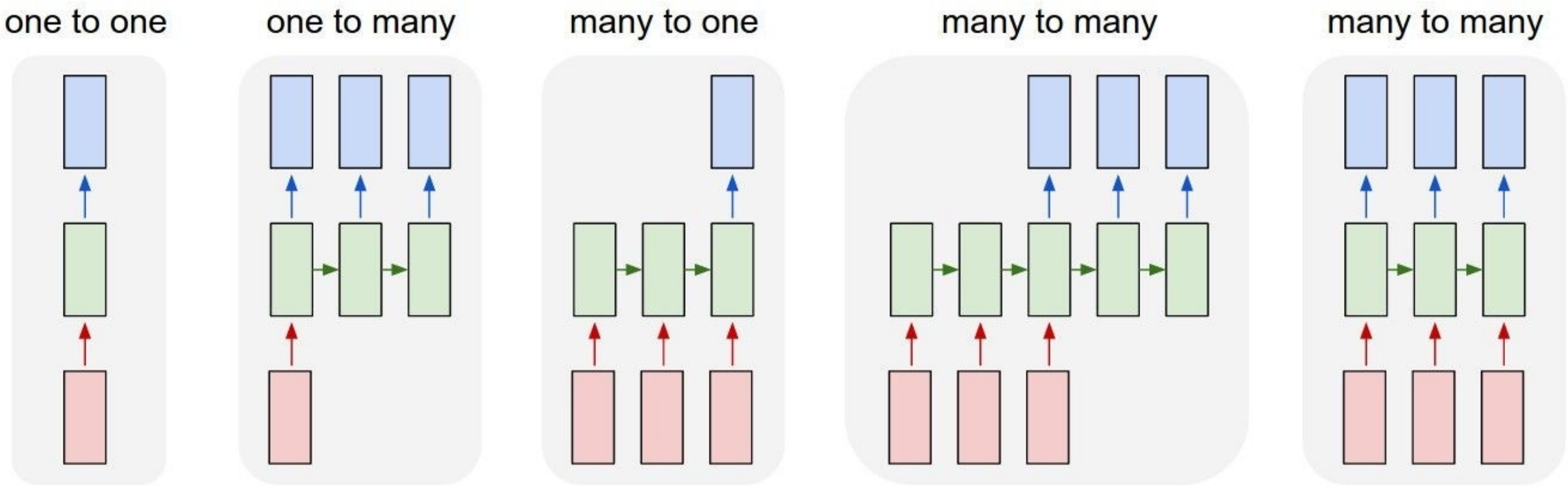
Exploits local dependencies

‘Interaction distance’ between positions linear or logarithmic.

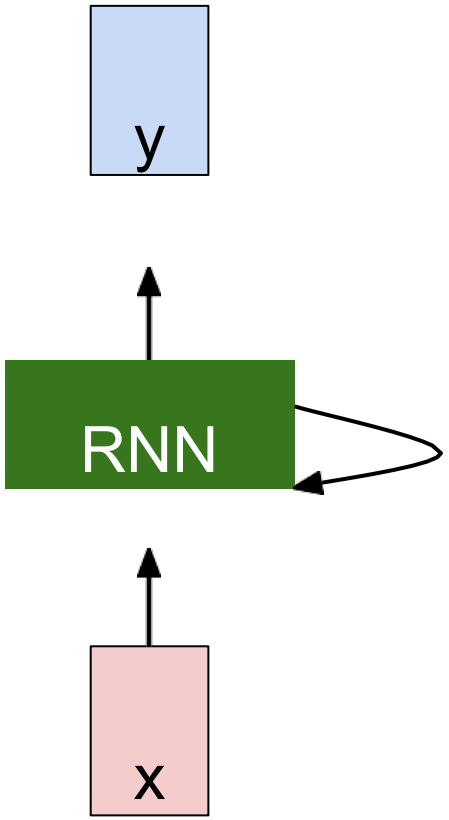
Long-distance dependencies require many layers.

Recurrent Neural Networks

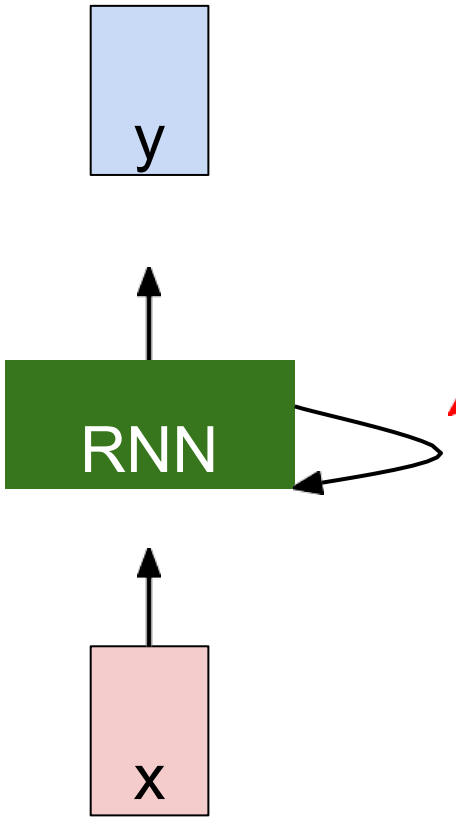
RNNs offer a lot of flexibility on all scenarios



Recurrent Neural Network



Recurrent Neural Network



Key idea: RNNs have an “internal state” that is updated as a sequence is processed

Recurrent Neural Network

We can process a sequence of vectors **x** by applying a **recurrence formula** at every time step:

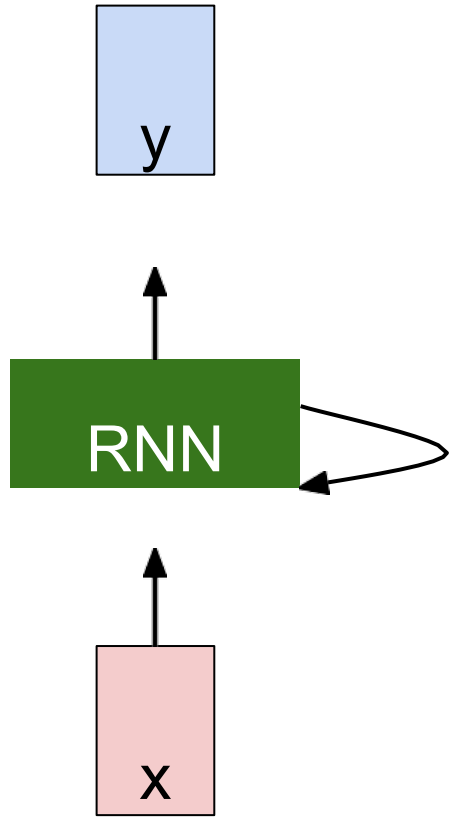
$$h_t = f_W(h_{t-1}, x_t)$$

new state

old state

input vector at some time step

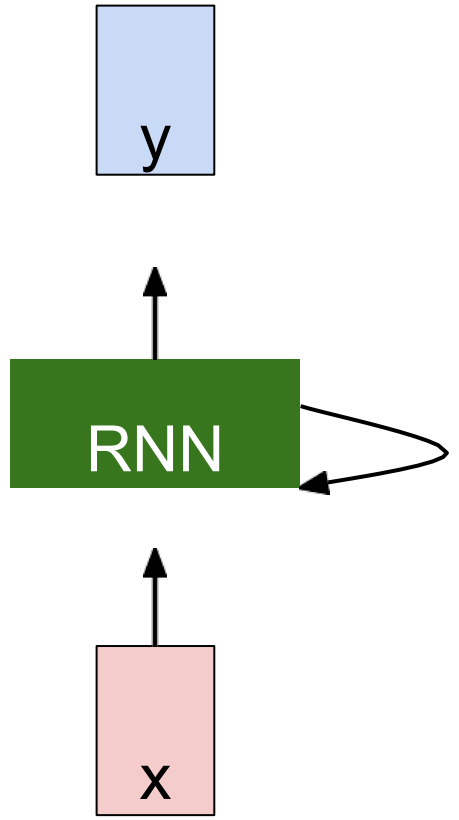
some function with parameters W



Recurrent Neural Network

We can process a sequence of vectors **x** by applying a **recurrence formula** at every time step:

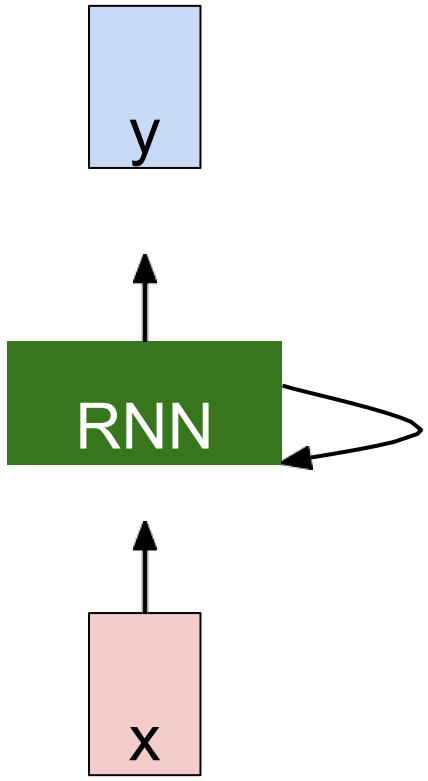
$$h_t = f_W(h_{t-1}, x_t)$$



Notice: the same function and the same set of parameters are used at every time step.

(Simple) Recurrent Neural Network

The state consists of a single *"hidden"* vector **h**:



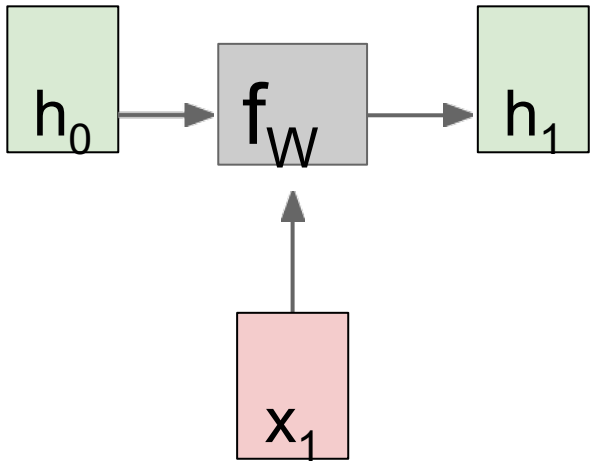
$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

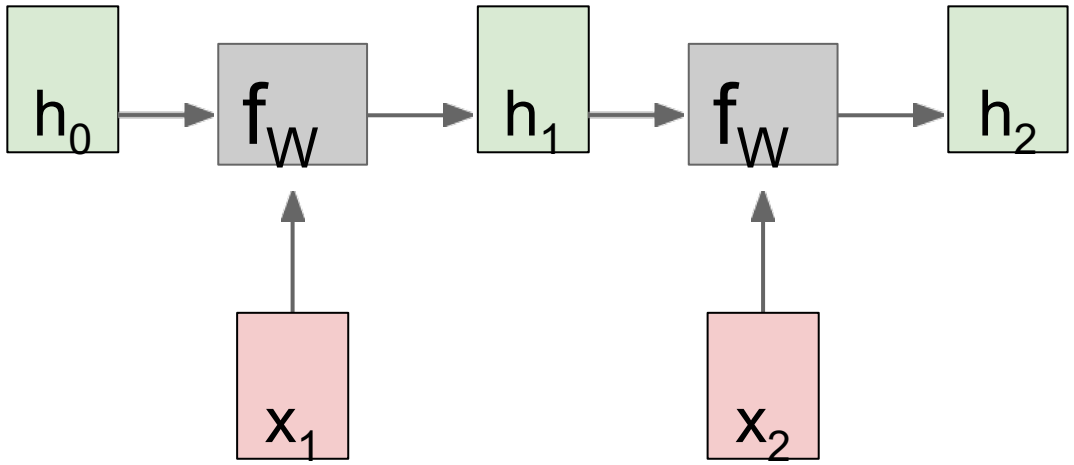
$$y_t = W_{hy}h_t$$

Sometimes called a "Vanilla RNN" or an "Elman RNN" after Prof. Jeffrey Elman

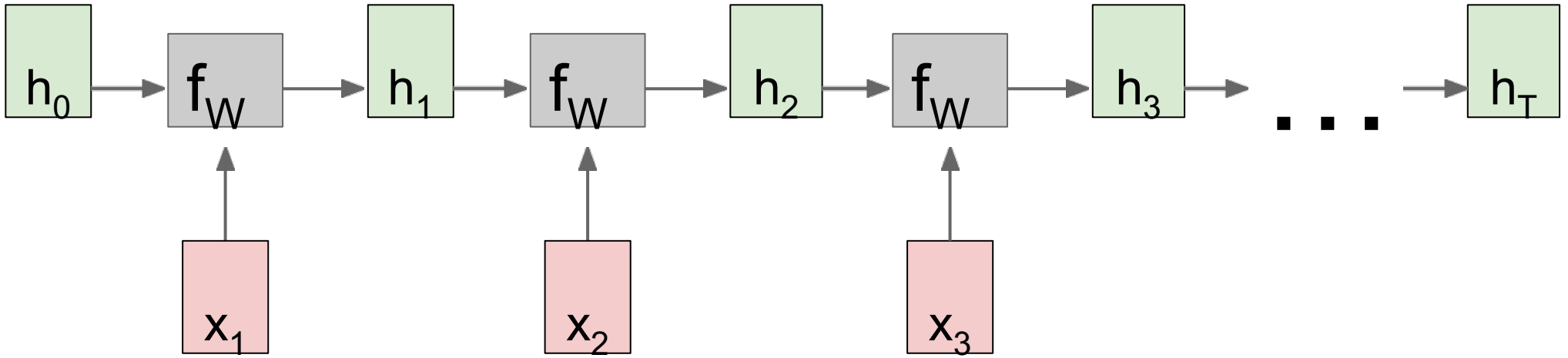
RNN: Computational Graph



RNN: Computational Graph

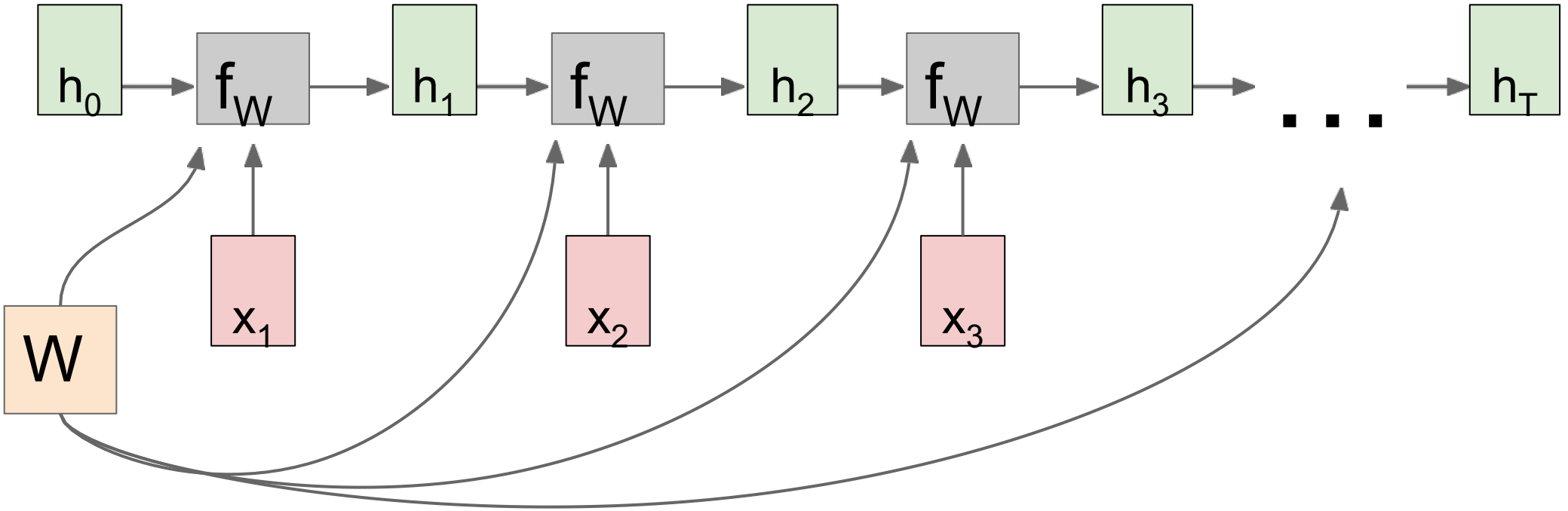


RNN: Computational Graph

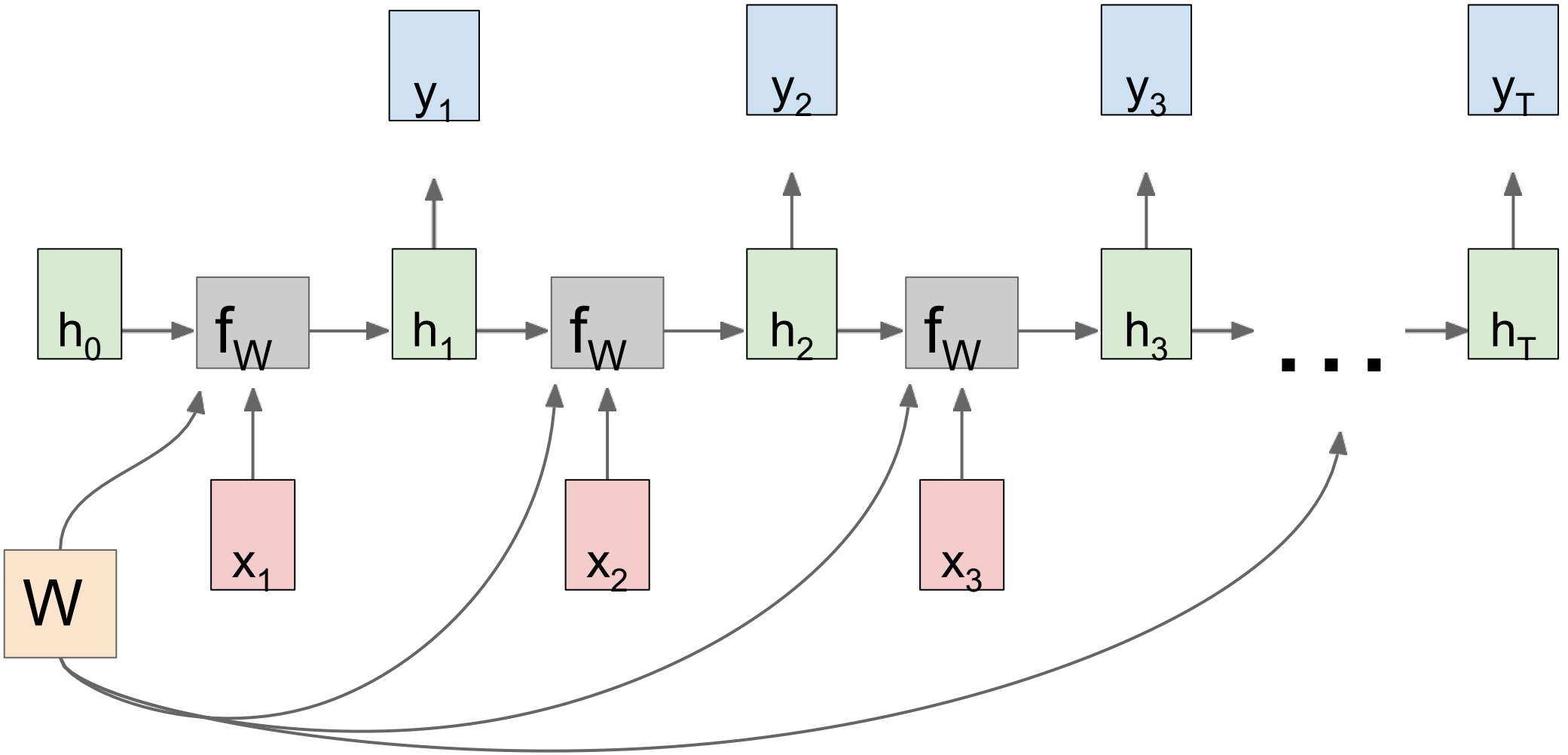


RNN: Computational Graph

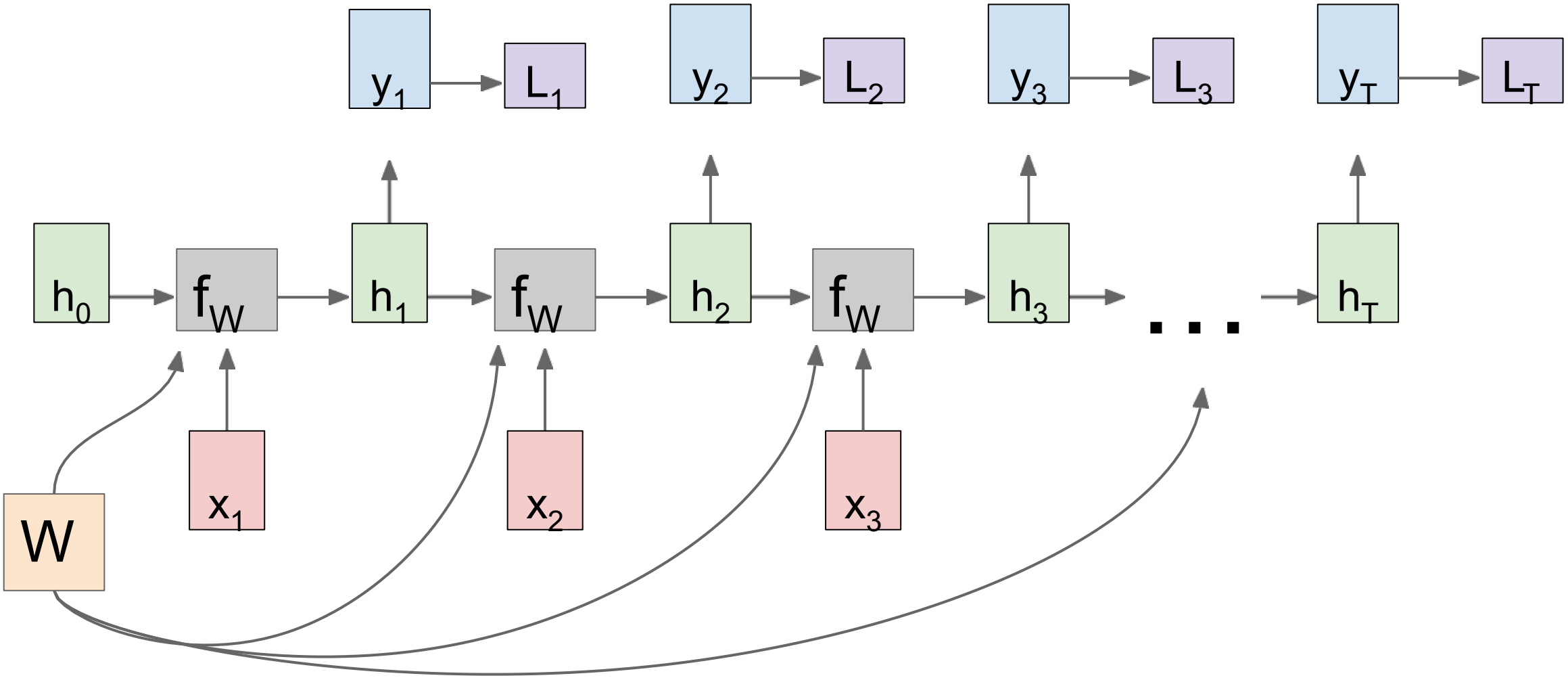
Re-use the same weight matrix at every time-step



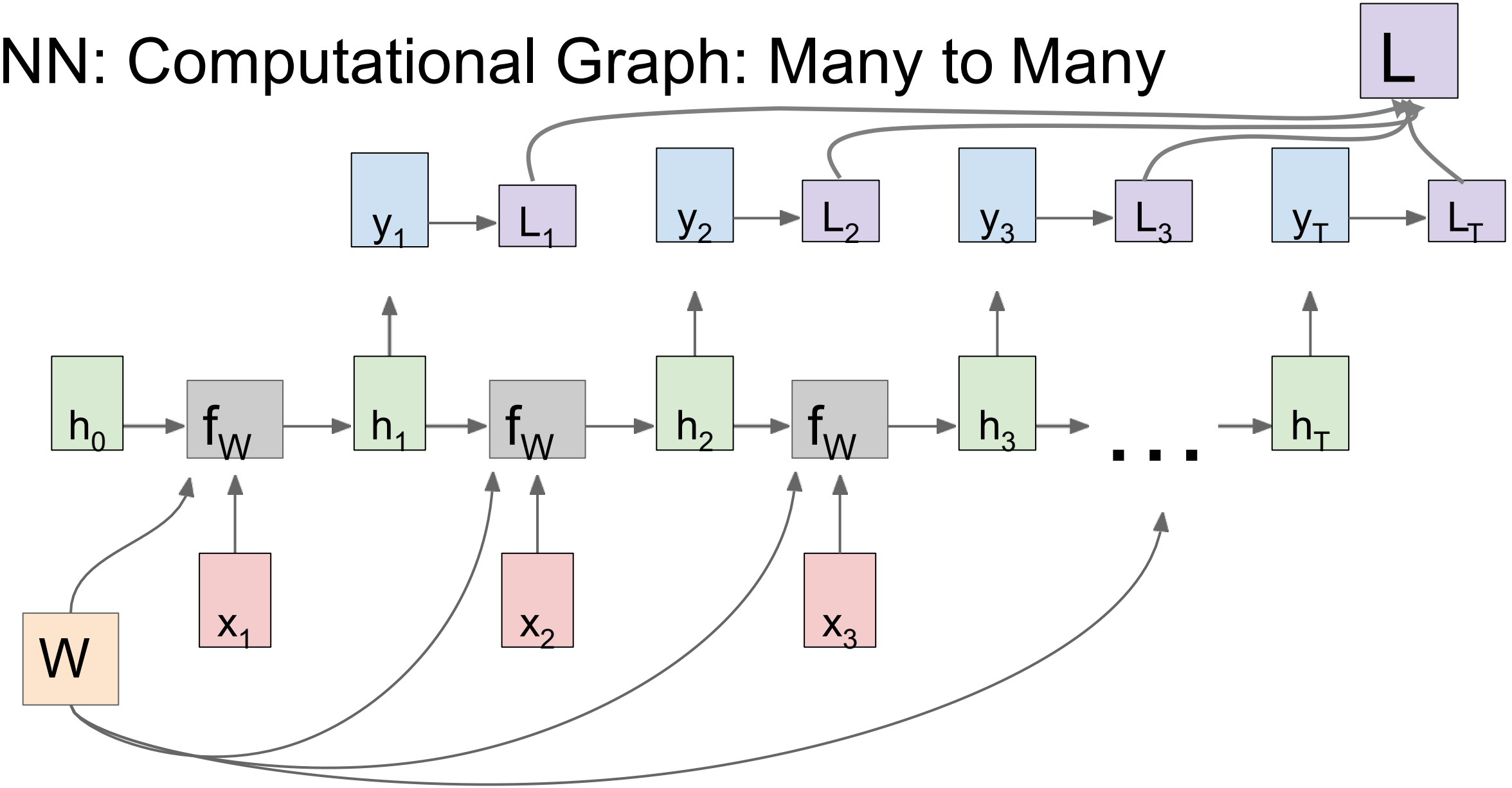
RNN: Computational Graph: Many to Many



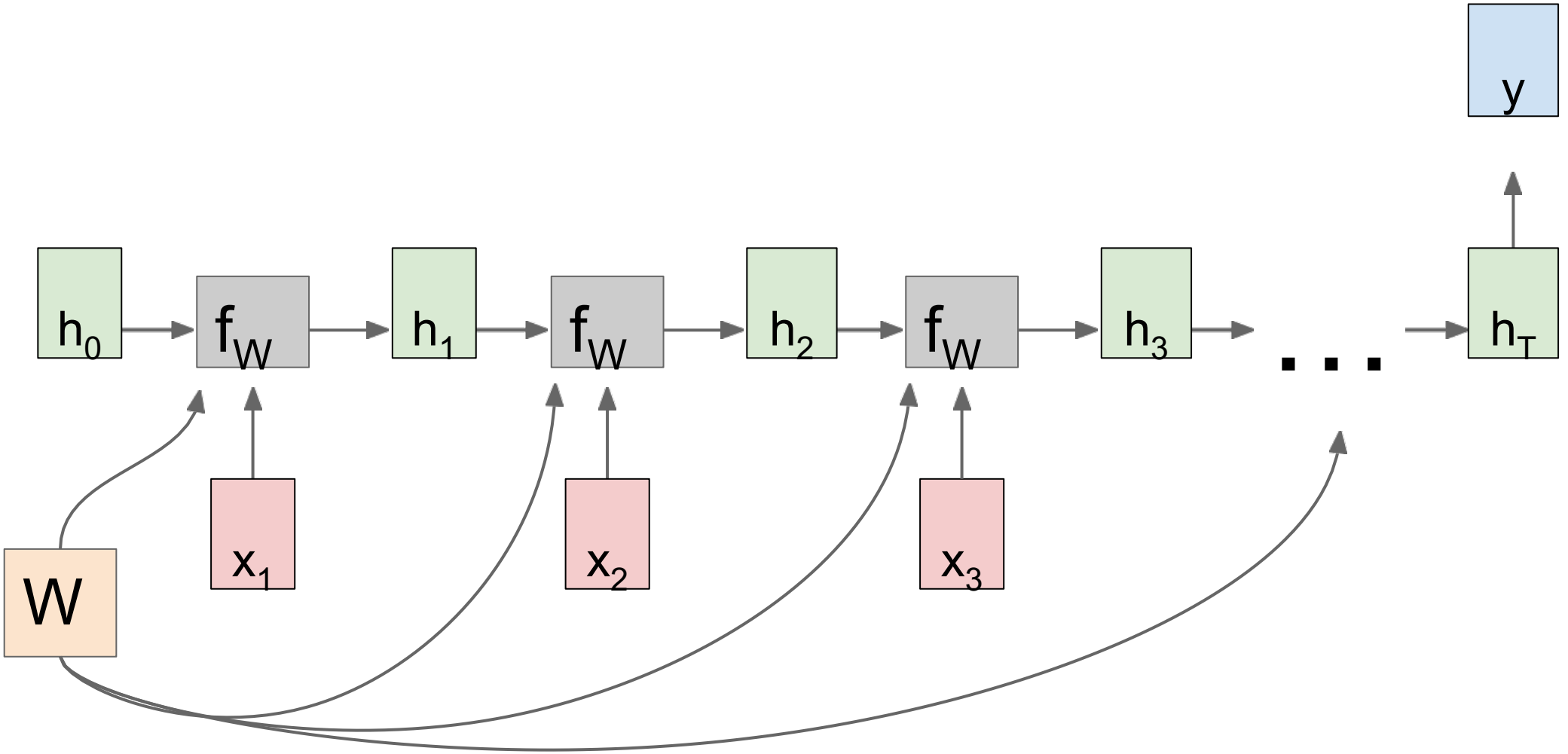
RNN: Computational Graph: Many to Many



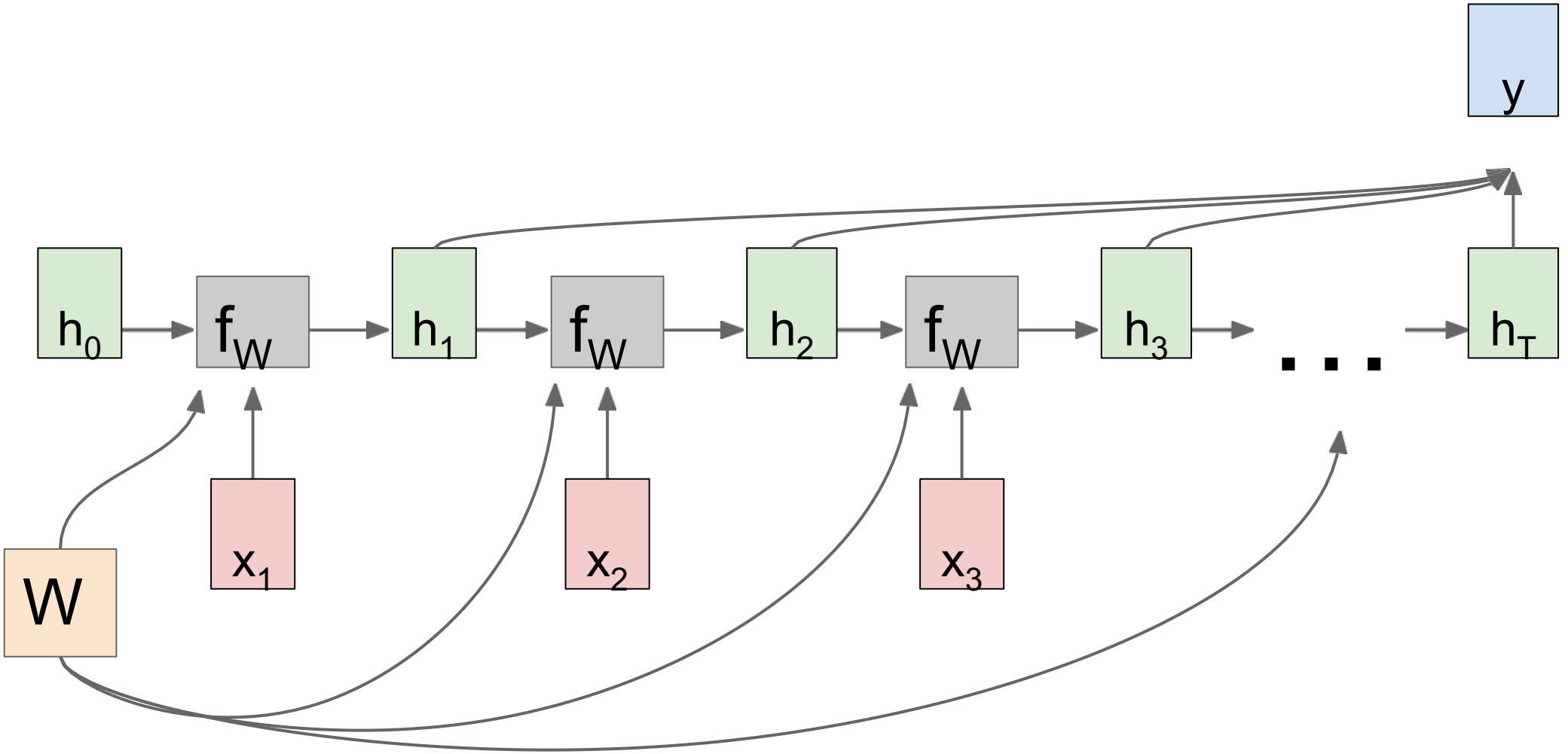
RNN: Computational Graph: Many to Many



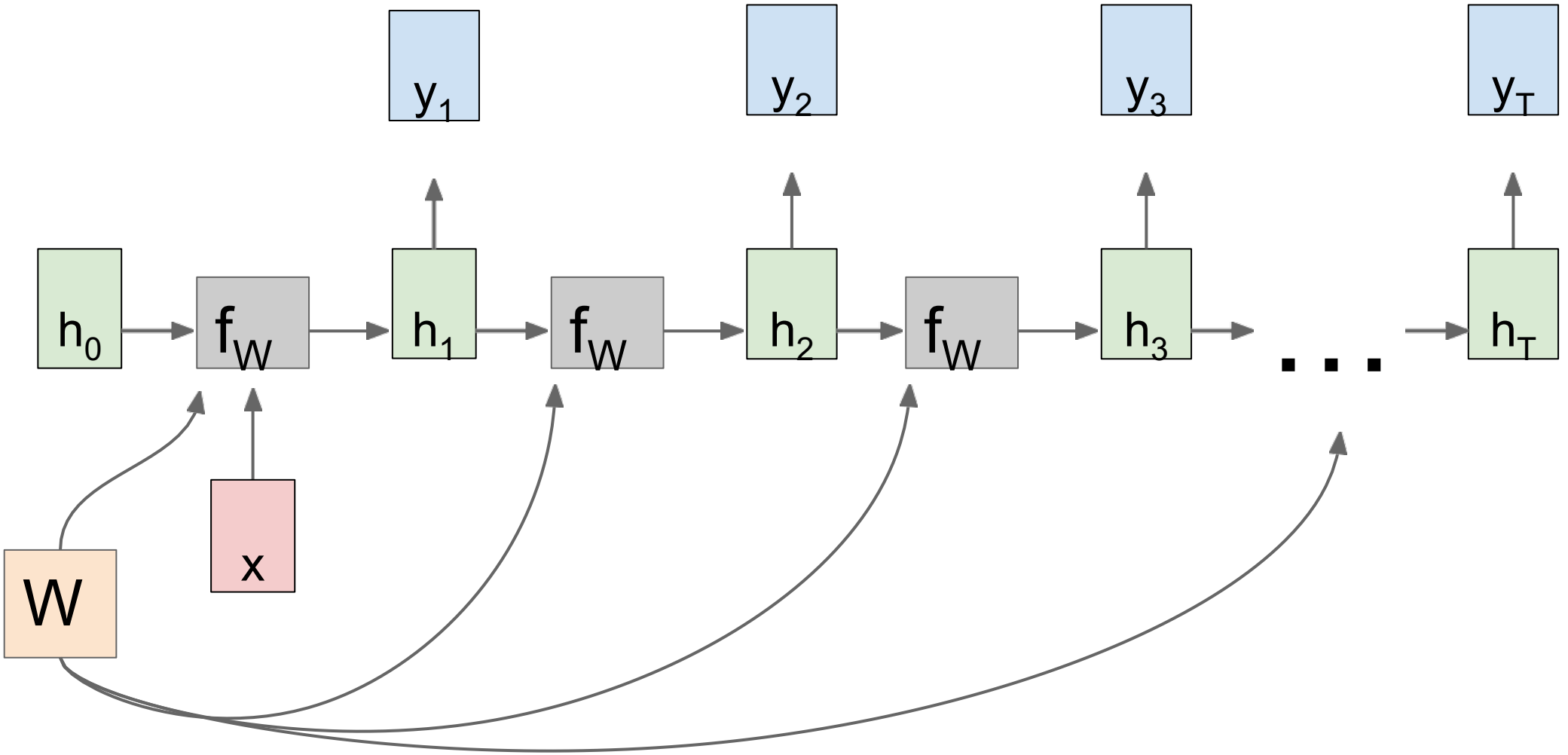
RNN: Computational Graph: Many to One



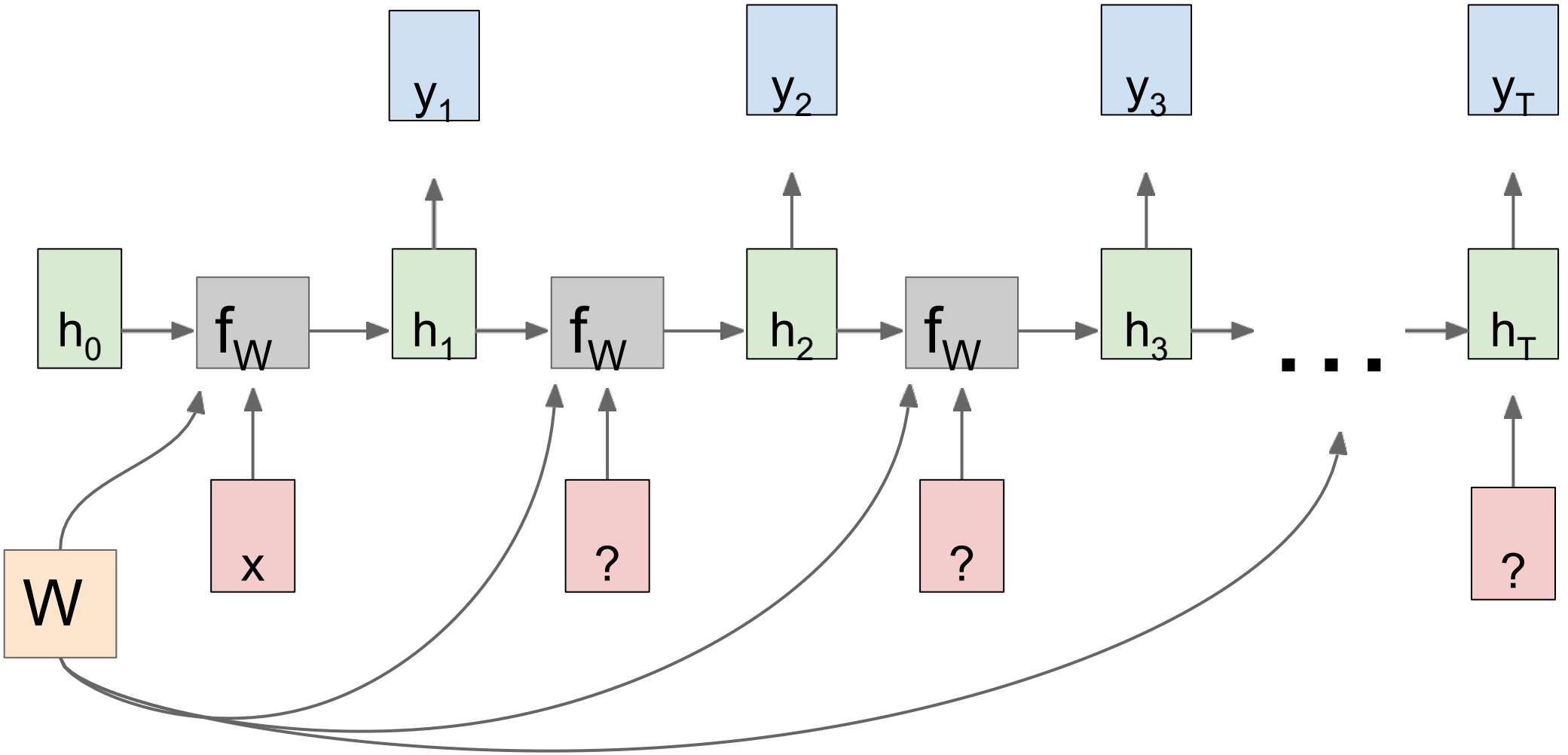
RNN: Computational Graph: Many to One



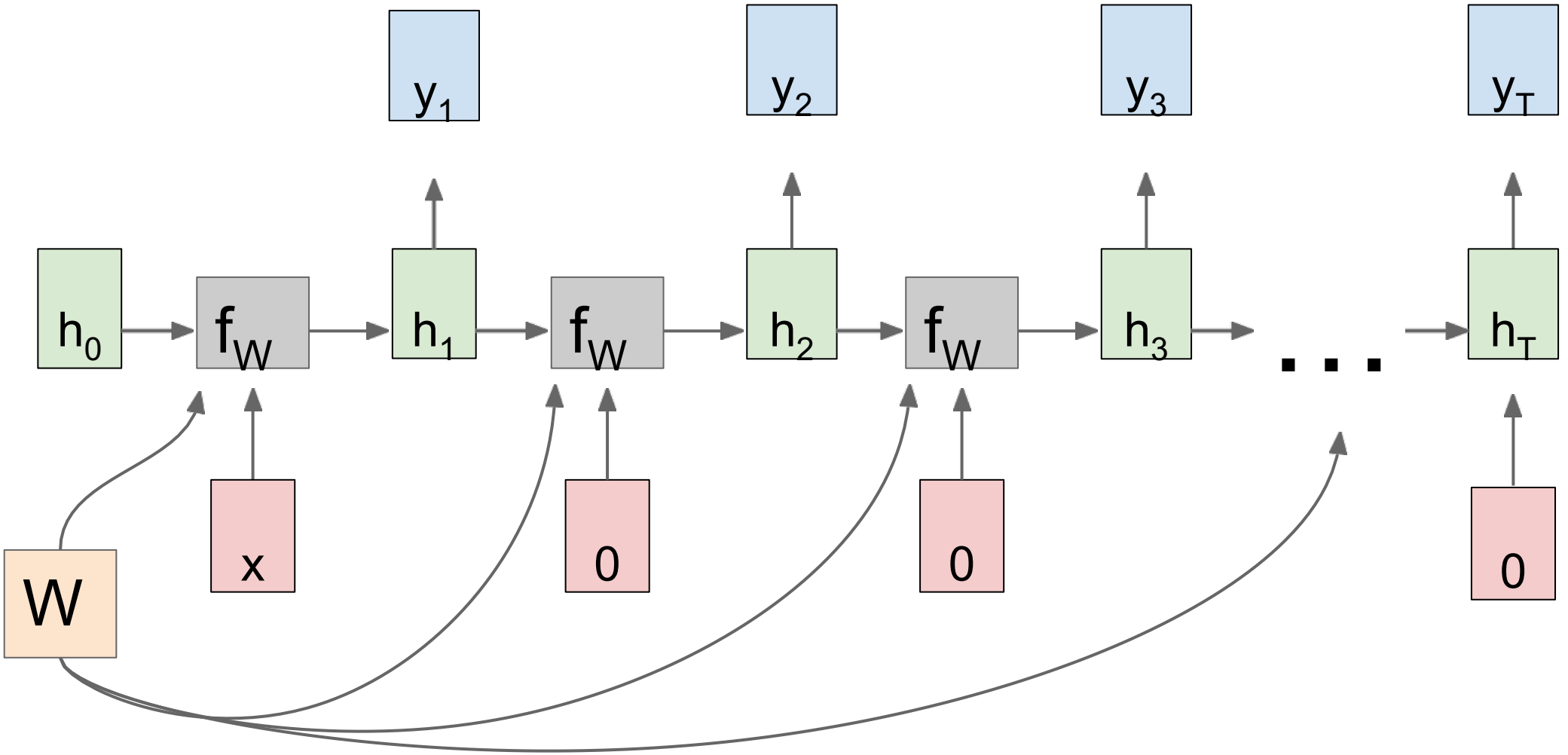
RNN: Computational Graph: One to Many



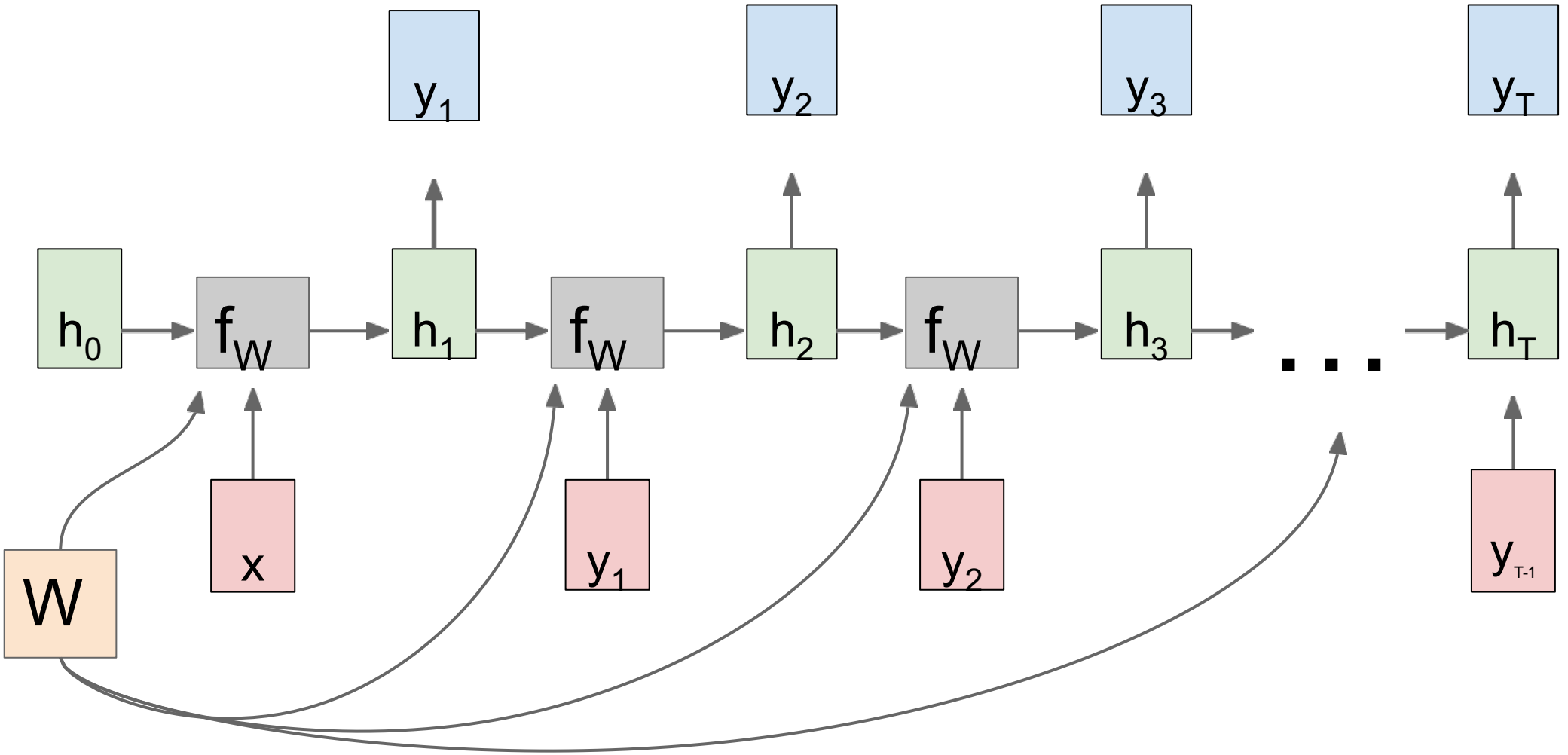
RNN: Computational Graph: One to Many



RNN: Computational Graph: One to Many

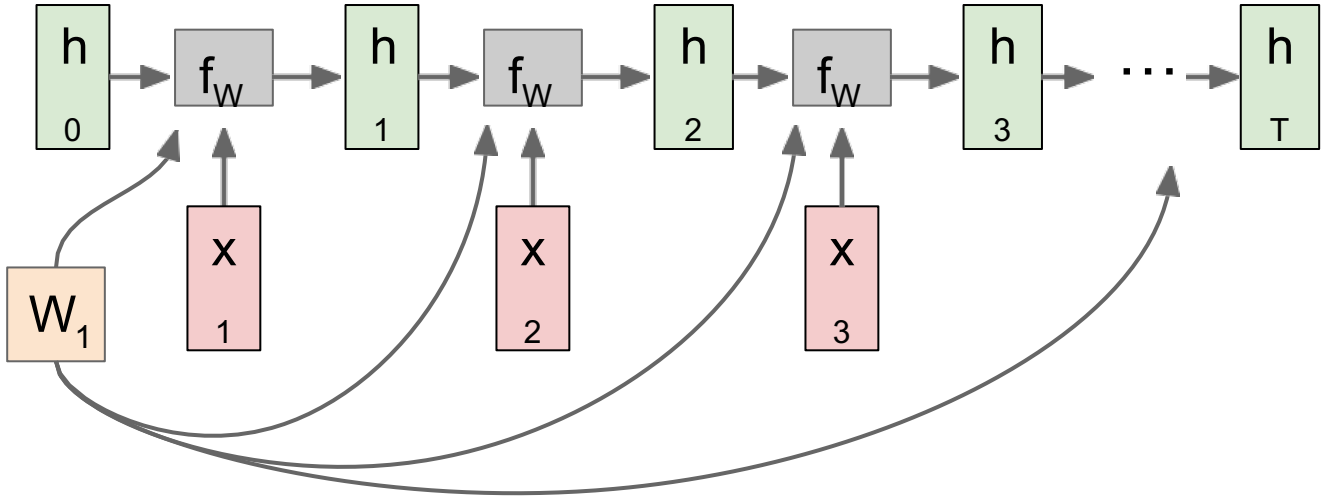


RNN: Computational Graph: One to Many



Sequence to Sequence: Many-to-one + one-to-many

Many to one: Encode input sequence in a single vector

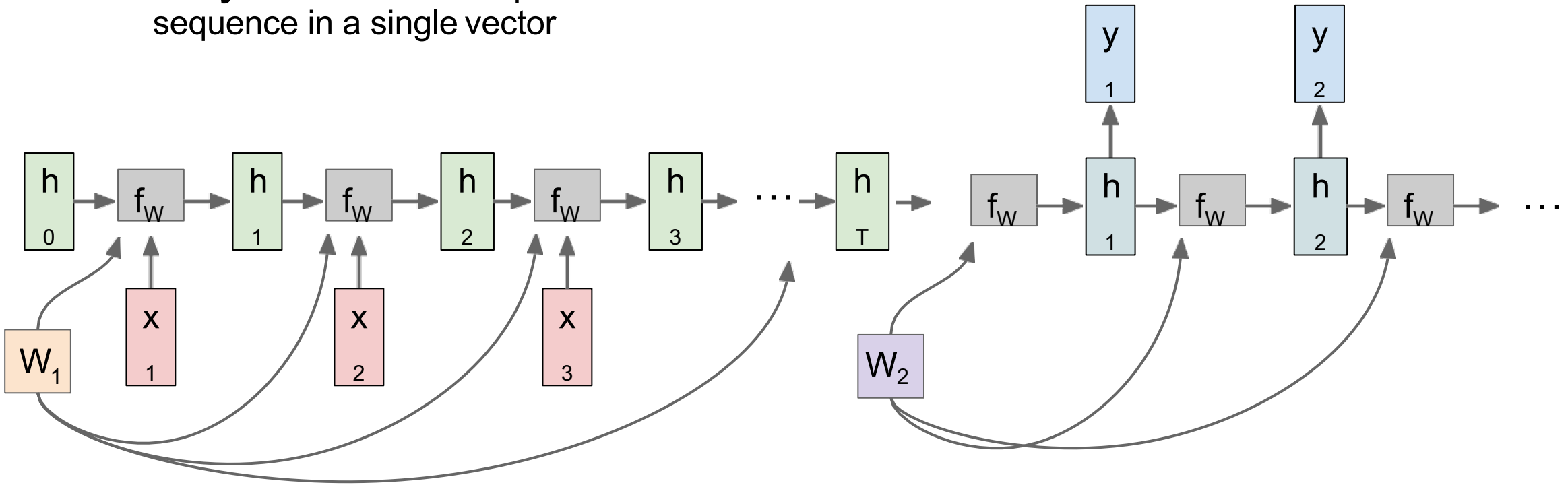


Sutskever et al, “Sequence to Sequence Learning with Neural Networks”, NIPS 2014

Sequence to Sequence: Many-to-one + one-to-many

One to many: Produce output sequence from single input vector

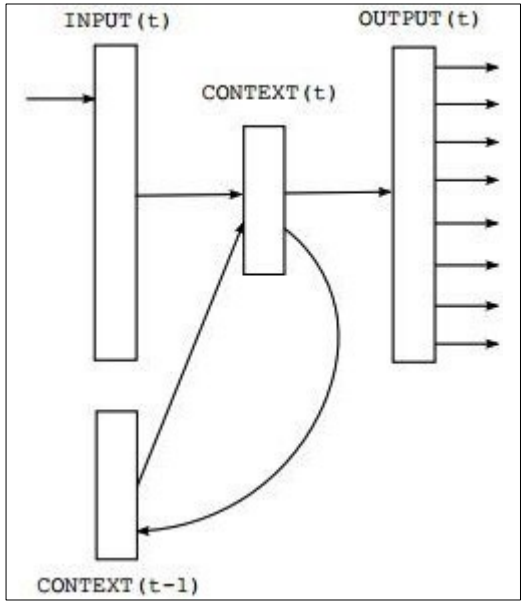
Many to one: Encode input sequence in a single vector



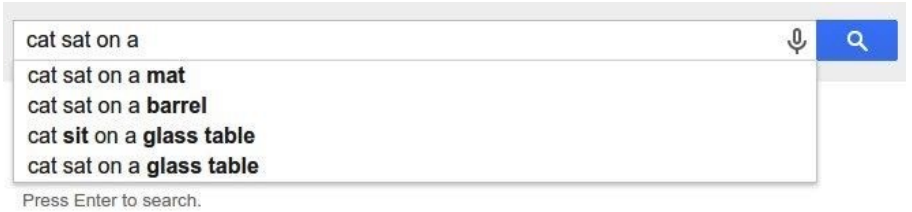
Sutskever et al, “Sequence to Sequence Learning with Neural Networks”, NIPS 2014

slide credit: Andrej Karpathy

Language Models



Word-level language model. Similar to:



Recurrent Neural Network Based Language Model
[Tomas Mikolov, 2010]

slide credit: Andrej Karpathy

Suppose we had the training sentence “cat sat on mat”

We want to train a **language model**:
P(next word | previous words)

i.e. want these to be high:
P(cat | [<S>])
P(sat | [<S>, cat])
P(on | [<S>, cat, sat])
P(mat | [<S>, cat, sat, on])
P(<E>| [<S>, cat, sat, on, mat])

slide credit: Andrej Karpathy

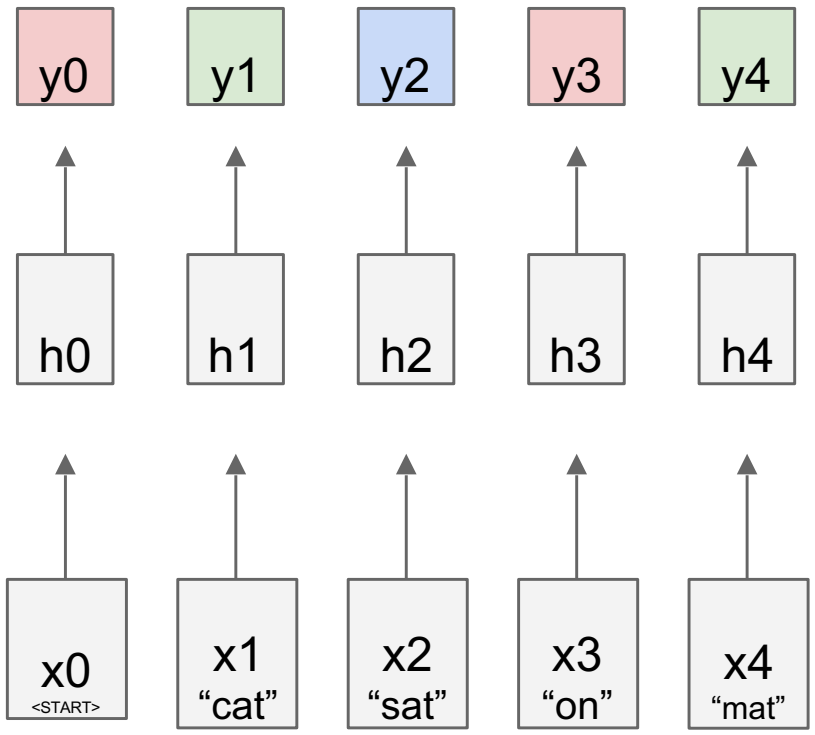
Suppose we had the training sentence “cat sat on mat”

We want to train a **language model**:
P(next word | previous words)

First, suppose we had only a finite, 1-word history:
i.e. want these to be high:
P(cat | <S>)
P(sat | cat)
P(on | sat)
P(mat | on)
P(<E>| mat)

slide credit: Andrej Karpathy

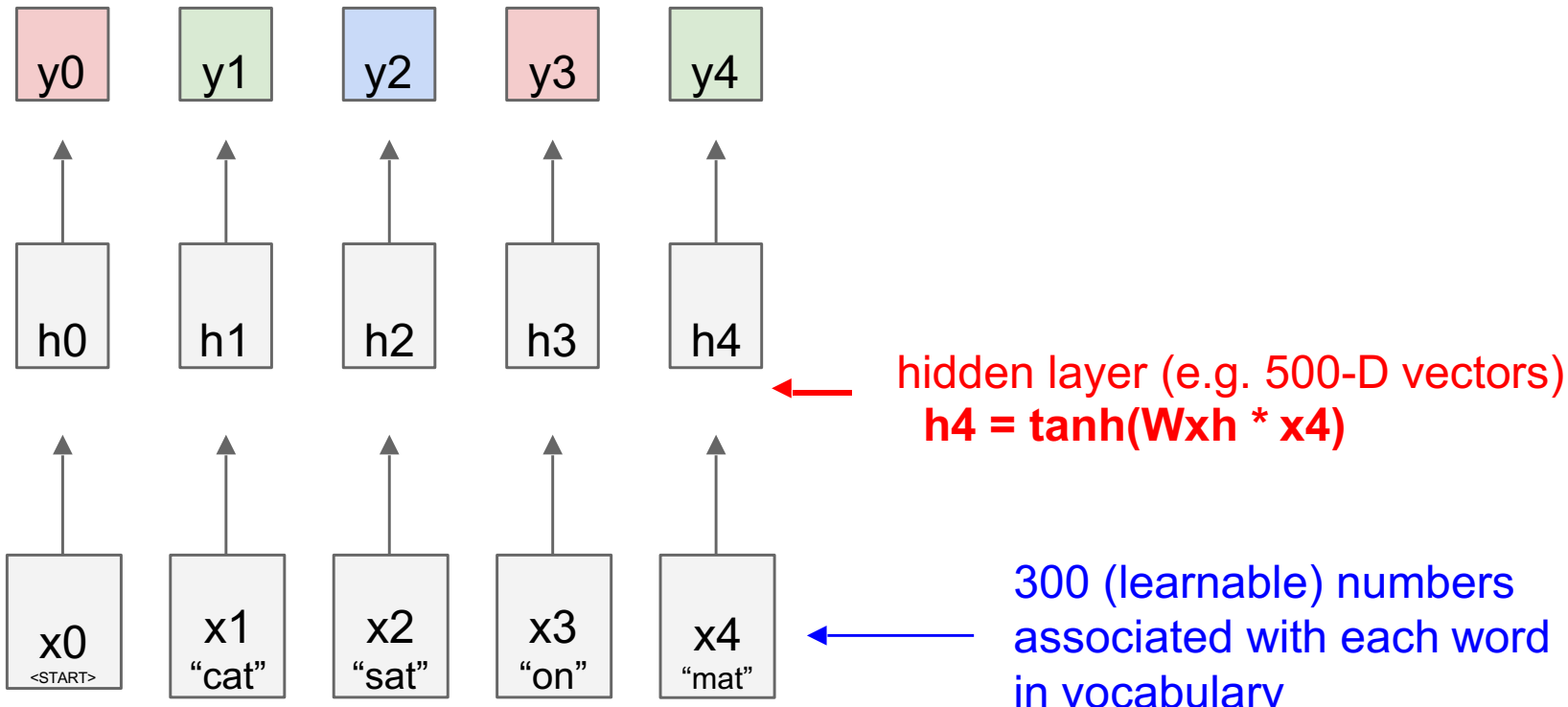
“cat sat on mat”



300 (learnable) numbers associated with each word in vocabulary

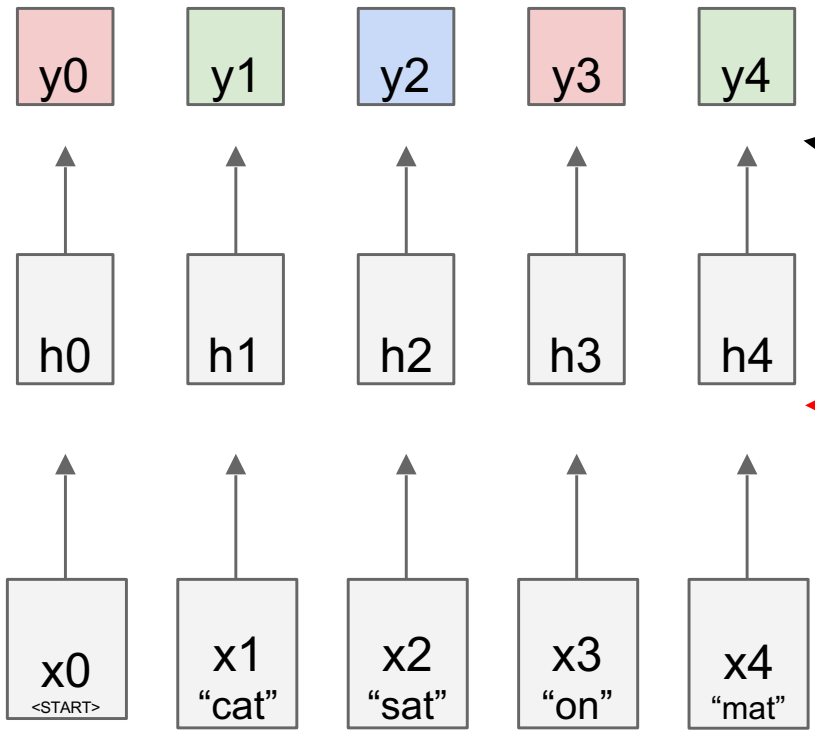
slide credit: Andrej Karpathy

“cat sat on mat”



slide credit: Andrej Karpathy

“cat sat on mat”



10,001-D class scores:
Softmax over 10,000 words and a special <END> token.
y4 = Why * h4

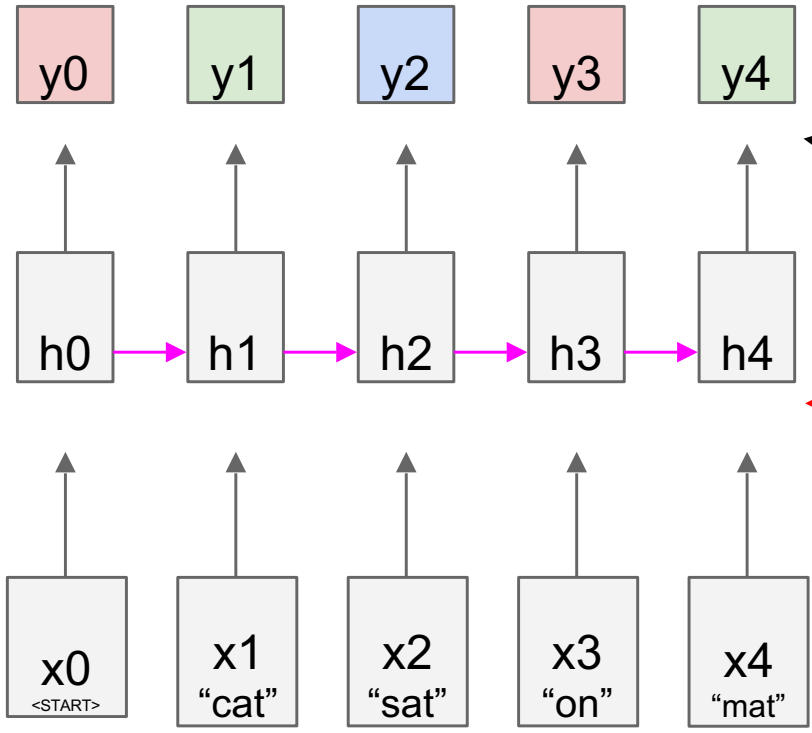
hidden layer (e.g. 500-D vectors)
h4 = tanh(Wxh * x4)

300 (learnable) numbers associated with each word in vocabulary

slide credit: Andrej Karpathy

Recurrent Neural Network:

“cat sat on mat”



10,001-D class scores:
Softmax over 10,000 words and a special <END> token.
y4 = Why * h4

hidden layer (e.g. 500-D vectors)
h4 = tanh(Wxh * x4 + Whh * h3)

300 (learnable) numbers associated with each word in vocabulary

slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

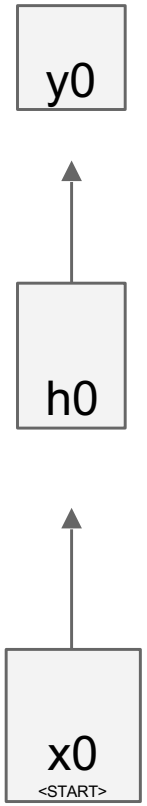
P(next word | previous words)

slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)

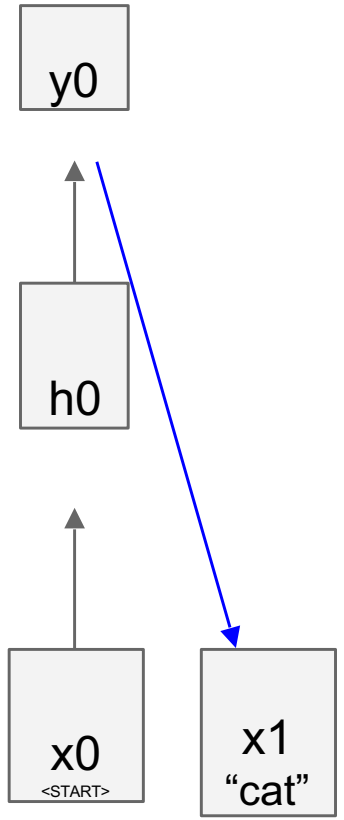


slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)



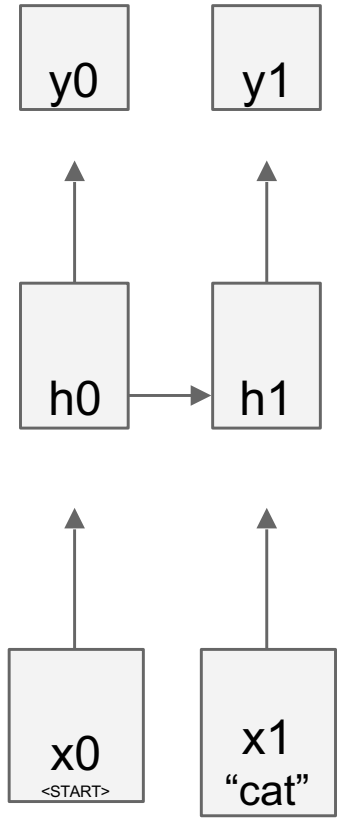
sample!

slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)

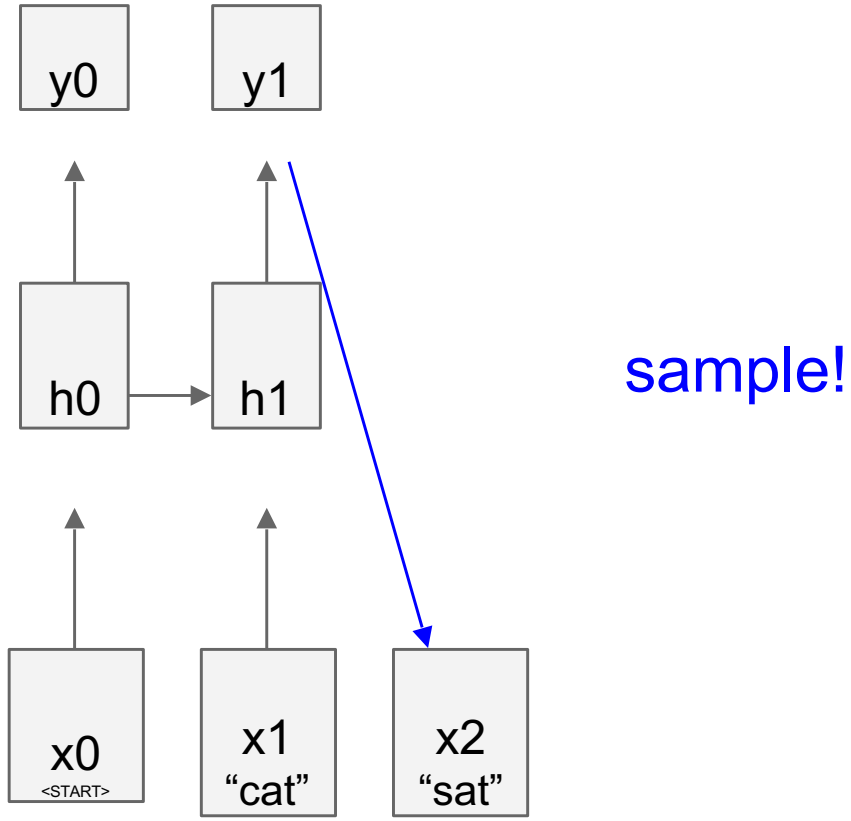


slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)

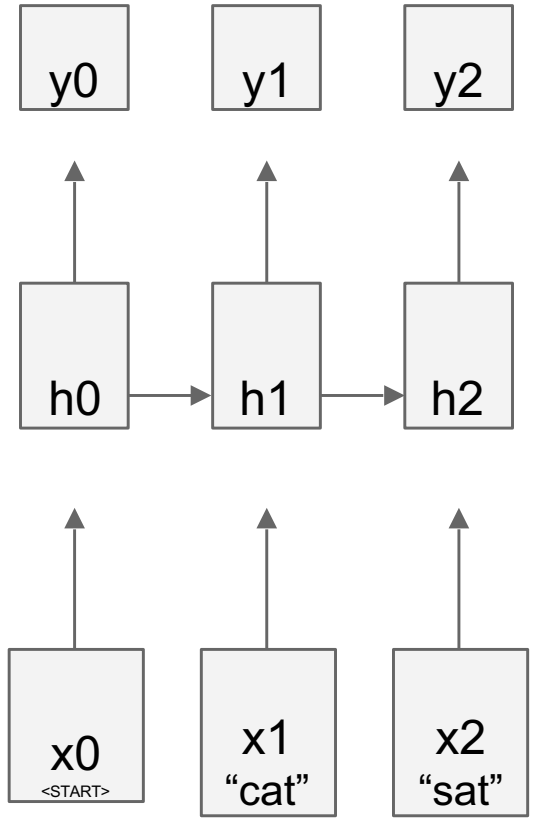


slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)

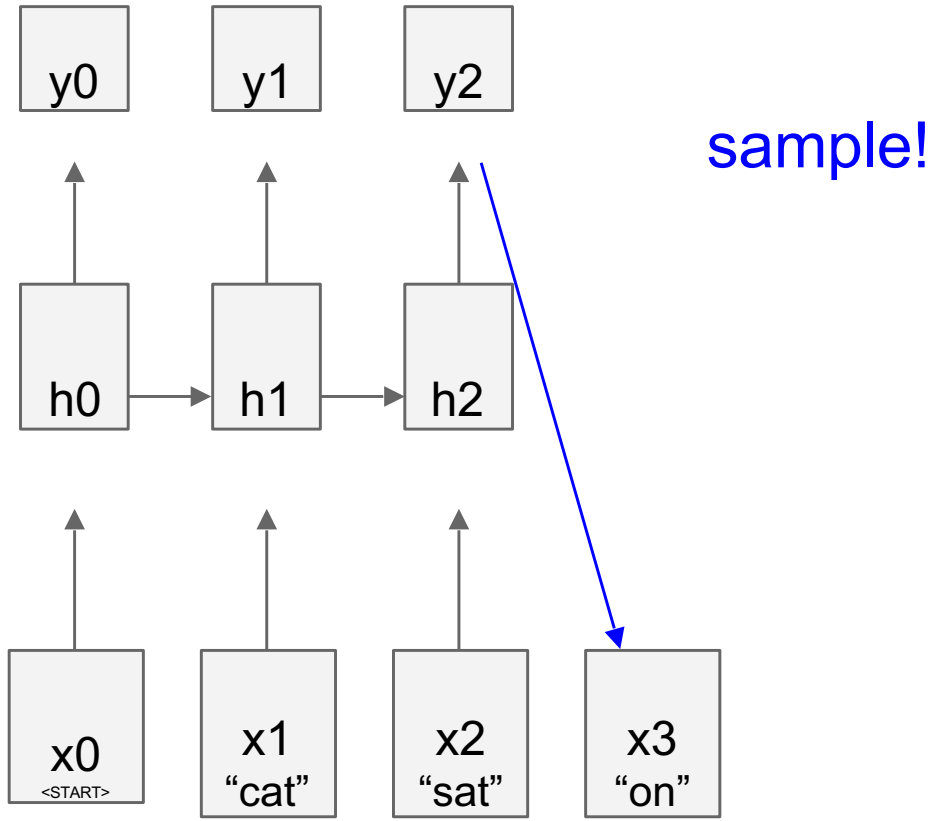


slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)

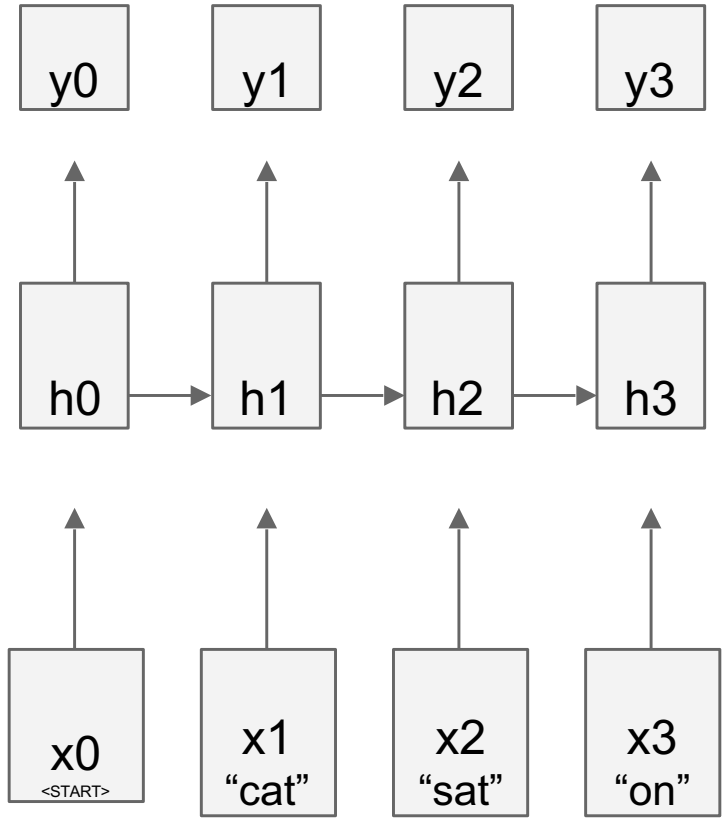


slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)

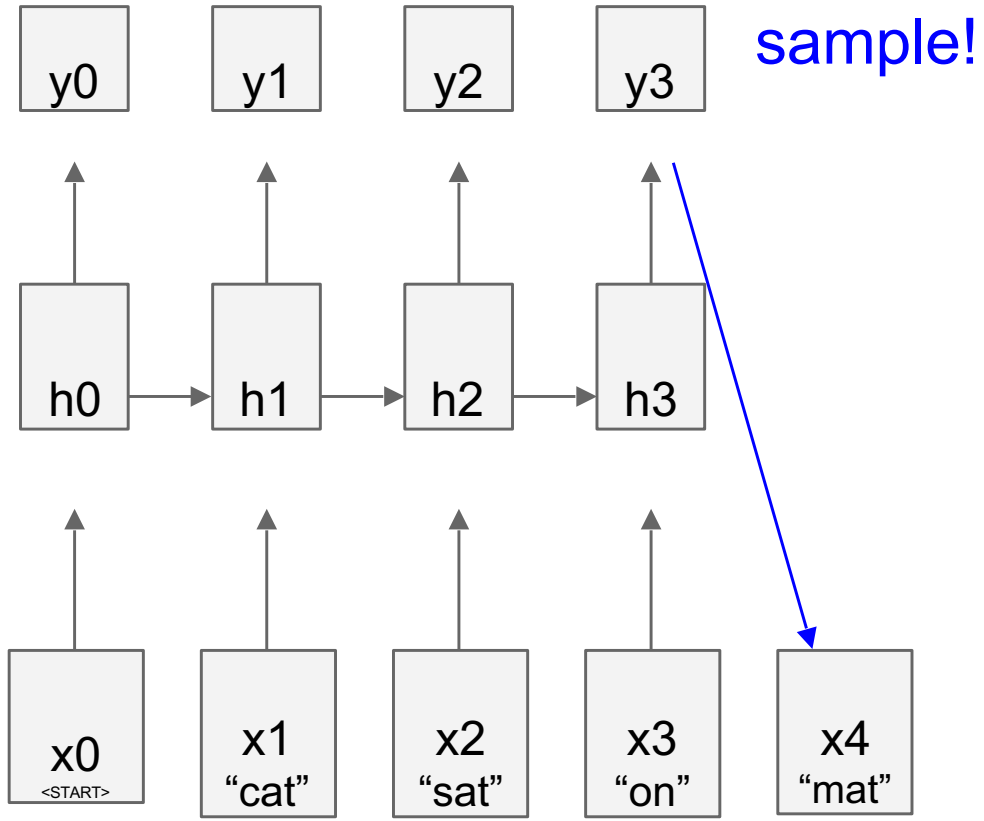


slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)

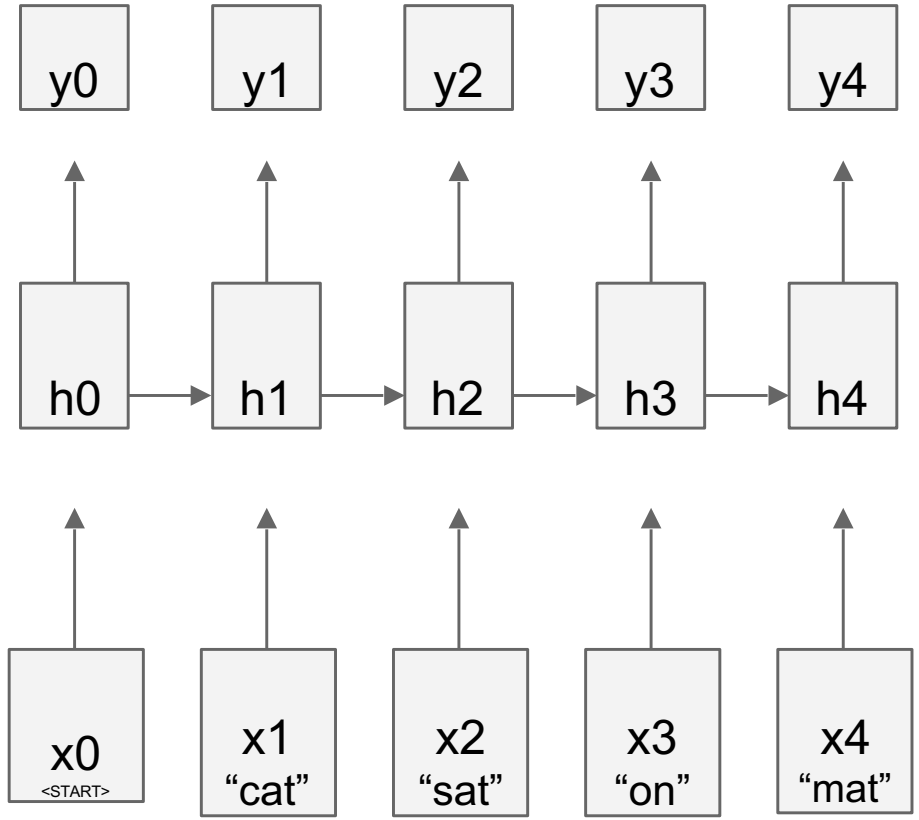


slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

P(next word | previous words)

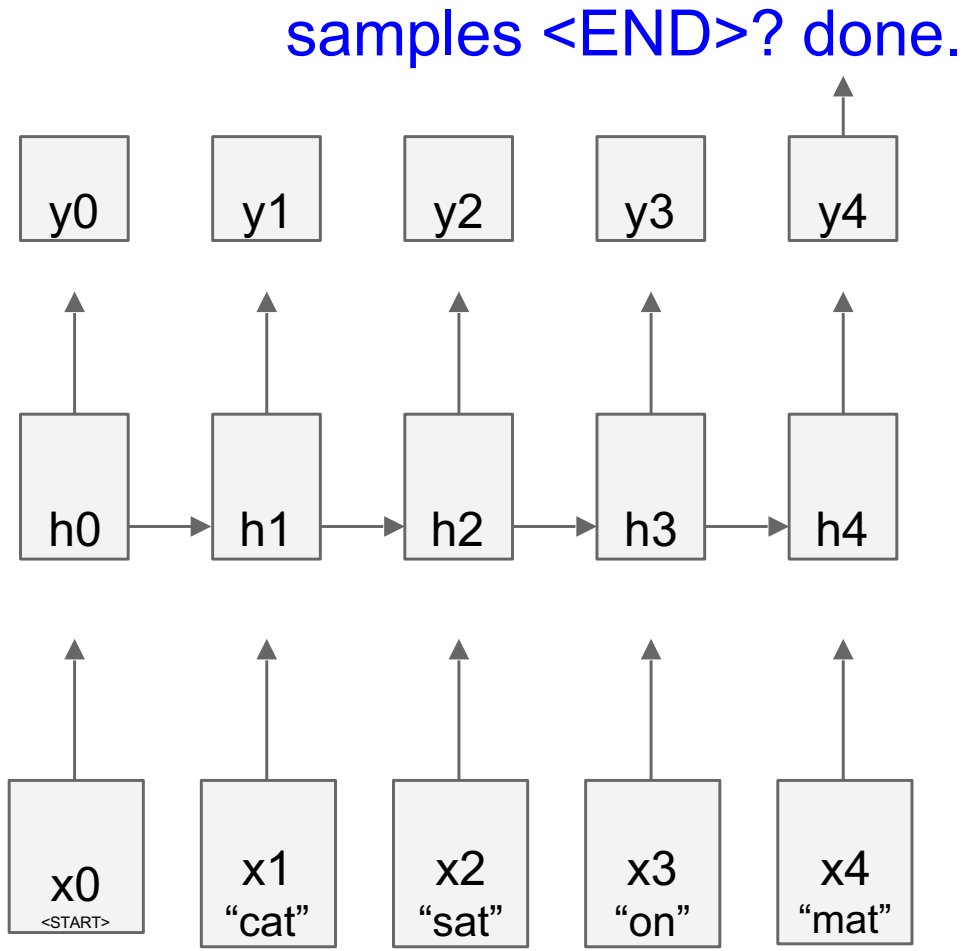


slide credit: Andrej Karpathy

Generating Sentences...

Training this on a lot of sentences would give us a language model. A way to predict

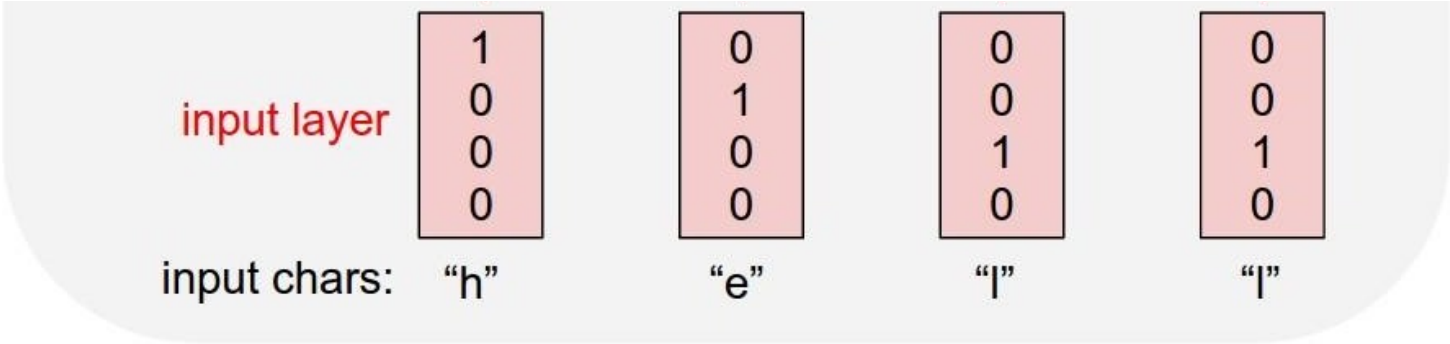
P(next word | previous words)



Example: Character-level Language Model

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

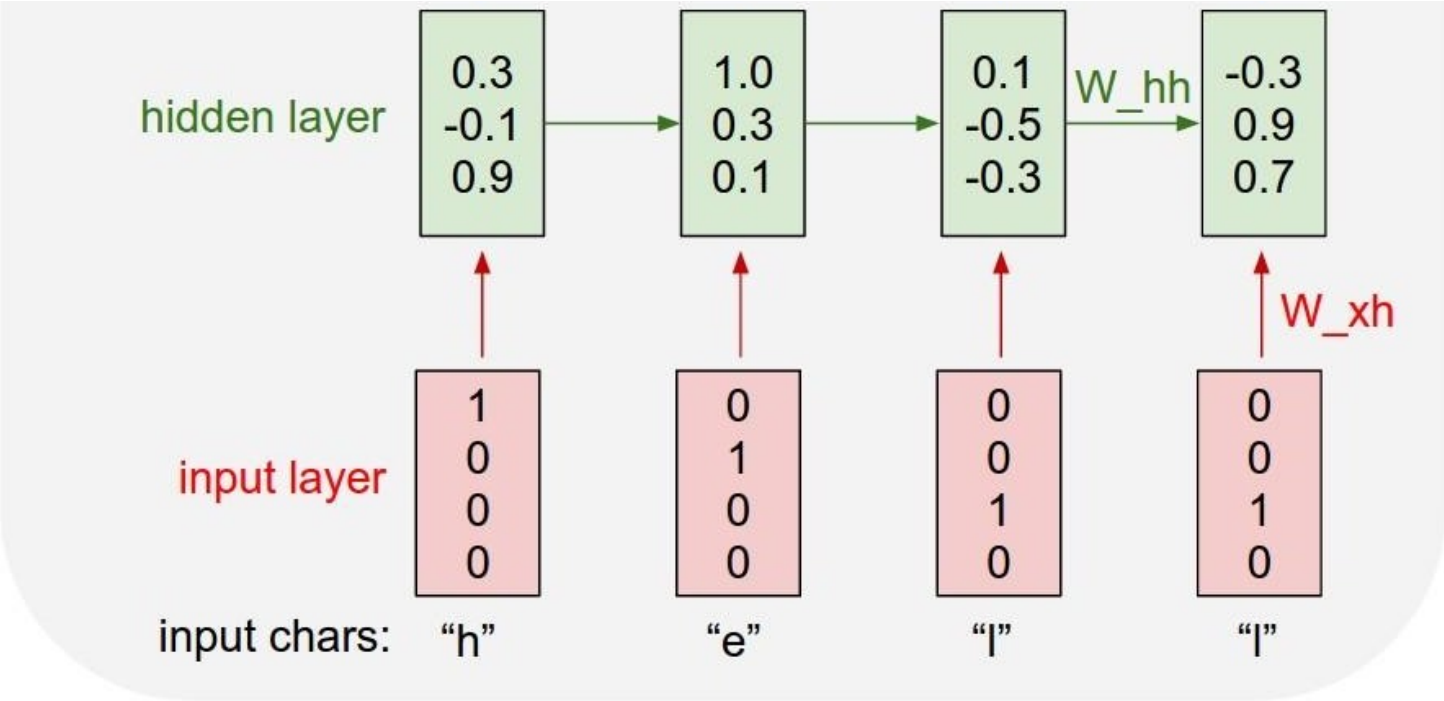


Example: Character-level Language Model

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t)$$

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

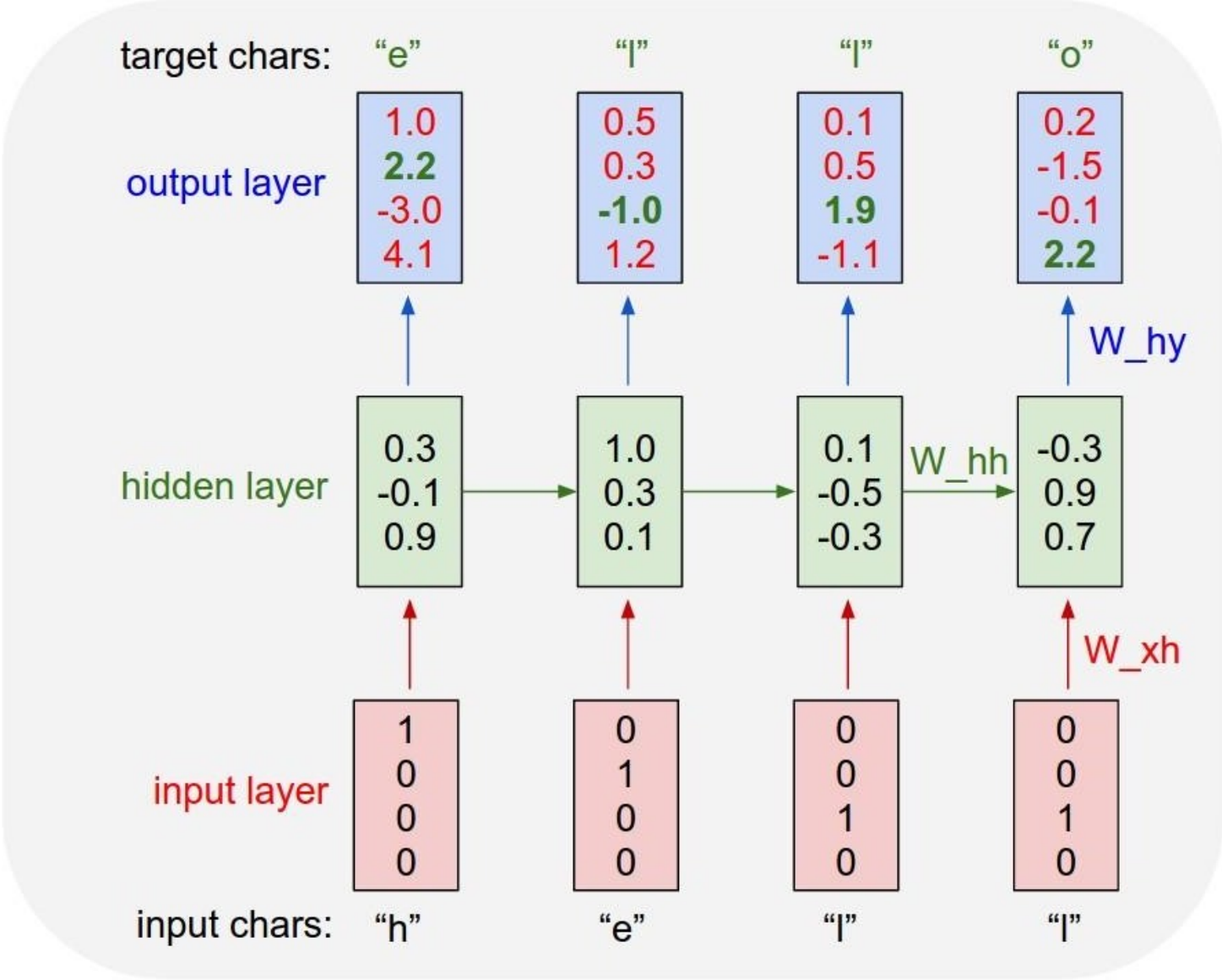


Example: Character-level Language Model

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

aka GT forcing

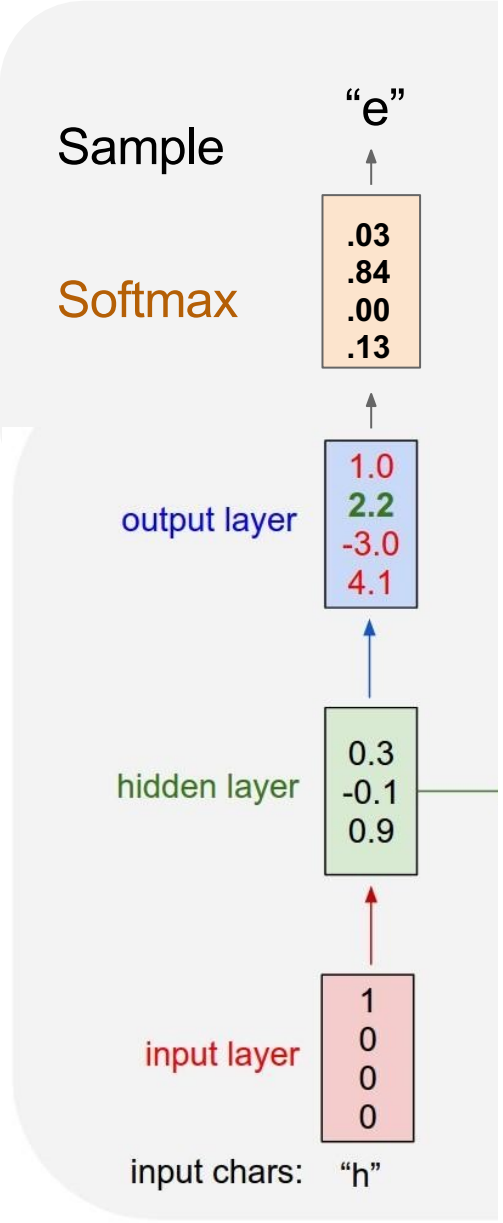


Exposure bias: He Zhang Zhou Glass ICLR 2020 Quantifying Exposure Bias for Neural Language Generation

Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

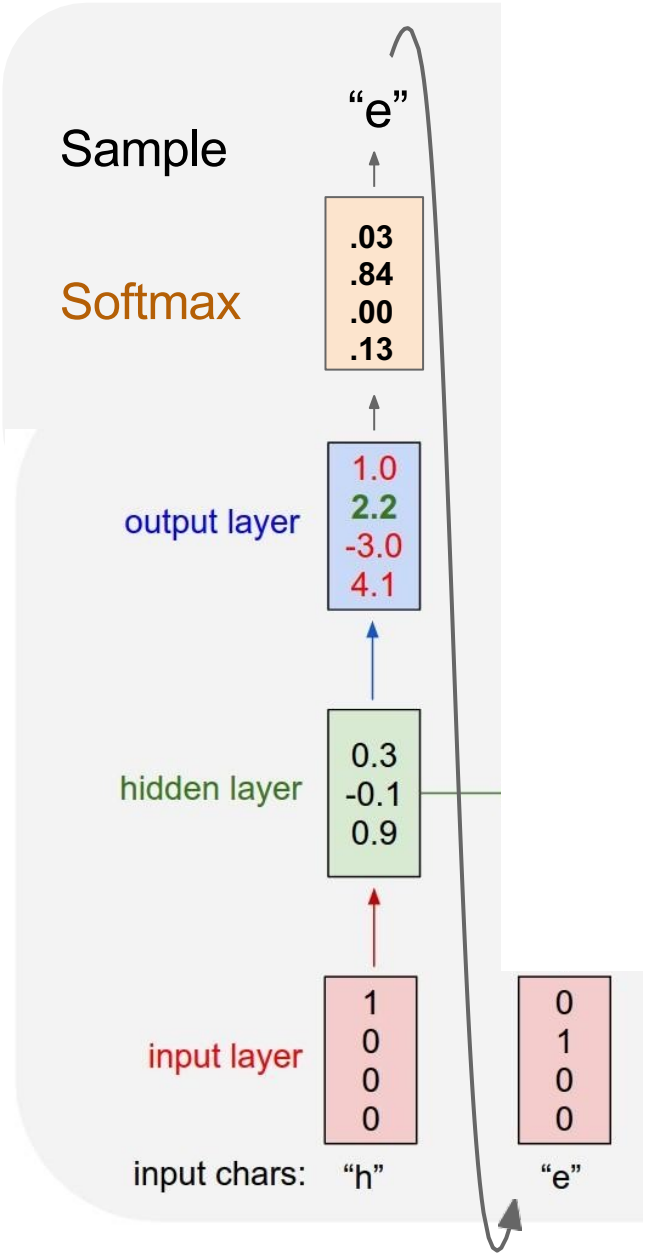
At test-time sample characters one at a time, feed back to model



Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

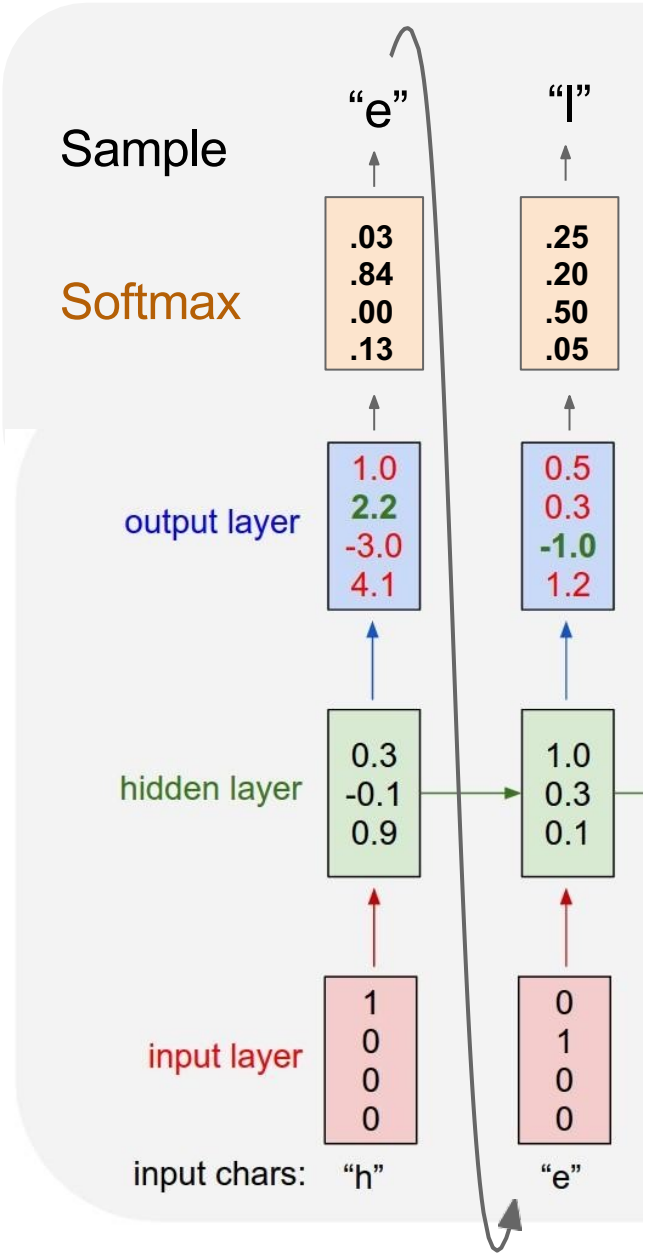
At test-time sample characters one at a time, feed back to model



Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

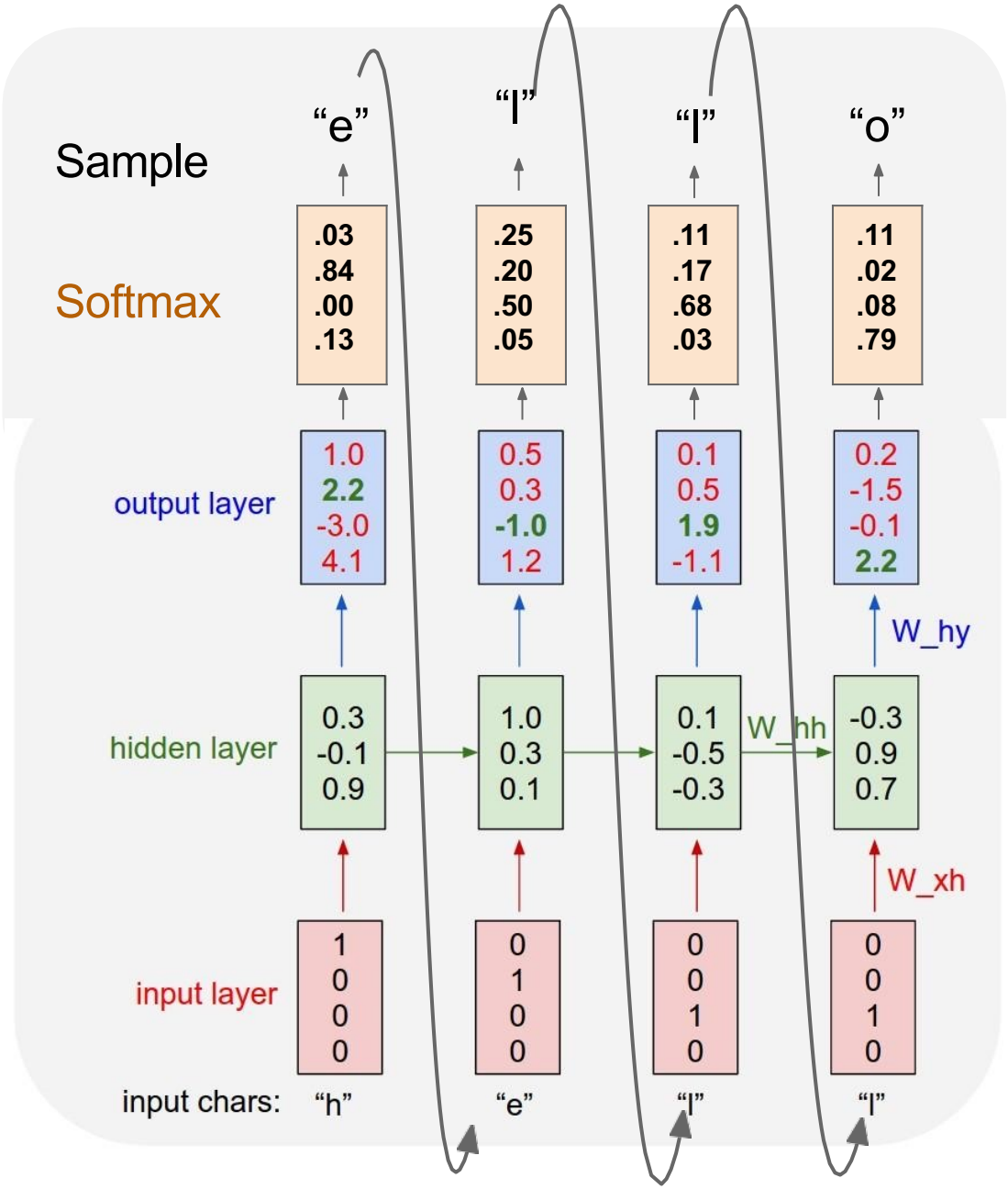
At test-time sample characters one at a time, feed back to model



Example: Character-level Language Model Sampling

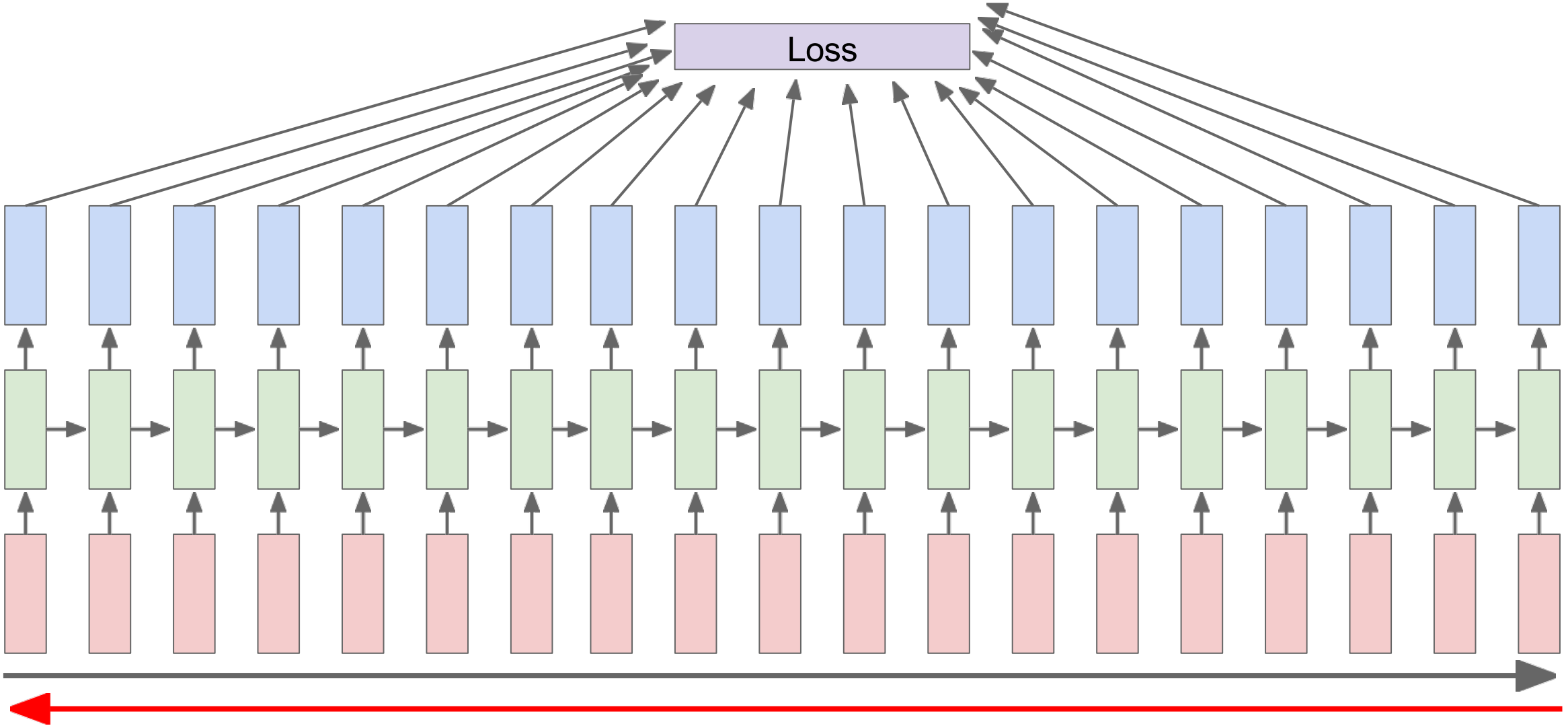
Vocabulary:
[h,e,l,o]

At test-time sample characters one at a time, feed back to model

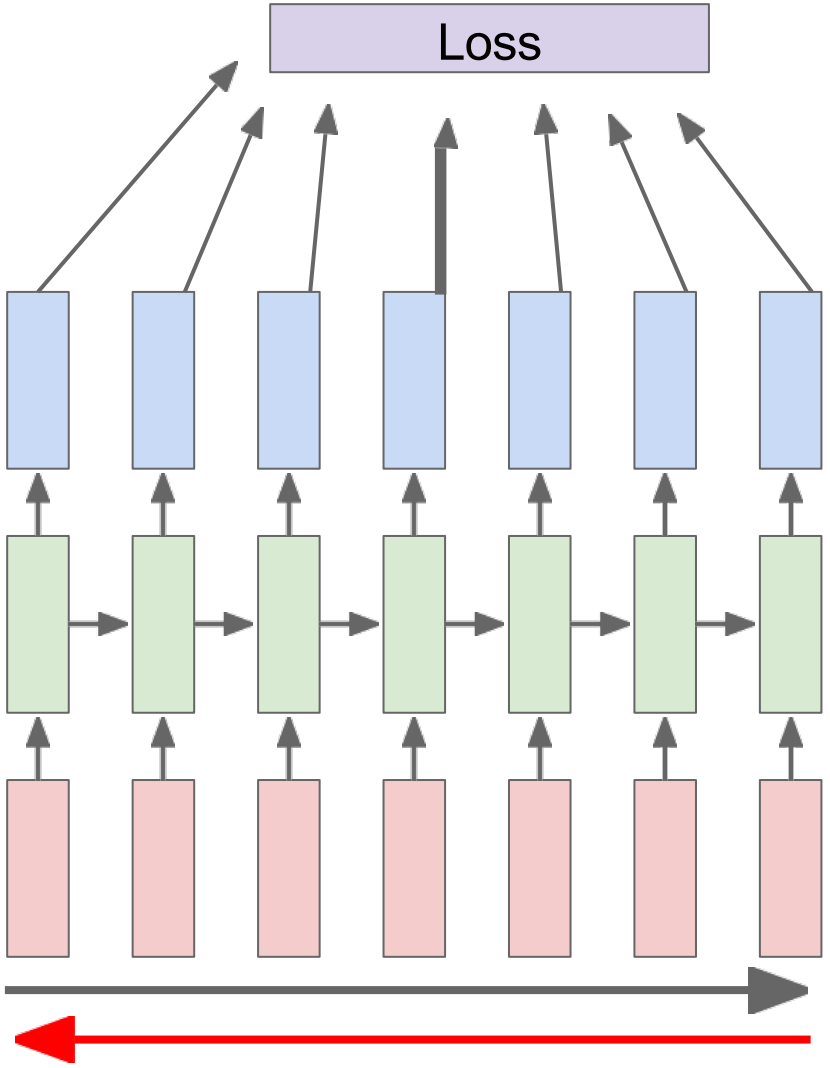


Backpropagation through time

Forward through entire sequence to compute loss, then backward through entire sequence to compute gradient

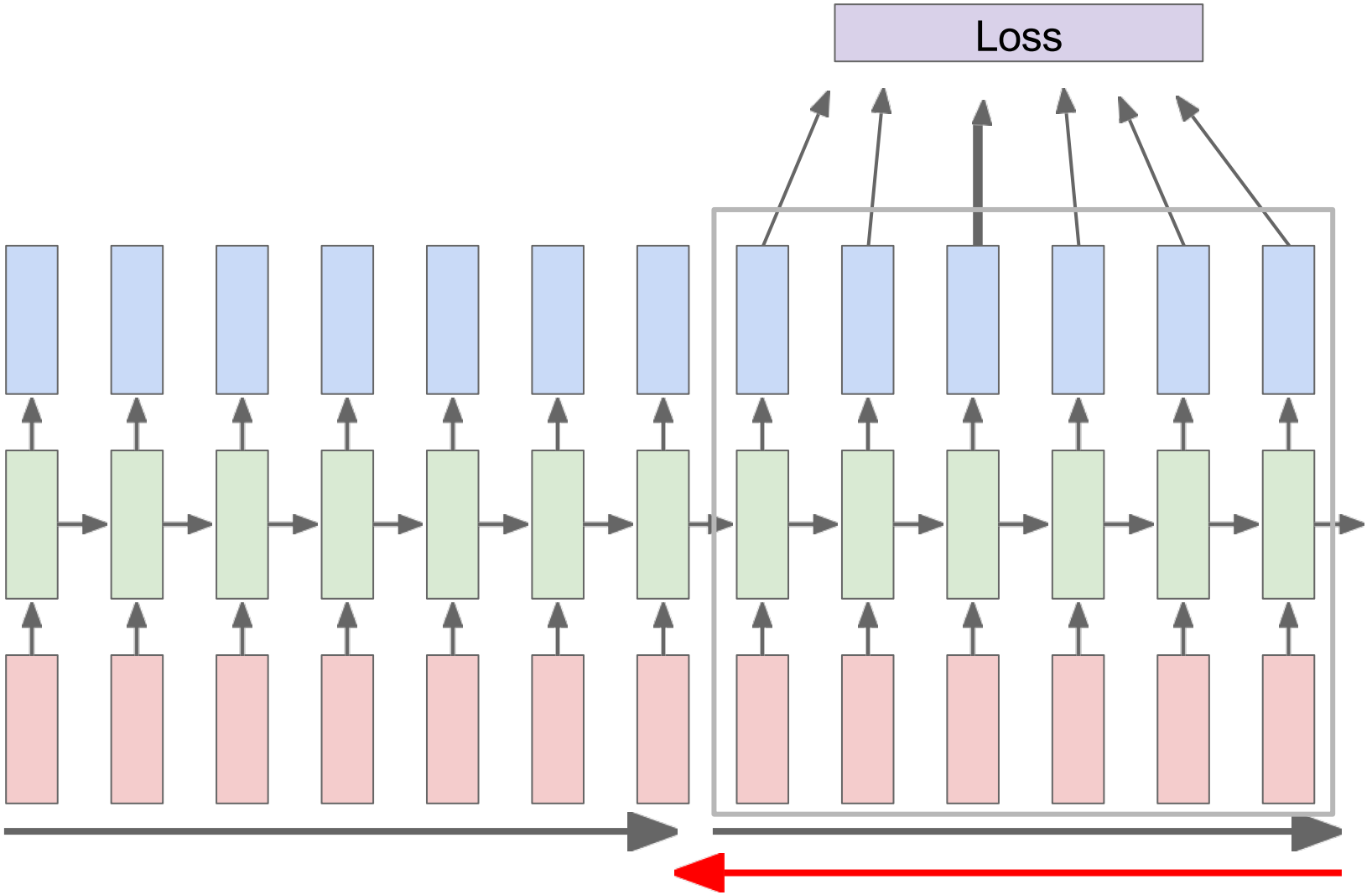


Truncated Backpropagation through time



Run forward and backward through chunks of the sequence instead of whole sequence

Truncated Backpropagation through time



Carry hidden states forward in time forever, but only backpropagate for some smaller number of steps

Truncated Backpropagation through time

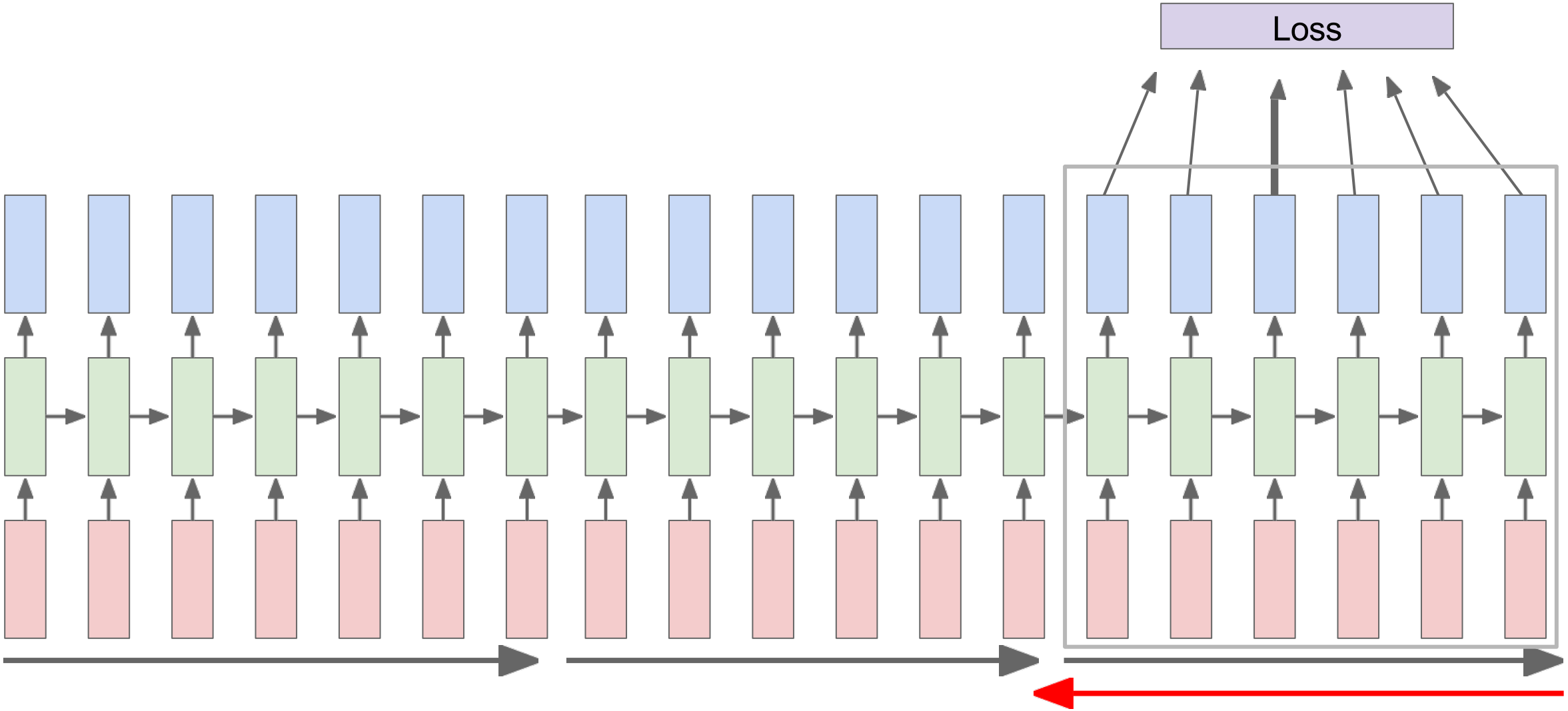


Image Captioning

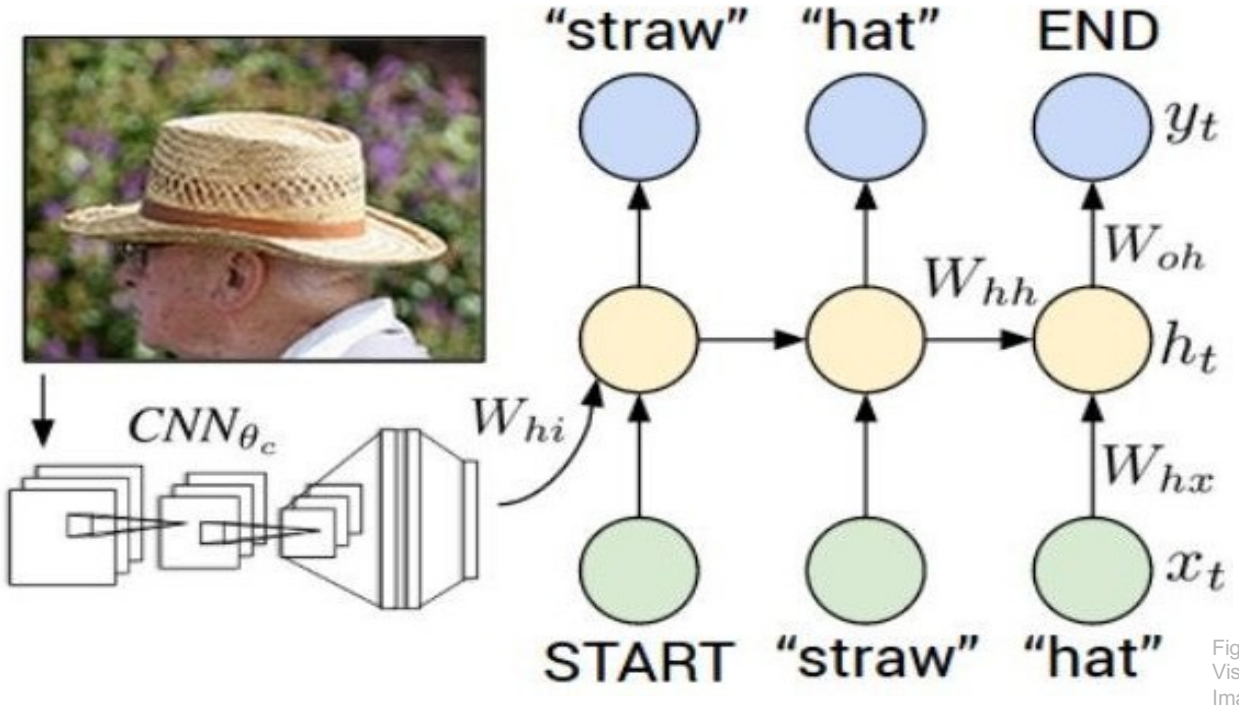
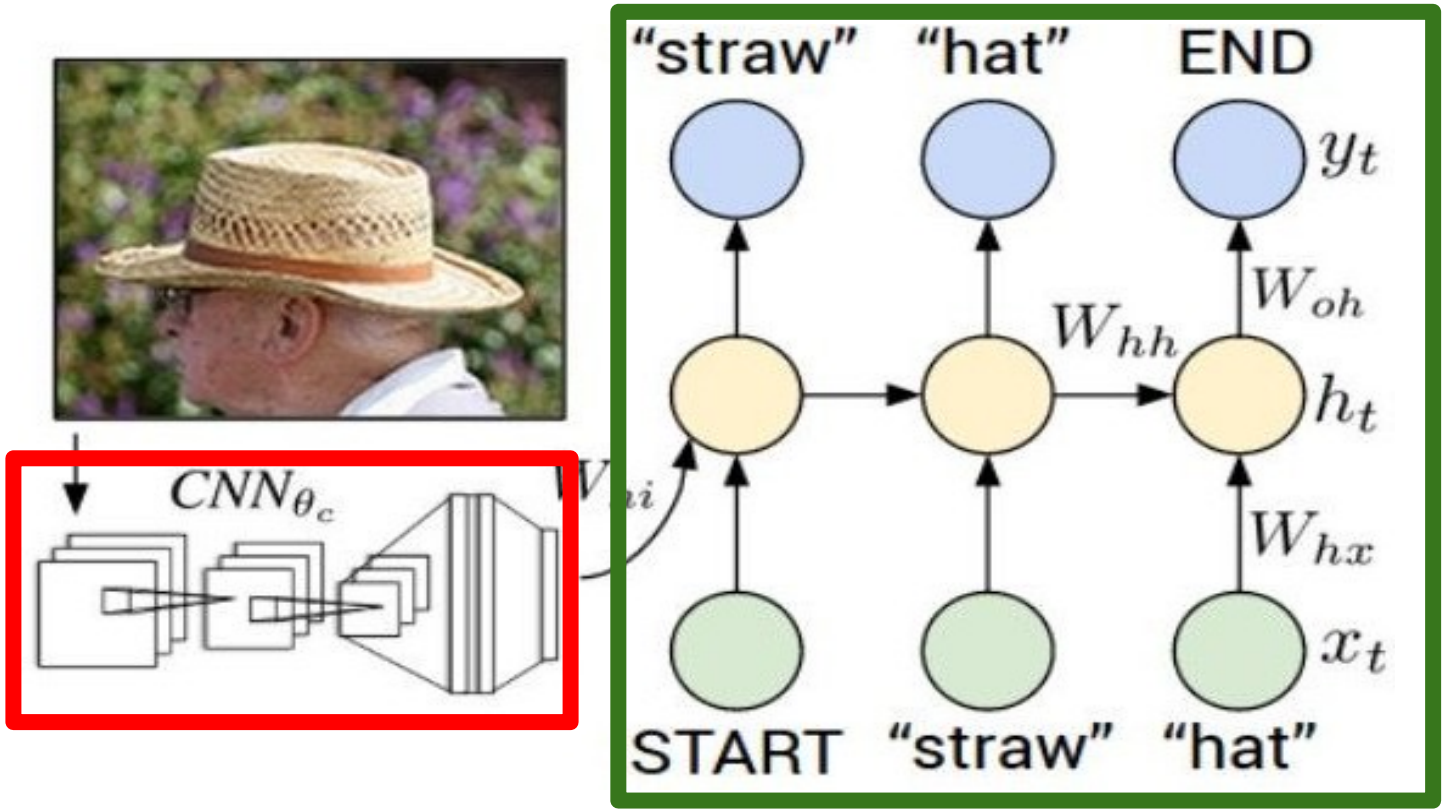


Figure from Karpathy et a, "Deep Visual-Semantic Alignments for Generating Image Descriptions", CVPR 2015; figure copyright IEEE, 2015.
Reproduced for educational purposes.

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

Recurrent Neural Network



Convolutional Neural Network

test image

This image is CC0 public domain

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

FC-4096

FC-4096

FC-1000

softmax

test image

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

FC-4096

FC-4096

FC-1000

softmax

X

test image

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

FC-4096

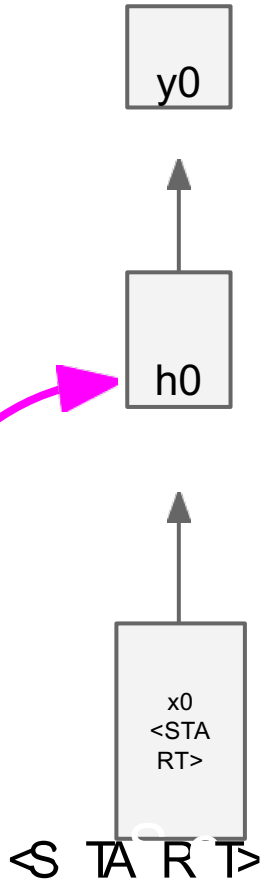
FC-4096

test image

x0
<STA
RT>

<START>

Wih



test image

before:

$h = \tanh(W_{xh} * x + W_{hh} * h)$

now:

$h = \tanh(W_{xh} * x + W_{hh} * h + \mathbf{W_{ih} * v})$

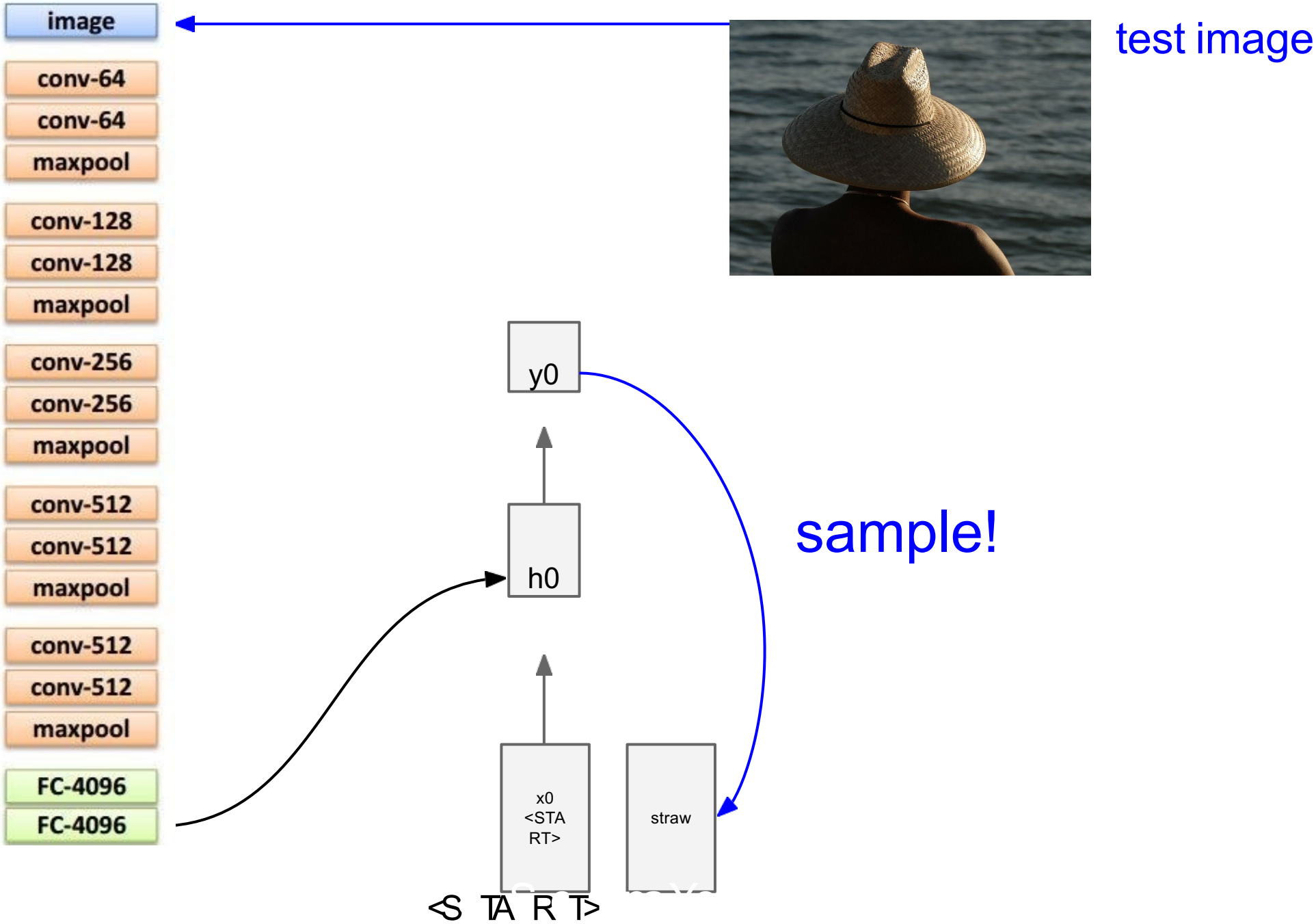


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
test image
y0
h0
x0
<STA
RT>
straw
<START>
sample!

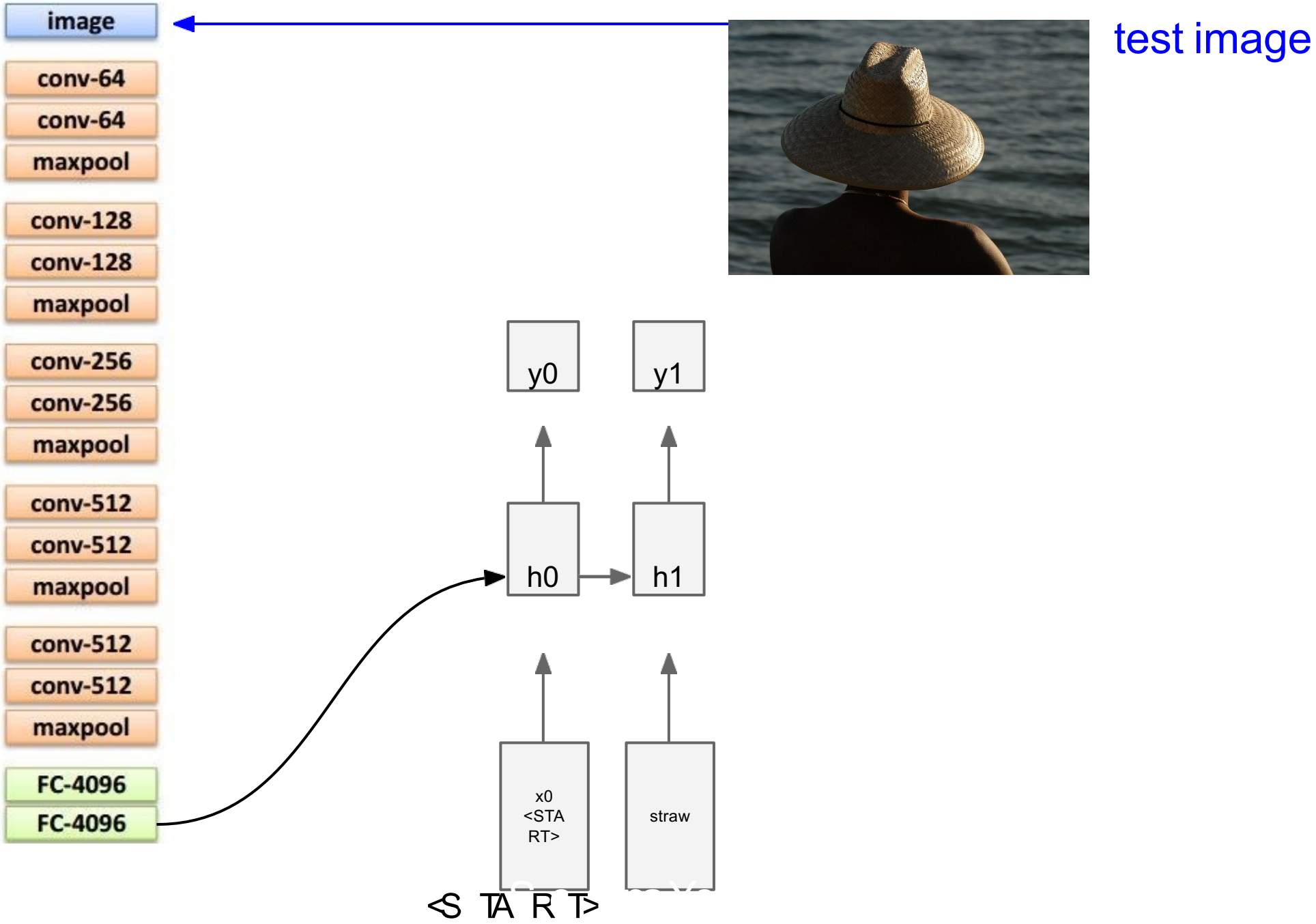


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
test image
y0
y1
h0
h1
x0
<STA
RT>
straw
<START>

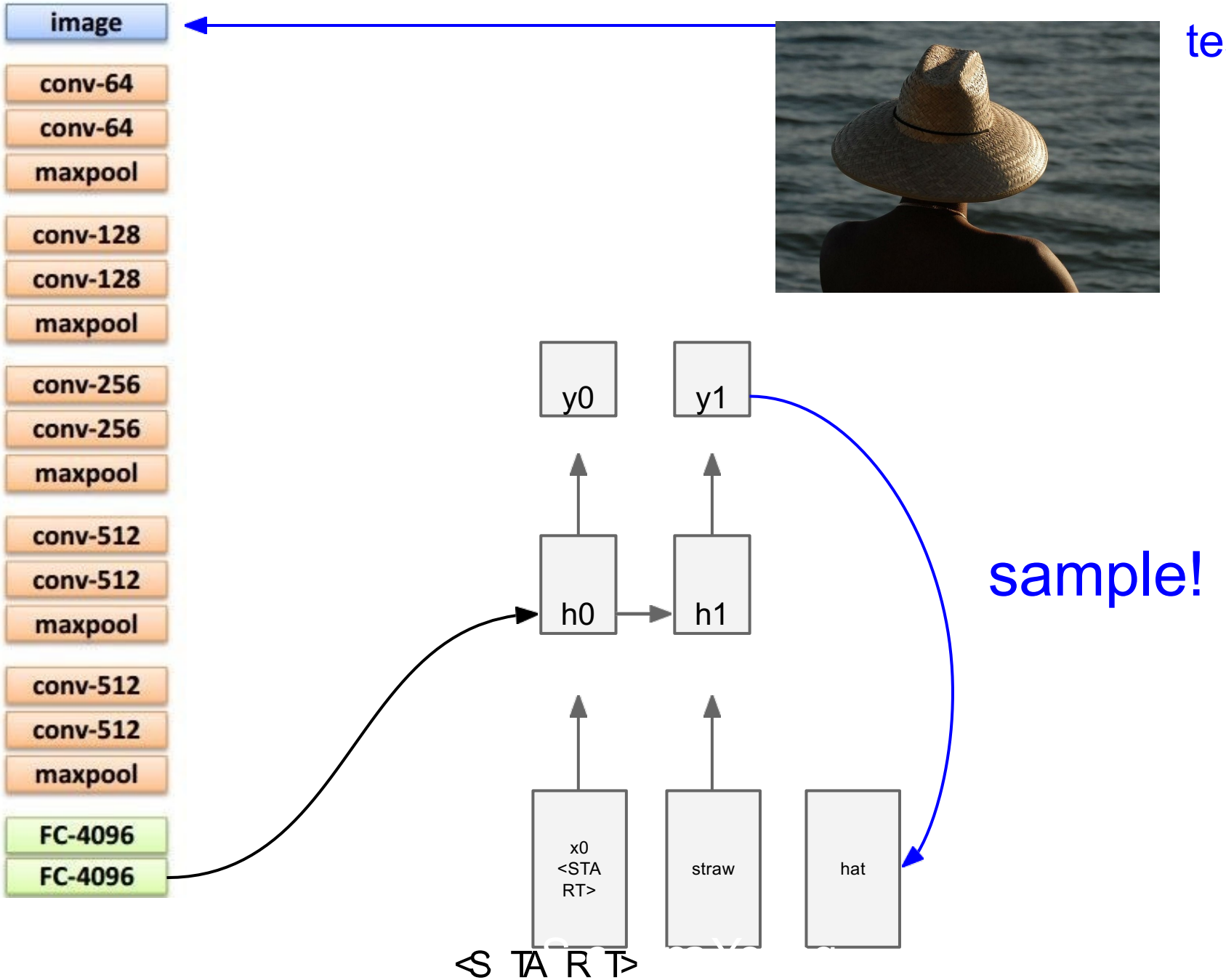


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
test image
y0
y1
h0
h1
sample!
x0
<STA
RT>
straw
hat
<START>

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

FC-4096

FC-4096

test image

y0 y1 y2

h0 h1 h2

x0 <STA RT>

straw

hat

<S TA R T>

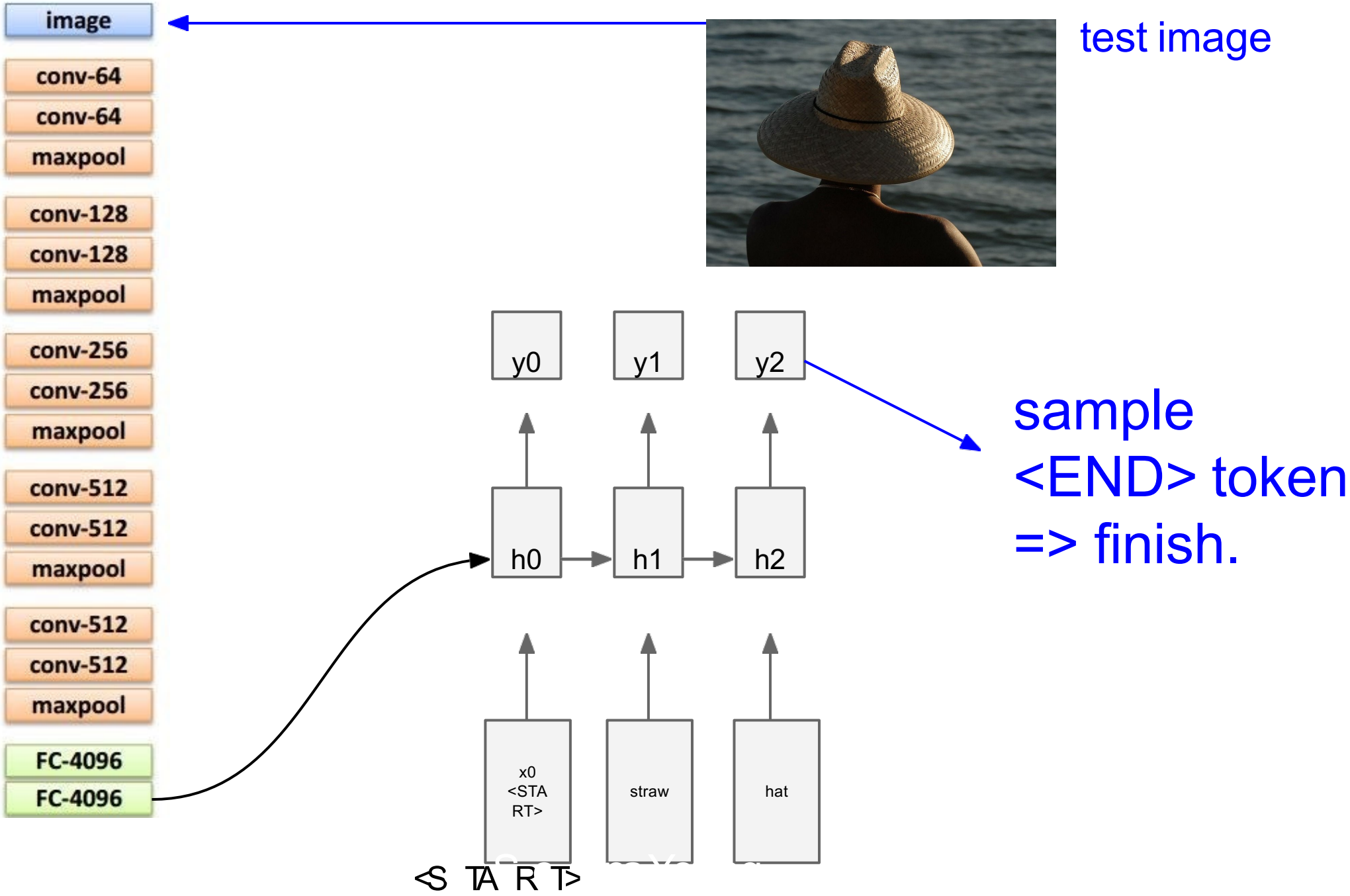


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
test image
y0
y1
y2
h0
h1
h2
x0
<STA
RT>
straw
hat
<START>
sample
<END> token
=> finish.

Image Captioning: Example Results

Captions generated using neuraltalk2
All images are CC0 Public domain: cat suitcase, cat tree, dog, bear, surfers, tennis, giraffe, motorcycle

A cat sitting on a suitcase on the floor

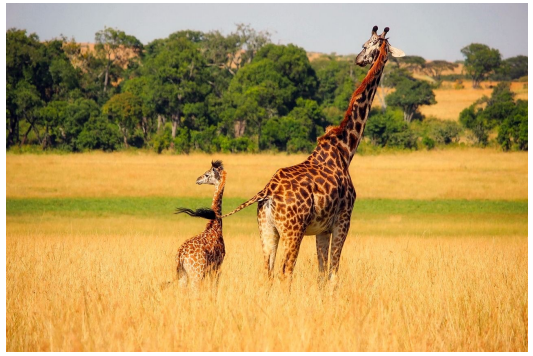
A cat is sitting on a tree branch

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

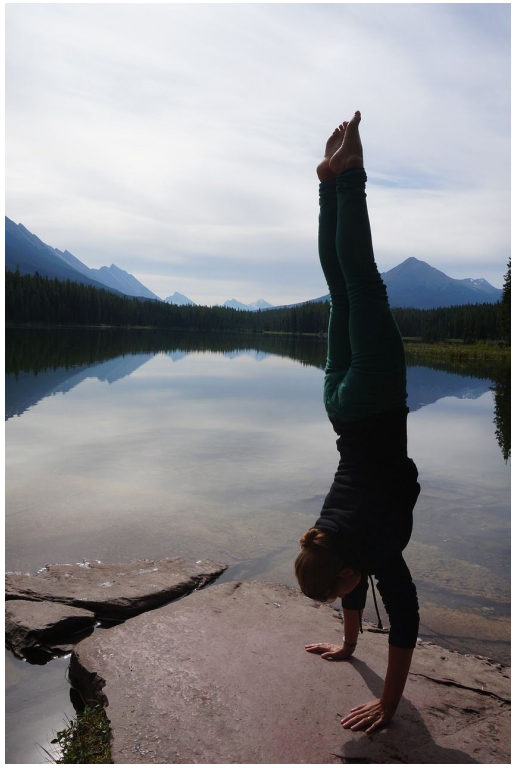
A man riding a dirt bike on a dirt track

Image Captioning: Failure Cases

Captions generated using neuraltalk2
All images are CC0 Public domain: fur coat, handstand, spider web, baseball

A woman is holding a cat in her hand

A person holding a computer mouse on a desk



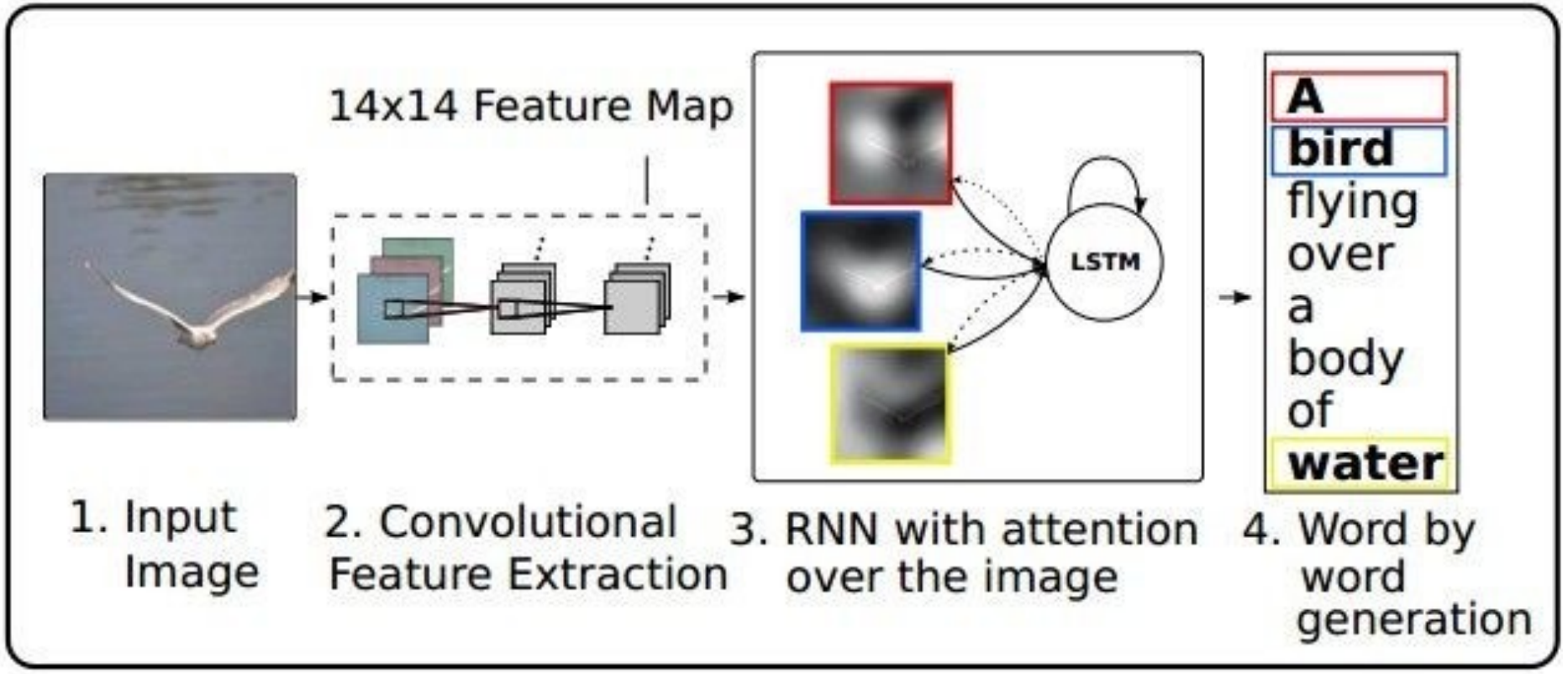
A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

A man in a baseball uniform throwing a ball

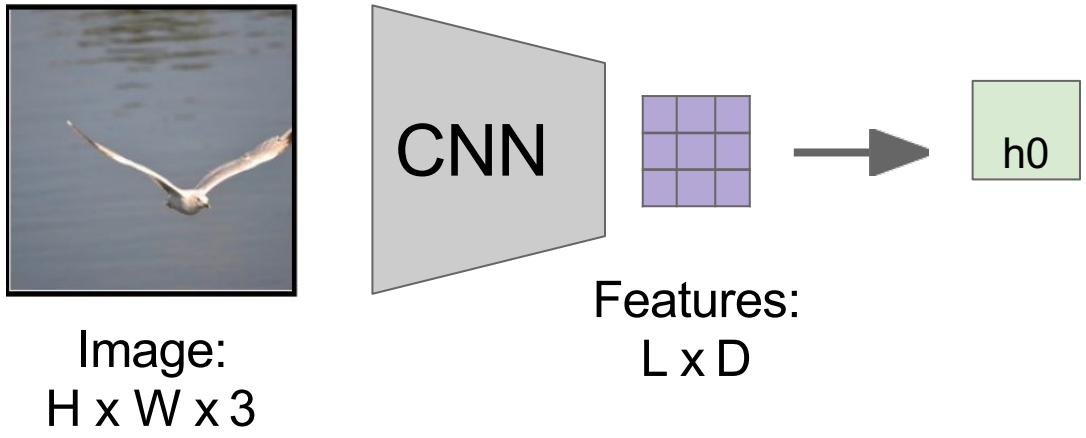
Image Captioning with Attention

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



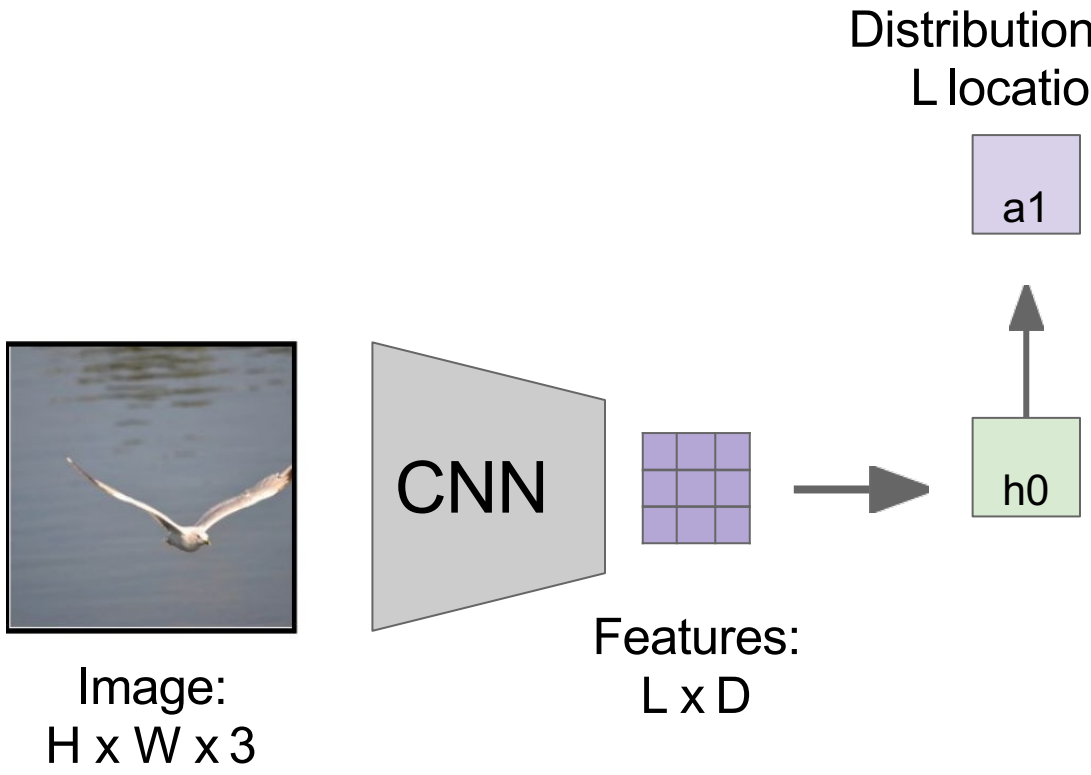
Xu et al, "Show, Attend, and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

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

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

a1

h0

Weighted
features: D

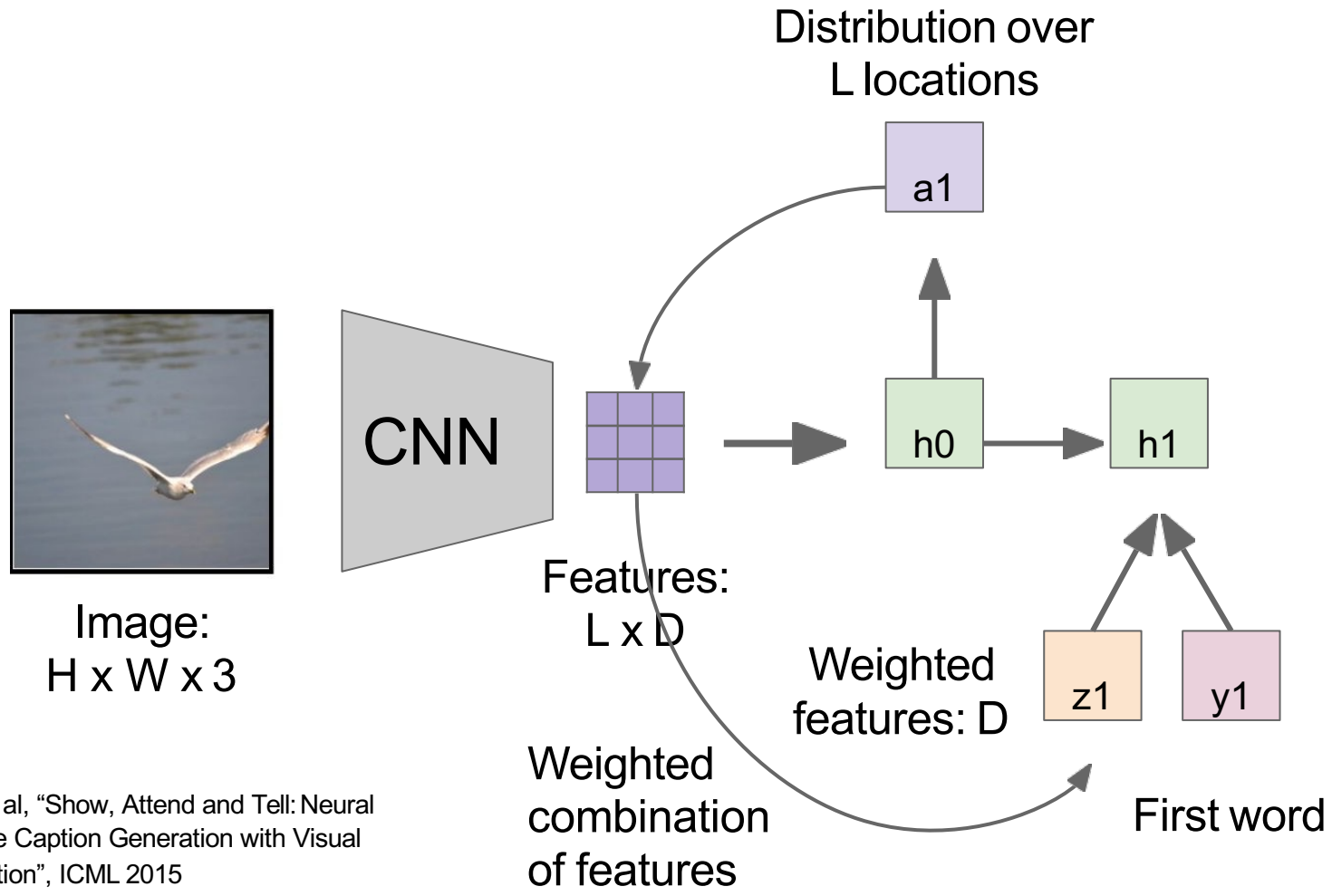
z1

Weighted
combination
of features

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

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

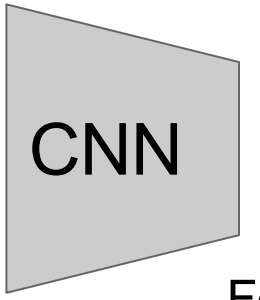
Weighted
combination
of features

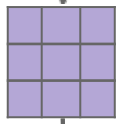
First word

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

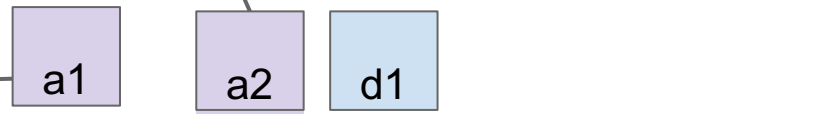


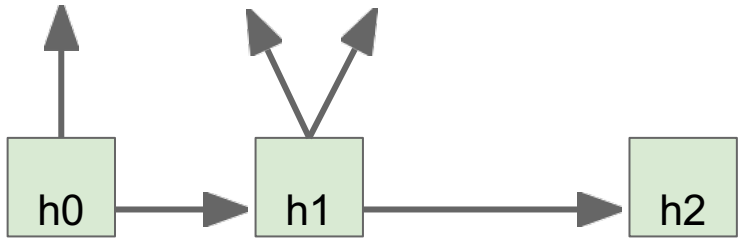


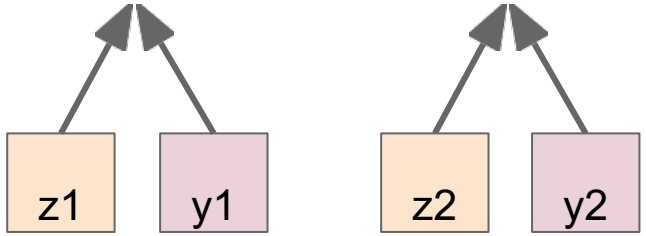
Features:
L x D

Distribution over
L locations

Distribution
over vocab







Weighted
features: D

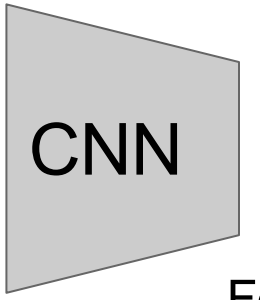
Weighted
combination
of features

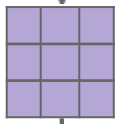
First word

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

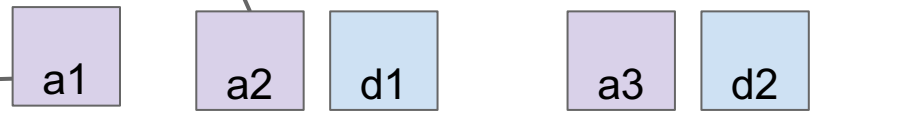


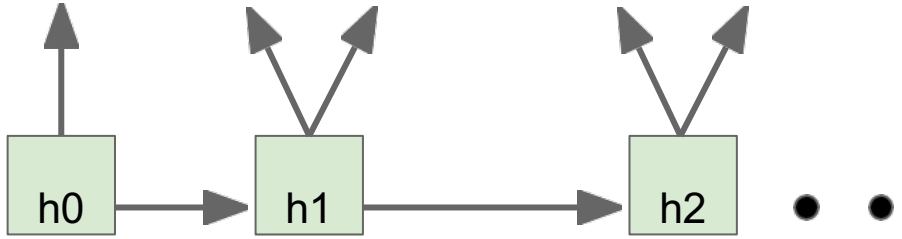


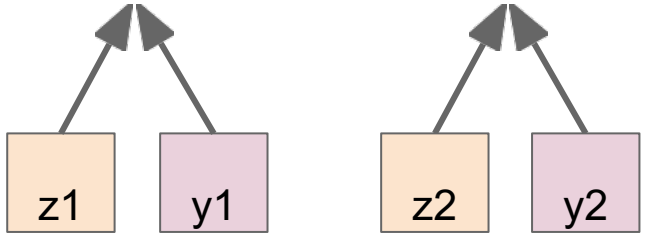
Features:
L x D

Distribution over
L locations

Distribution
over vocab







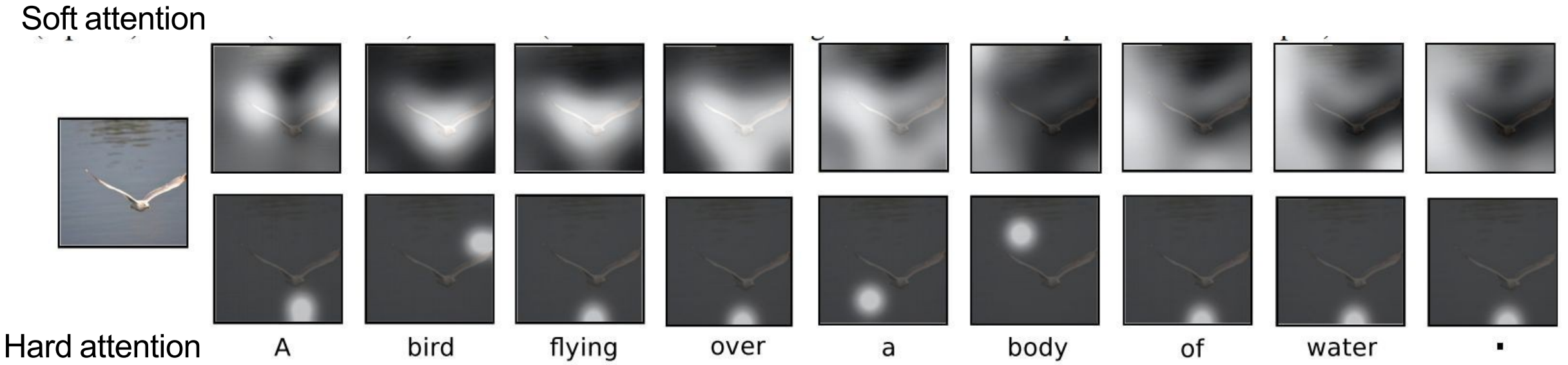
Weighted
features: D

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

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.

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?

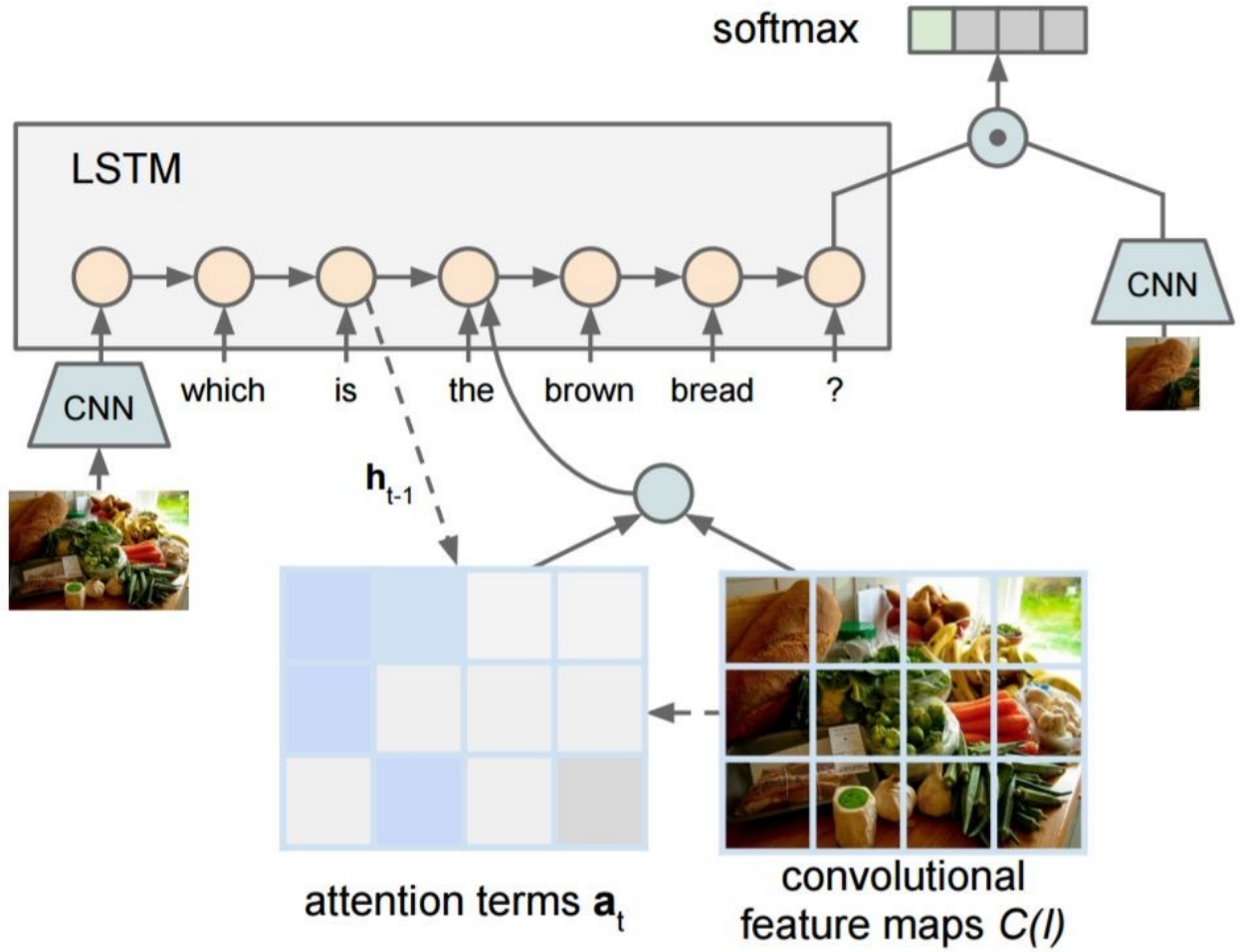
A: **During a wedding.**
A: During a bar mitzvah.
A: During a funeral.
A: During a Sunday church service

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



What kind of animal is in the photo?

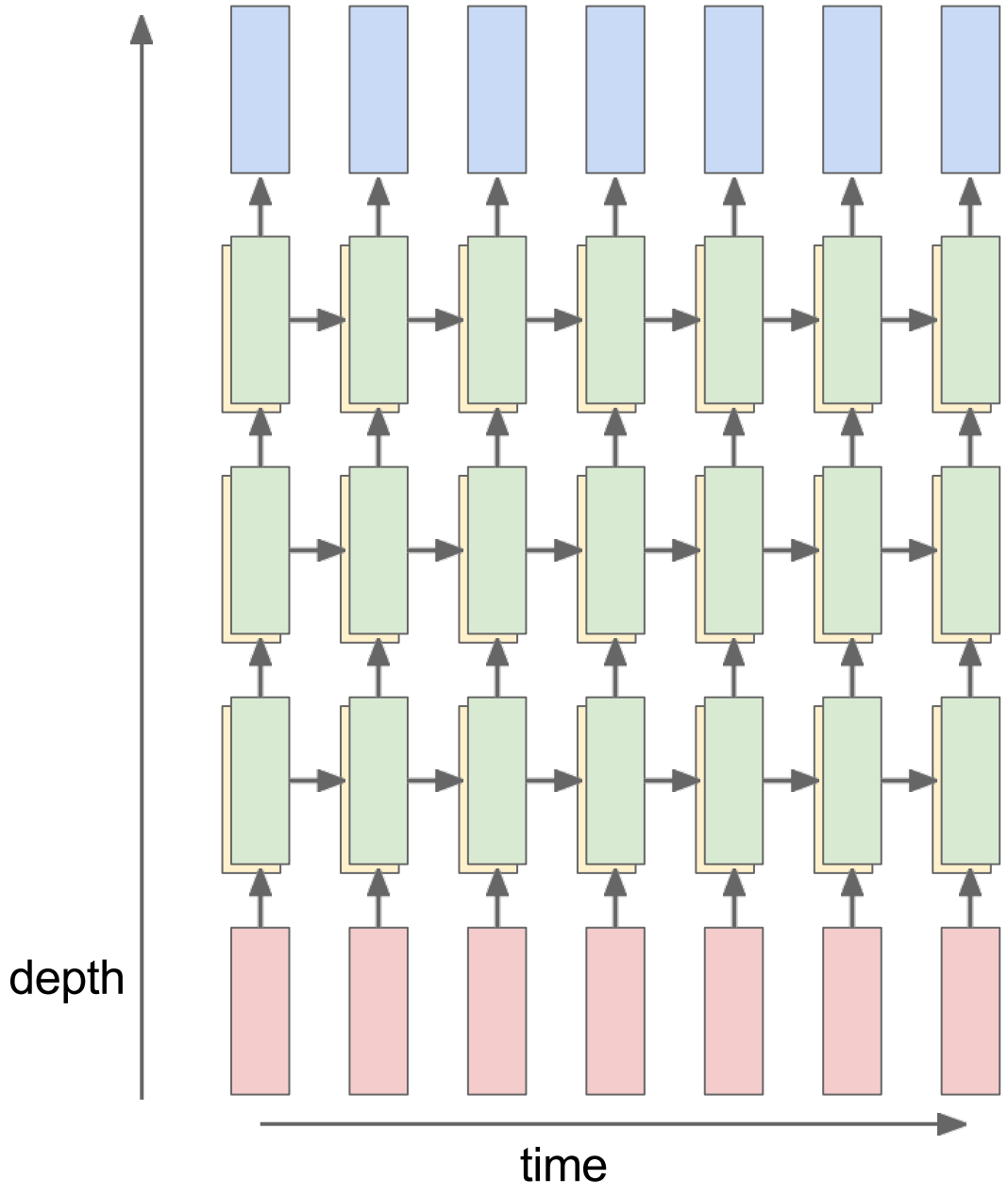
A **cat**.

Why is the person holding a knife?

To cut the **cake** with.

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

Multilayer RNNs



Long Short Term Memory (LSTM)

Vanilla RNN

$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

LSTM

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

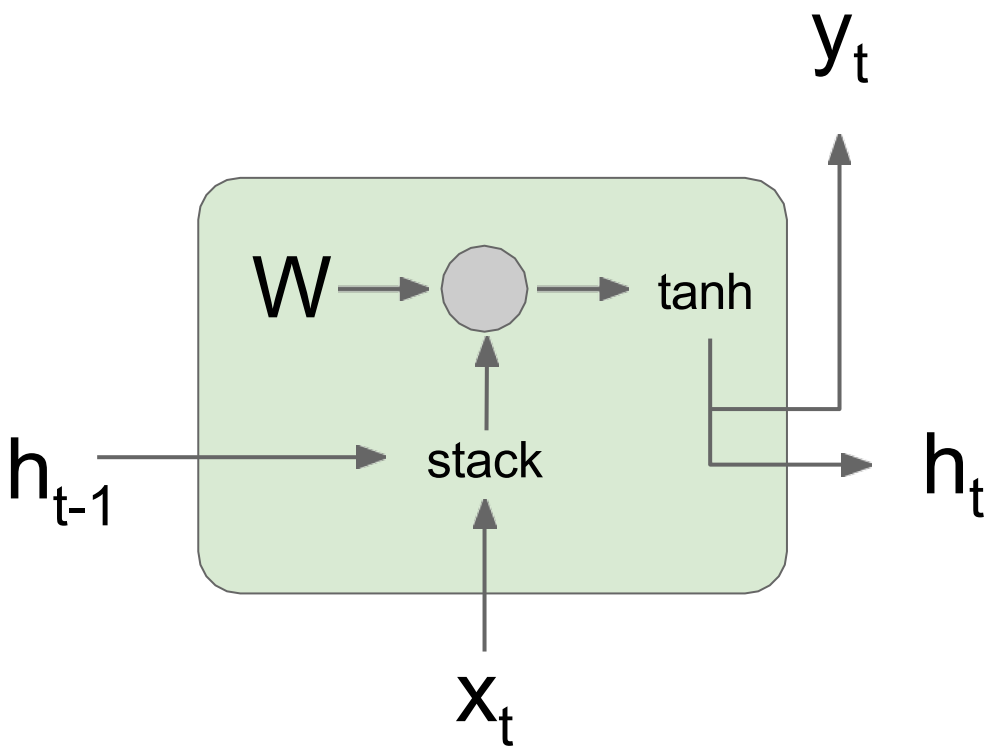
$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Hochreiter and Schmidhuber, “Long Short Term Memory”, Neural Computation 1997

Vanilla RNN Gradient Flow

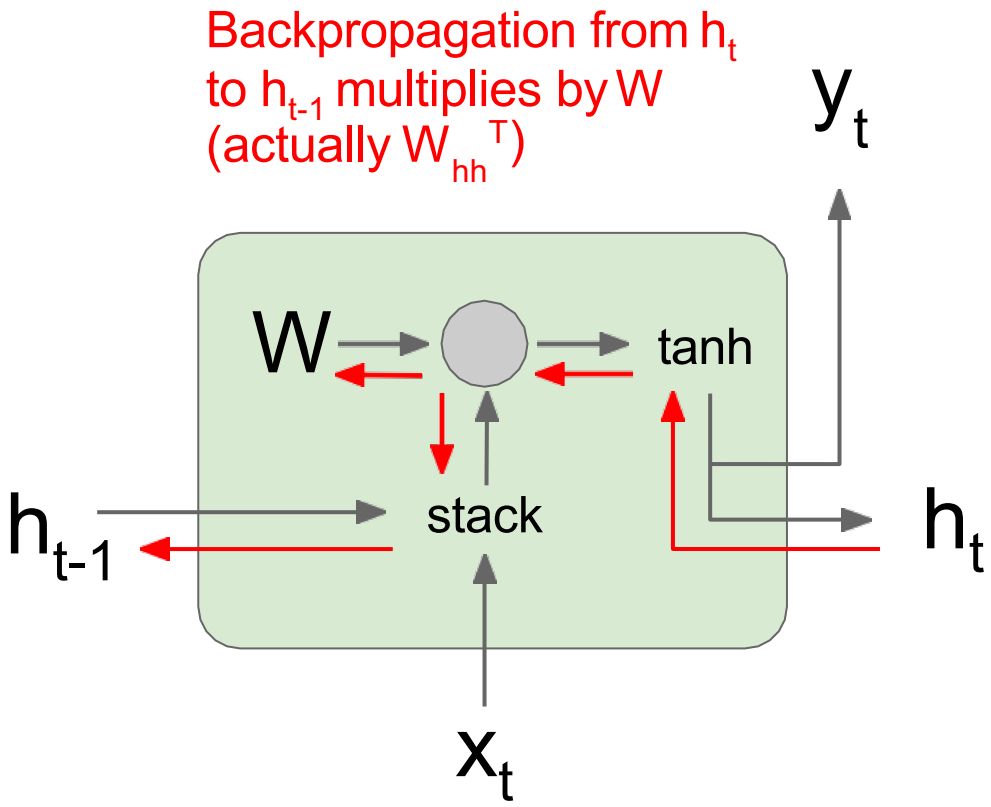
Bengio et al, “Learning long-term dependencies with gradient descent is difficult”, IEEE Transactions on Neural Networks, 1994
Pascanu et al, “On the difficulty of training recurrent neural networks”, ICML 2013



$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$
$$= \tanh\left((W_{hh} \quad W_{hx})\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$
$$= \tanh\left(W\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

Vanilla RNN Gradient Flow

Bengio et al, “Learning long-term dependencies with gradient descent is difficult”, IEEE Transactions on Neural Networks, 1994
Pascanu et al, “On the difficulty of training recurrent neural networks”, ICML 2013



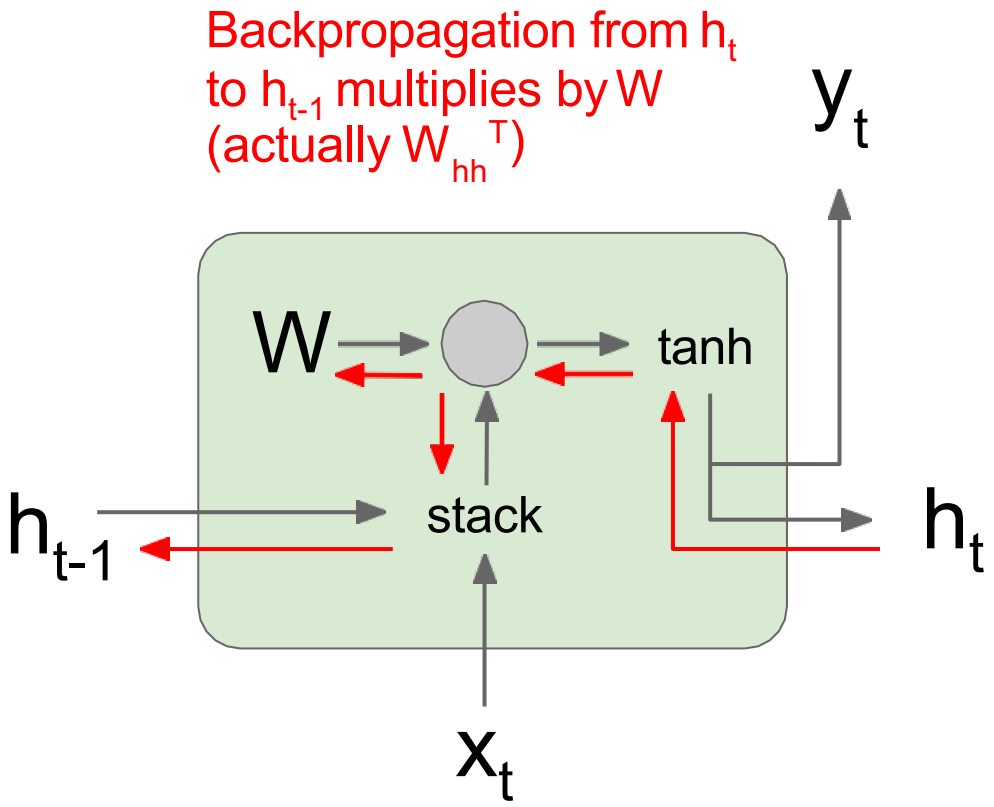
$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$= \tanh\left((W_{hh} \quad W_{hx})\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

$$= \tanh\left(W\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

Vanilla RNN Gradient Flow

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Pascanu et al, "On the difficulty of training recurrent neural networks", ICML 2013

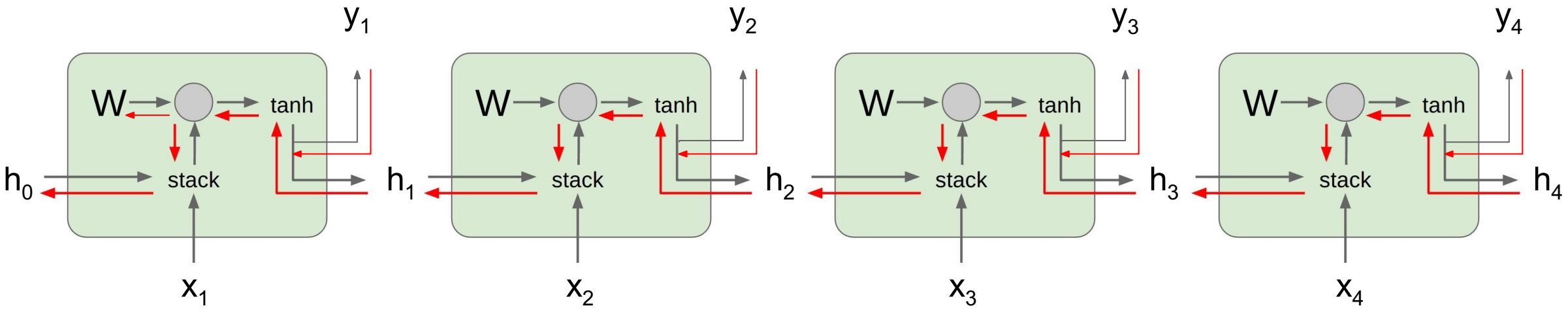


$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$
$$= \tanh\left((W_{hh} \quad W_{hx})\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$
$$= \tanh\left(W\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

$$\frac{\partial h_t}{\partial h_{t-1}} = tanh'(W_{hh}h_{t-1} + W_{xh}x_t)W_{hh}$$

Vanilla RNN Gradient Flow

Bengio et al, “Learning long-term dependencies with gradient descent is difficult”, IEEE Transactions on Neural Networks, 1994
Pascanu et al, “On the difficulty of training recurrent neural networks”, ICML 2013

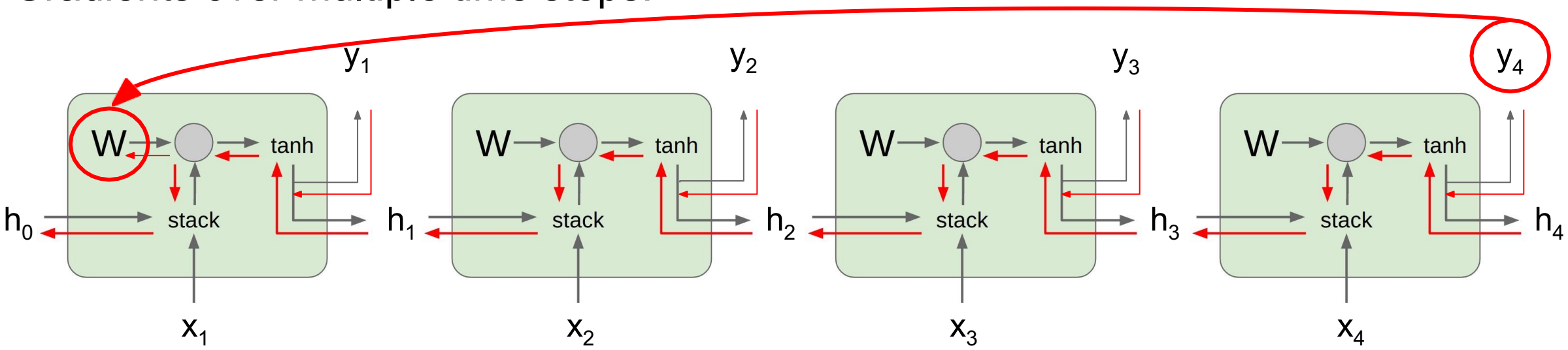


$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

Vanilla RNN Gradient Flow

Bengio et al, “Learning long-term dependencies with gradient descent is difficult”, IEEE Transactions on Neural Networks, 1994
Pascanu et al, “On the difficulty of training recurrent neural networks”, ICML 2013

Gradients over multiple time steps:



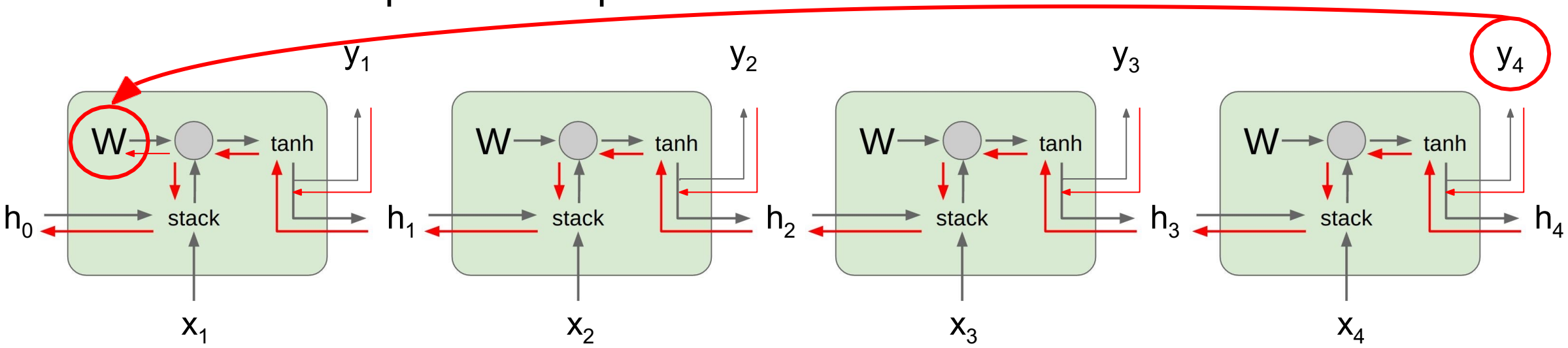
$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \frac{\partial h_t}{\partial h_{t-1}} \cdots \frac{\partial h_1}{\partial W}$$

Vanilla RNN Gradient Flow

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Pascanu et al, "On the difficulty of training recurrent neural networks", ICML 2013

Gradients over multiple time steps:



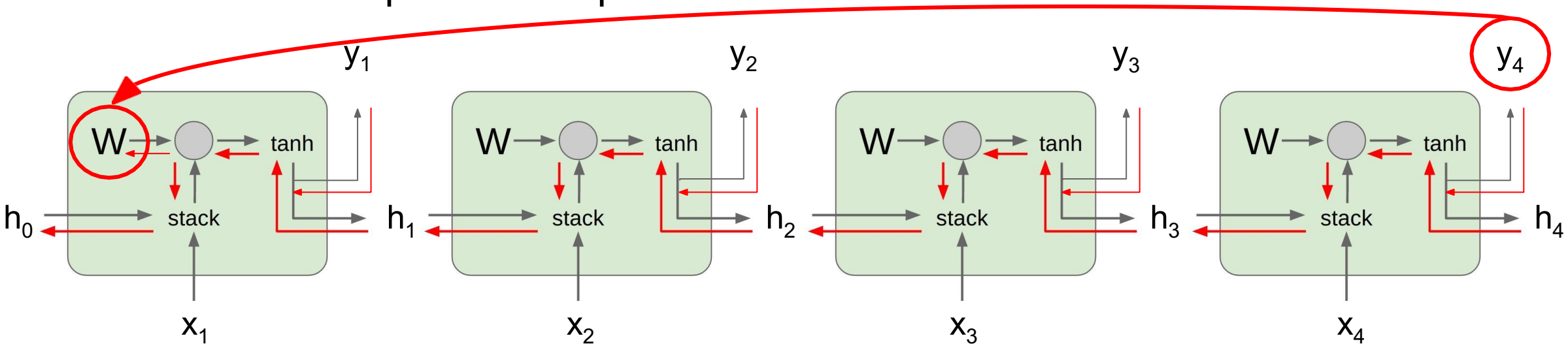
$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \frac{\partial h_t}{\partial h_{t-1}} \cdots \frac{\partial h_1}{\partial W} = \frac{\partial L_T}{\partial h_T} \left(\prod_{t=2}^{T} \frac{\partial h_t}{\partial h_{t-1}} \right) \frac{\partial h_1}{\partial W}$$

Vanilla RNN Gradient Flow

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Pascanu et al, "On the difficulty of training recurrent neural networks", ICML 2013

Gradients over multiple time steps:



$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

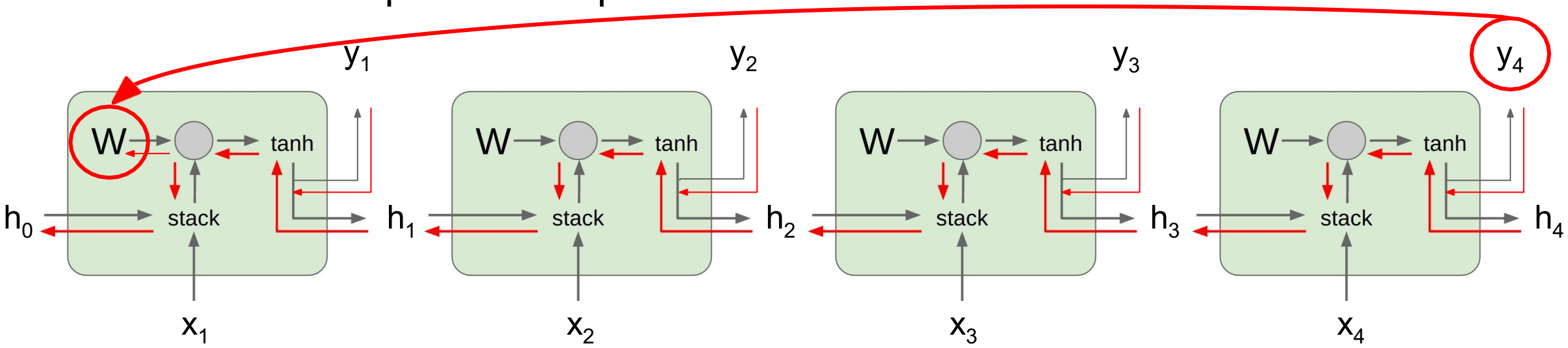
$$\frac{\partial h_t}{\partial h_{t-1}} = tanh'(W_{hh} h_{t-1} + W_{xh} x_t) W_{hh}$$

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \frac{\partial h_t}{\partial h_{t-1}} \cdots \frac{\partial h_1}{\partial W} = \frac{\partial L_T}{\partial h_T} \left(\prod_{t=2}^{T} \frac{\partial h_t}{\partial h_{t-1}} \right) \frac{\partial h_1}{\partial W}$$

Vanilla RNN Gradient Flow

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Pascanu et al, "On the difficulty of training recurrent neural networks", ICML 2013

Gradients over multiple time steps:



$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

Almost always < 1

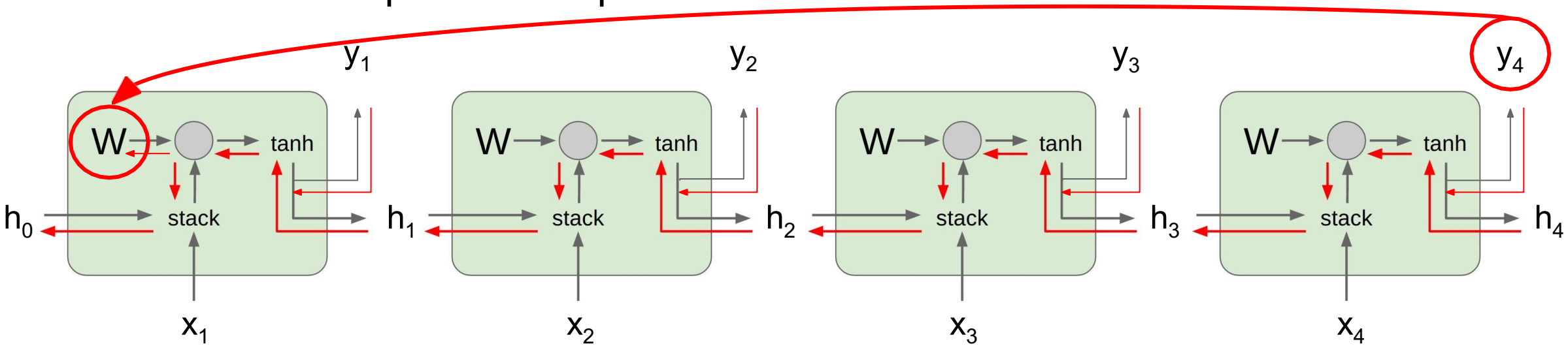
Vanishing gradients

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \left(\prod_{t=2}^{T} tanh'\left(W_{hh} h_{t-1} + W_{xh} x_t\right) \right) W_{hh}^{T-1} \frac{\partial h_1}{\partial W}$$

Vanilla RNN Gradient Flow

Bengio et al, “Learning long-term dependencies with gradient descent is difficult”, IEEE Transactions on Neural Networks, 1994
Pascanu et al, “On the difficulty of training recurrent neural networks”, ICML 2013

Gradients over multiple time steps:



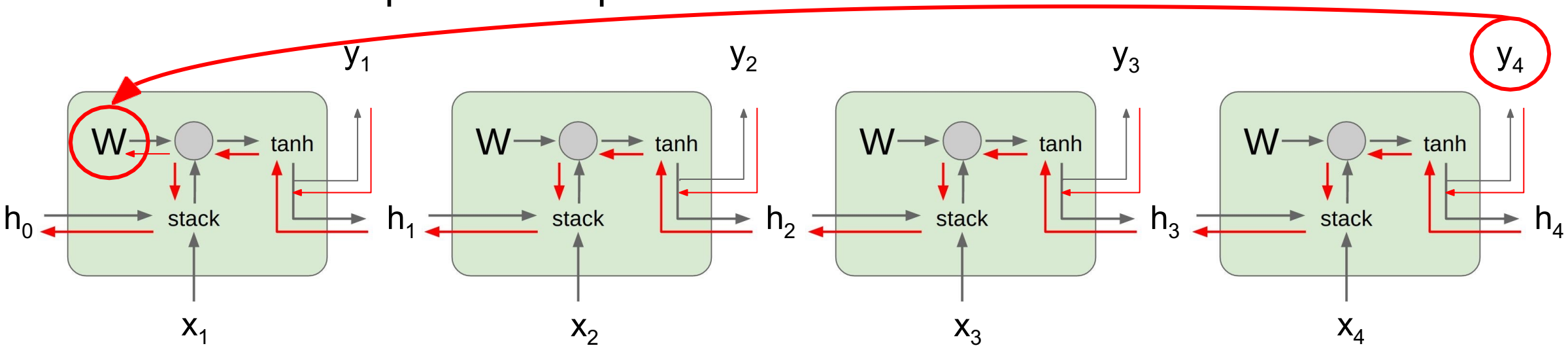
$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

What if we assumed no non-linearity?

Vanilla RNN Gradient Flow

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Pascanu et al, "On the difficulty of training recurrent neural networks", ICML 2013

Gradients over multiple time steps:



What if we assumed no non-linearity?

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} W_{in}^{T-1} \frac{\partial h_1}{\partial W}$$

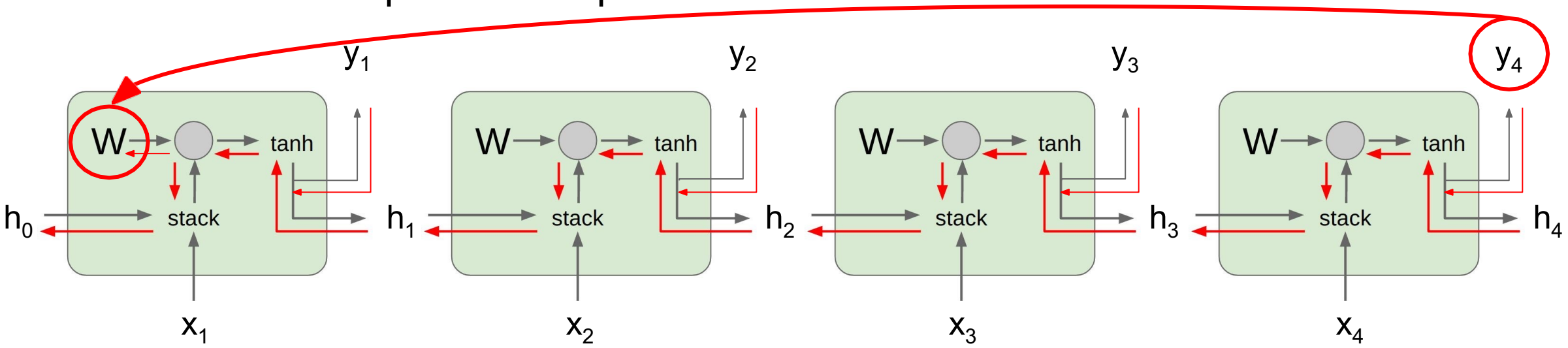
Largest singular value > 1:
Exploding gradients

Largest singular value < 1:
Vanishing gradients

Vanilla RNN Gradient Flow

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Pascanu et al, "On the difficulty of training recurrent neural networks", ICML 2013

Gradients over multiple time steps:



What if we assumed no non-linearity?

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} W_{hh}^{T-1} \frac{\partial h_1}{\partial W}$$

Largest singular value > 1:
Exploding gradients

→ **Gradient clipping**: Scale gradient if its norm is too big

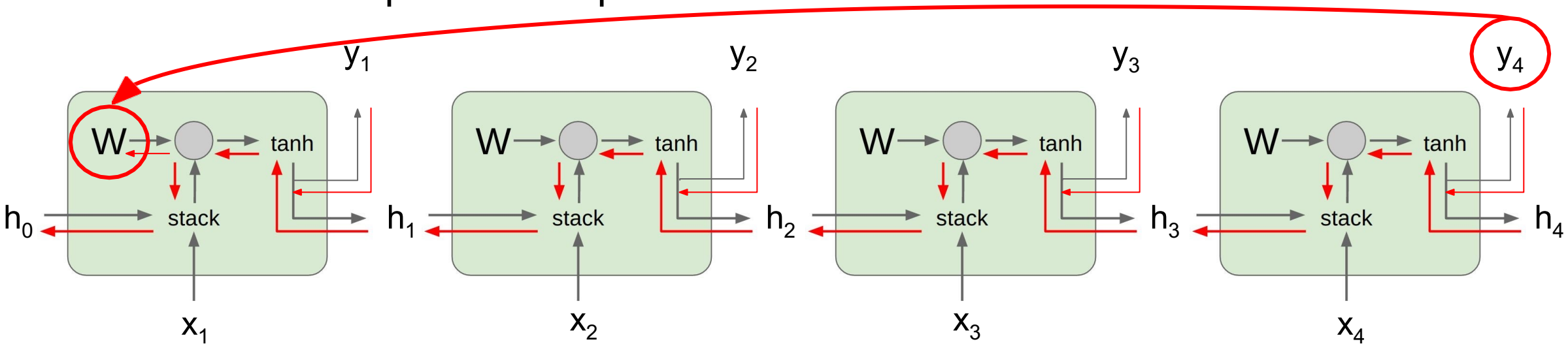
```
grad_norm = np.sum(grad * grad)
if grad_norm > threshold:
  grad *= (threshold / grad_norm)
```

Largest singular value < 1:
Vanishing gradients

Vanilla RNN Gradient Flow

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Pascanu et al, "On the difficulty of training recurrent neural networks", ICML 2013

Gradients over multiple time steps:



What if we assumed no non-linearity?

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} W_{in}^{T-1} \frac{\partial h_1}{\partial W}$$

Largest singular value > 1:
Exploding gradients

Largest singular value < 1:
Vanishing gradients

→ Change RNN architecture

RNN tradeoffs

RNN Advantages:

- Can process any length input
- Computation for step *t* can (in theory) use information from many steps back
- Model size doesn't increase for longer input
- Same weights applied on every timestep, so there is symmetry in how inputs are processed.

RNN Disadvantages:

- Recurrent computation is slow
- In practice, difficult to access information from many steps back

Long Short Term Memory (LSTM)

Vanilla RNN

$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

LSTM

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

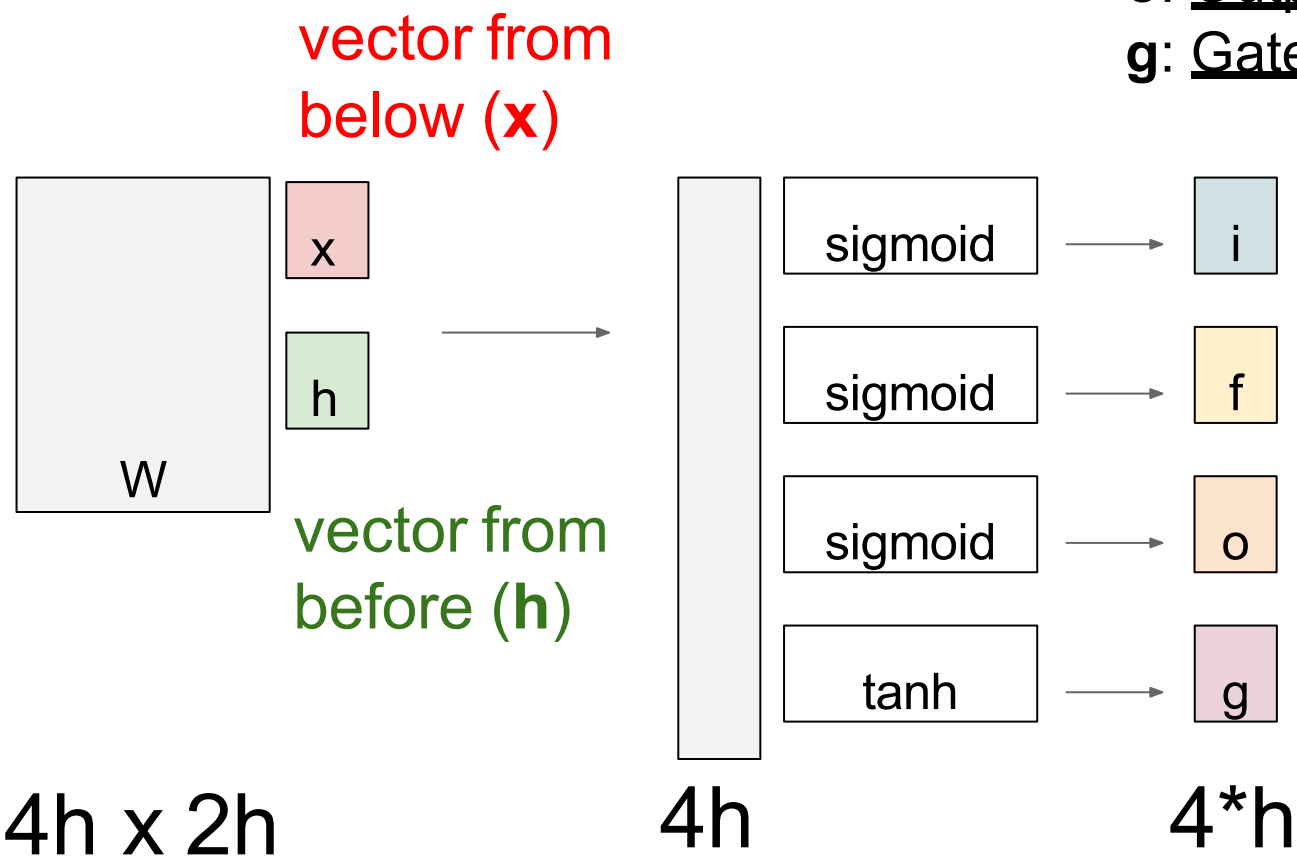
$$h_t = o \odot \tanh(c_t)$$

Hochreiter and Schmidhuber, “Long Short Term Memory”, Neural Computation 1997

Long Short Term Memory (LSTM)

[Hochreiter et al., 1997]

i: Input gate, whether to write to cell
f: Forget gate, Whether to erase cell
o: Output gate, How much to reveal cell
g: Gate gate (?), How much to write to cell



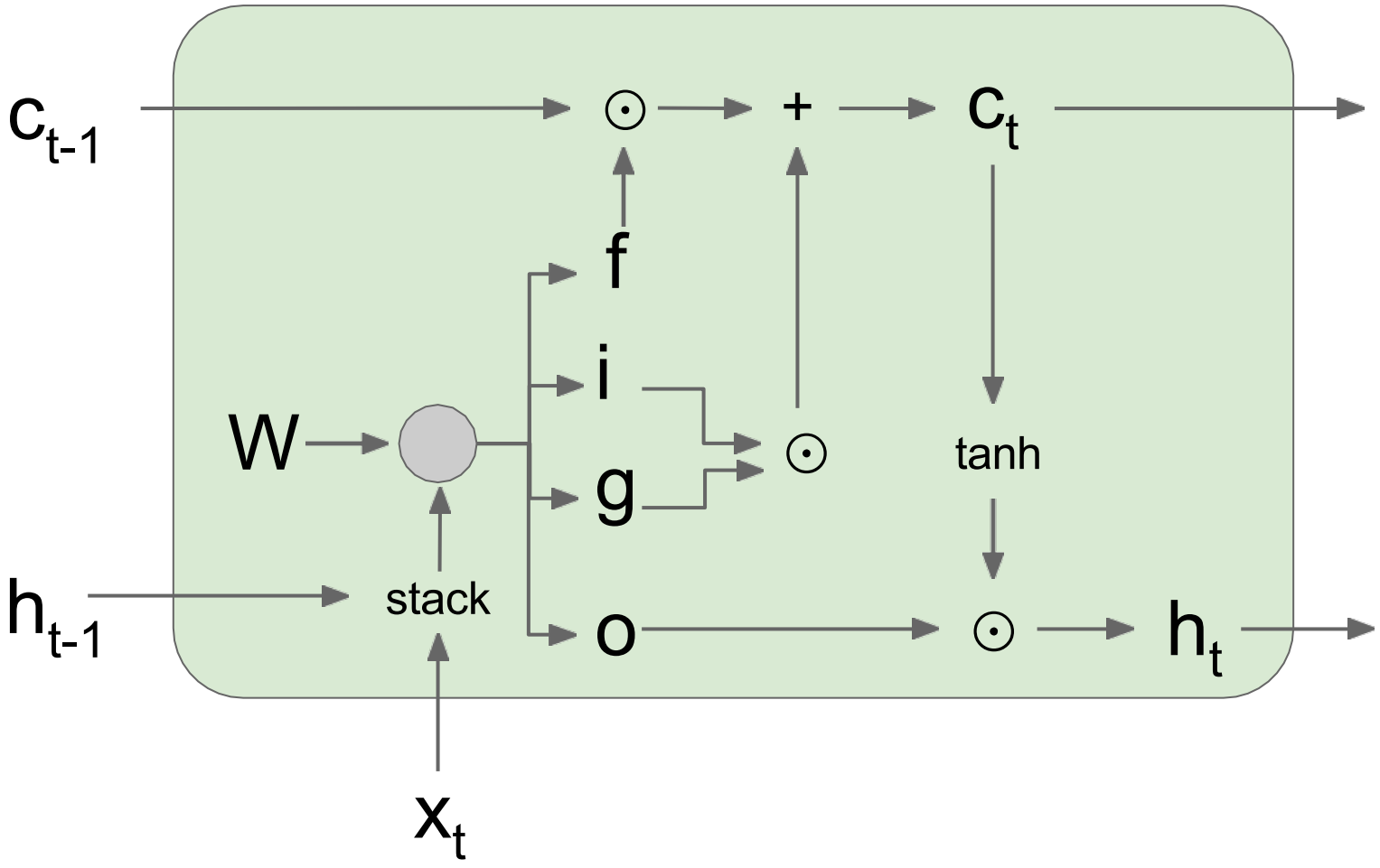
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Long Short Term Memory (LSTM)

[Hochreiter et al., 1997]



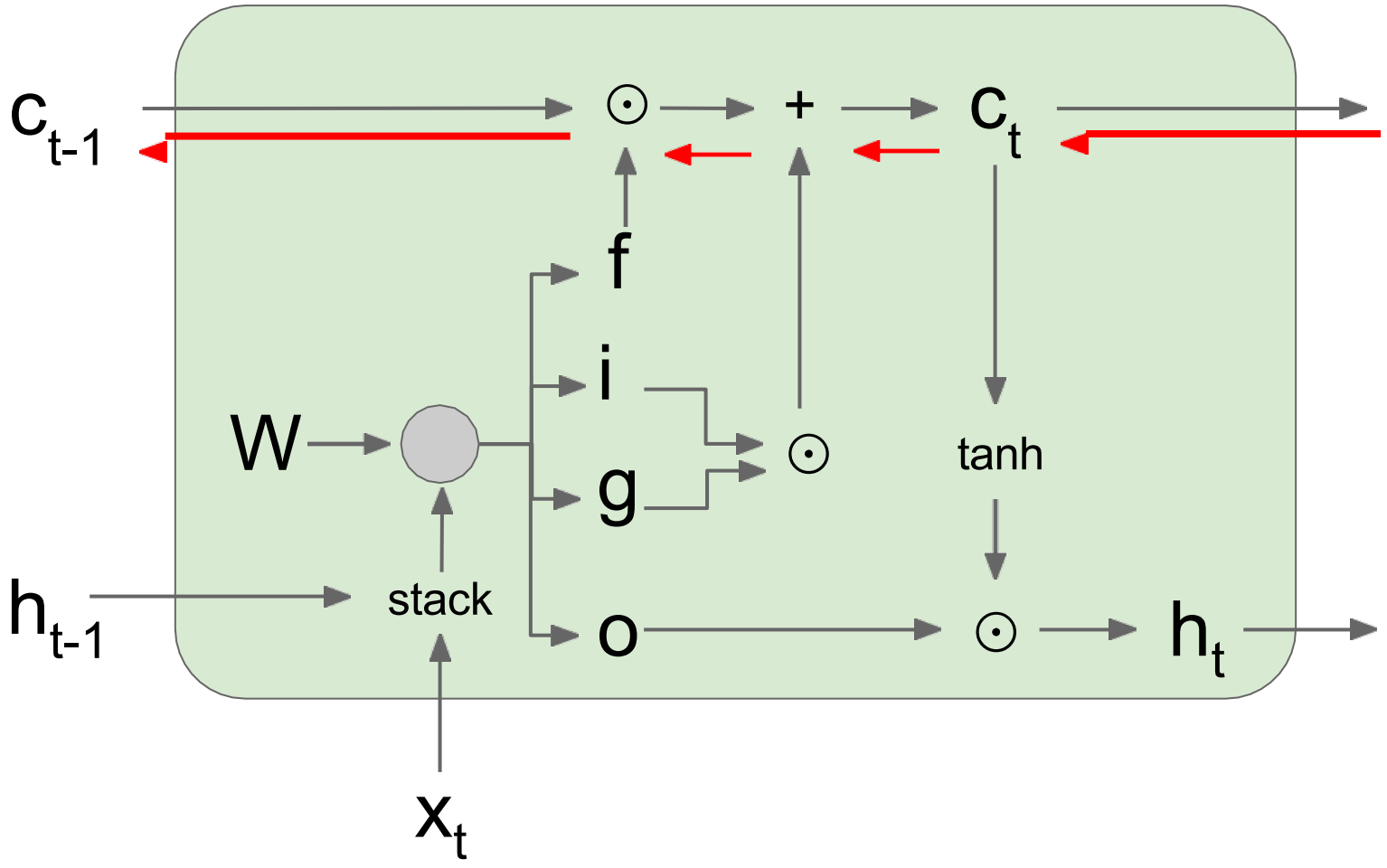
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Long Short Term Memory (LSTM): Gradient Flow

[Hochreiter et al., 1997]



Backpropagation from c_t to c_{t-1} only elementwise multiplication by f, no matrix multiply by W

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

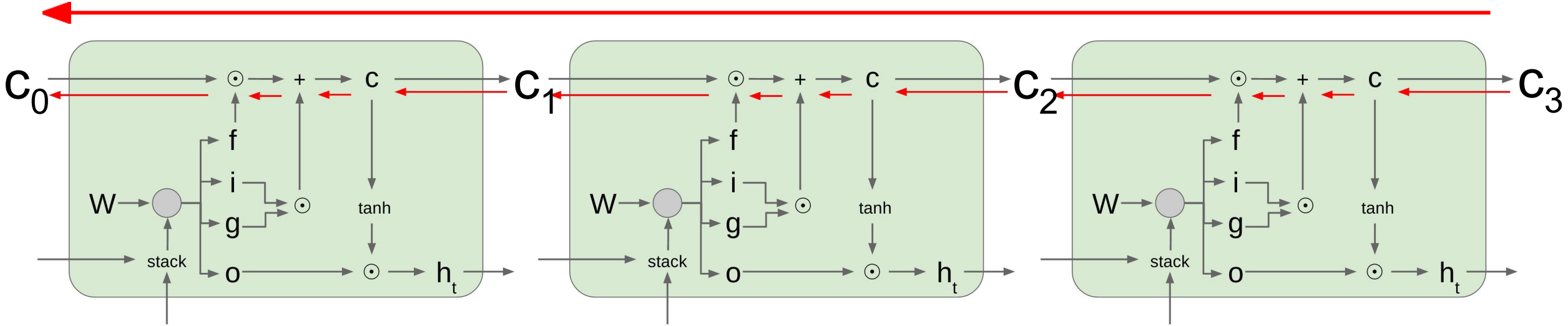
$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Long Short Term Memory (LSTM): Gradient Flow

[Hochreiter et al., 1997]

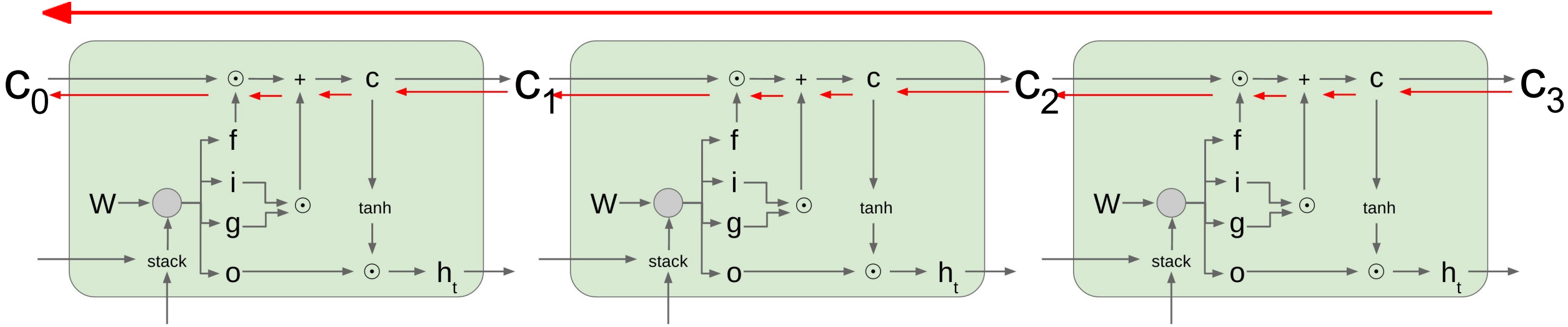
Uninterrupted gradient flow!



Long Short Term Memory (LSTM): Gradient Flow

[Hochreiter et al., 1997]

Uninterrupted gradient flow!



Notice that the gradient contains the **f** gate's vector of activations

- allows better control of gradients values, using suitable parameter updates of the forget gate.

Also notice that more control is added through the **f, i, g,** and **o** gates

- better balancing of gradient values

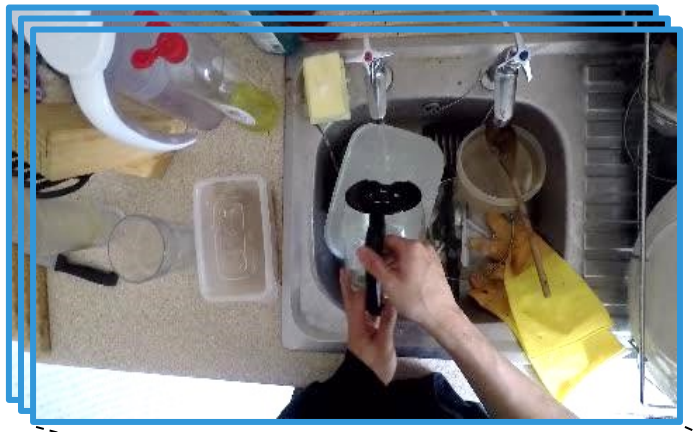
Do LSTMs solve the vanishing gradient problem?

The LSTM architecture makes it easier for the RNN to preserve information over many timesteps

- e.g. if the f = 1 and the i = 0, then the information of that cell is preserved indefinitely.
- By contrast, it's harder for vanilla RNN to learn a recurrent weight matrix Wh that preserves info in hidden state
- LSTM doesn't guarantee that there is no vanishing/exploding gradient, but it does provide an easier way for the model to learn long-distance dependencies

Egocentric Action Anticipation

PAST FRAMES (observed)



"predicting the occurrence of an action before it happens"

FUTURE FRAMES (unobserved)

$\tau_s - (\tau_o + \tau_a)$ $\tau_s - \tau_a$ τ_s:action start τ_e:action end

EGOCENTRIC VIDEO →

BEFORE THE ACTION **(observed)**

ANTICIPATION TIME τ_a **(unobserved)**

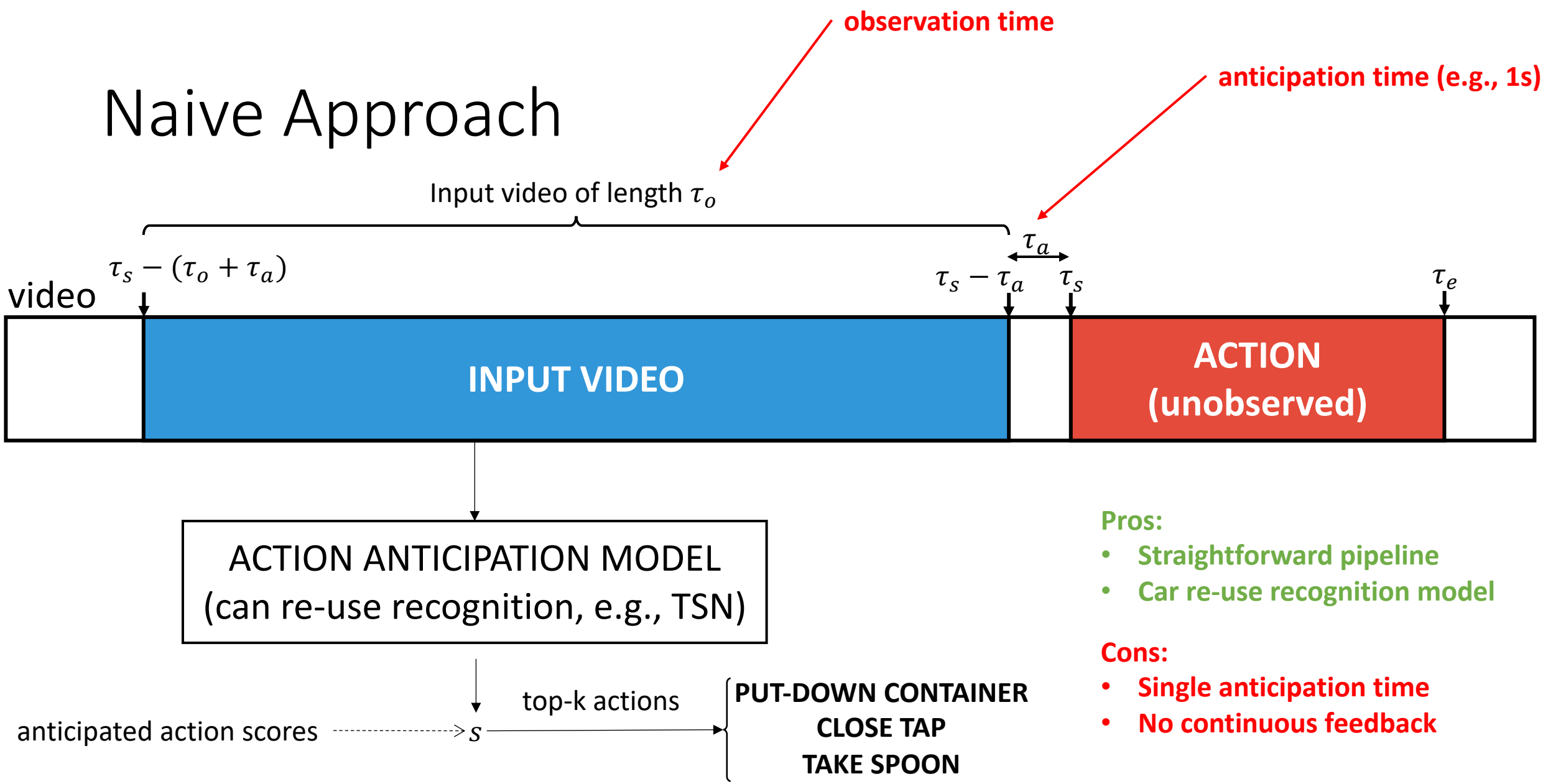
"**PUT-DOWN CONTAINER**" ACTION **(unobserved)**

MODEL → Top-k anticipated actions

PUT-DOWN CONTAINER
CLOSE TAP
TAKE SPOON

A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

Naive Approach



Pros:

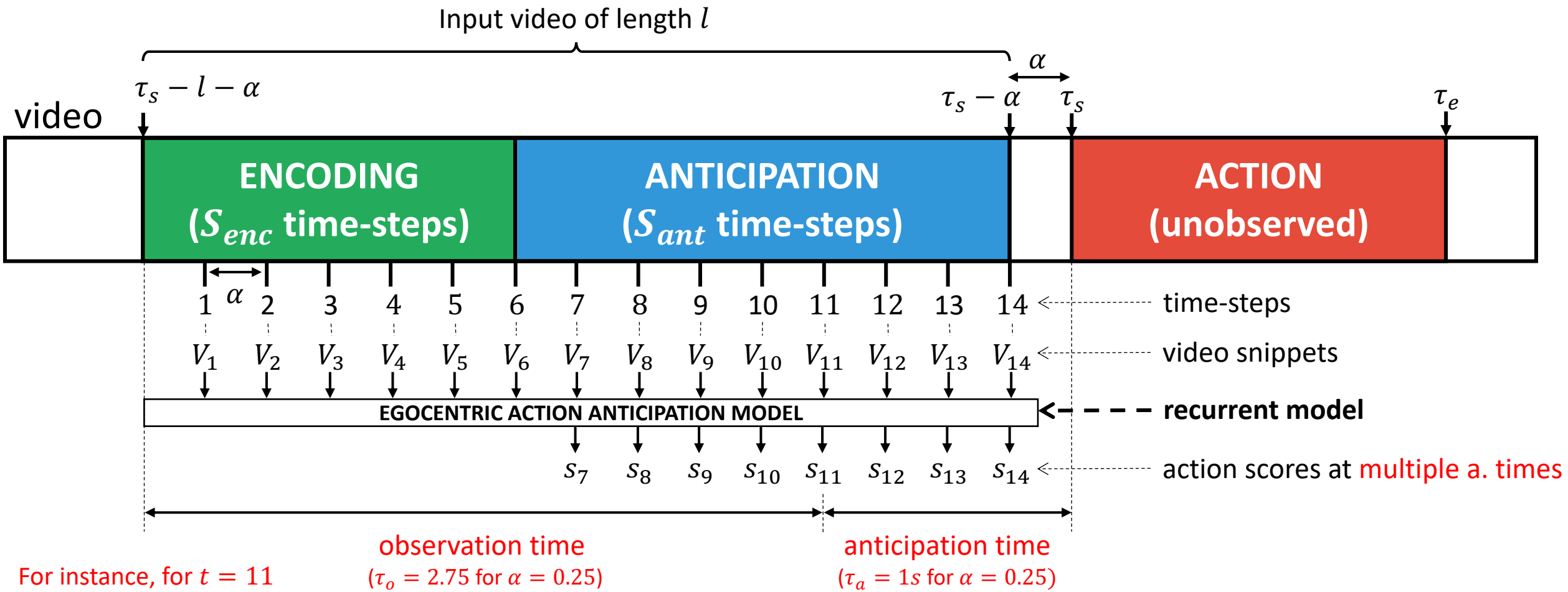
- **Straightforward pipeline**
- **Car re-use recognition model**

Cons:

- **Single anticipation time**
- **No continuous feedback**

A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

Proposed Processing Scheme

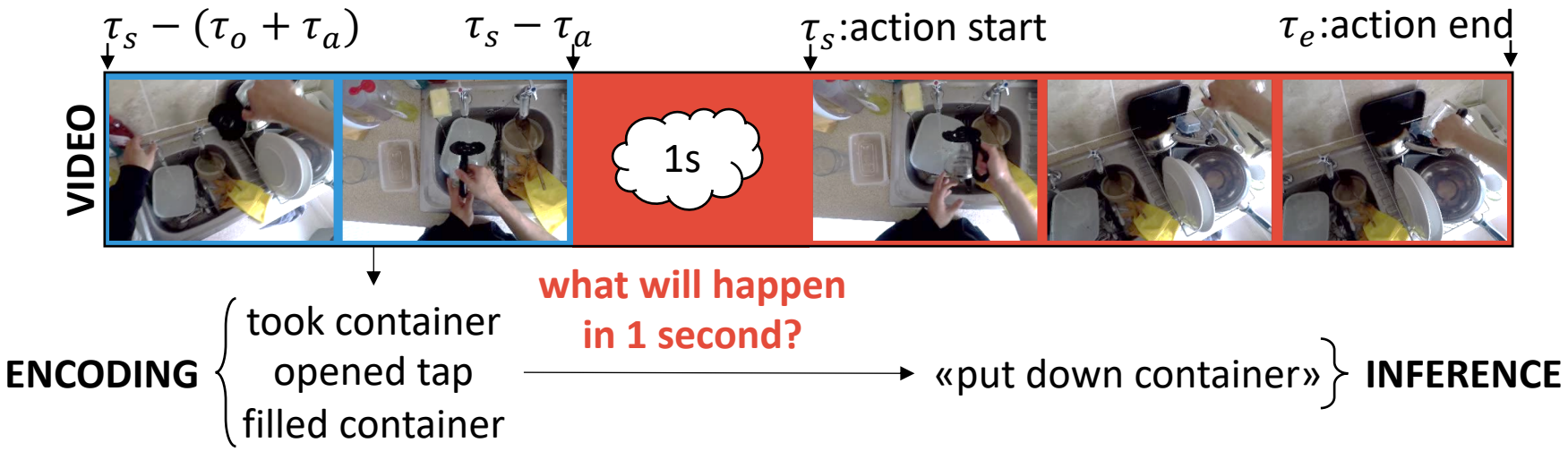


A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

Encoding vs Inference

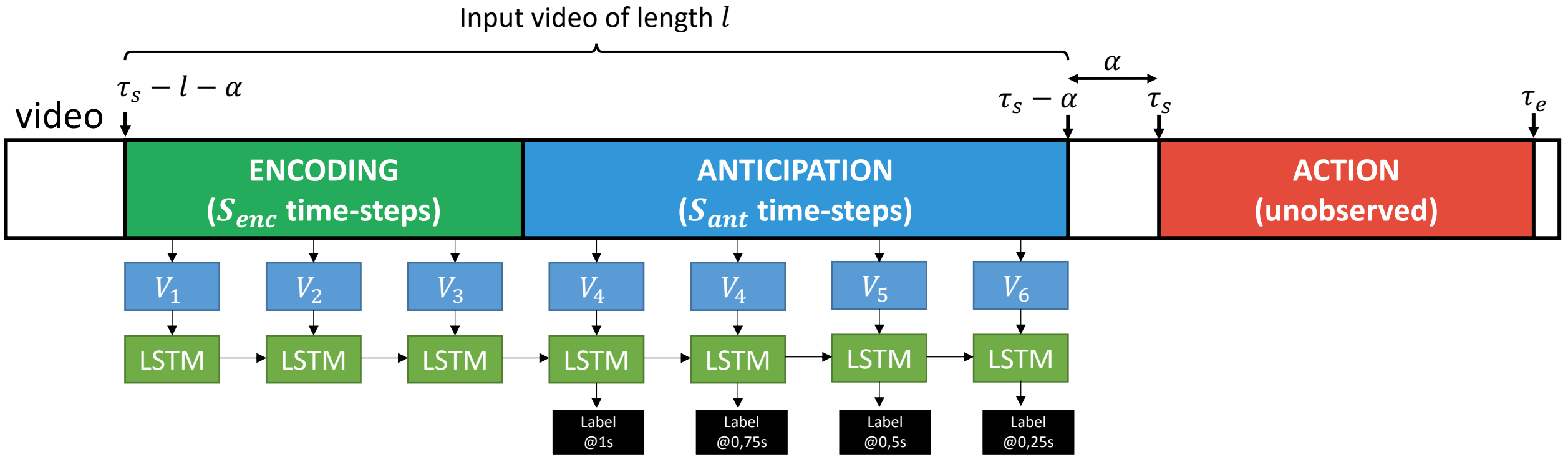
Action anticipation methods need to tackle two tasks:

- Encoding: summarizing past observations;
- Inference: anticipating what is going to happen next.



A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

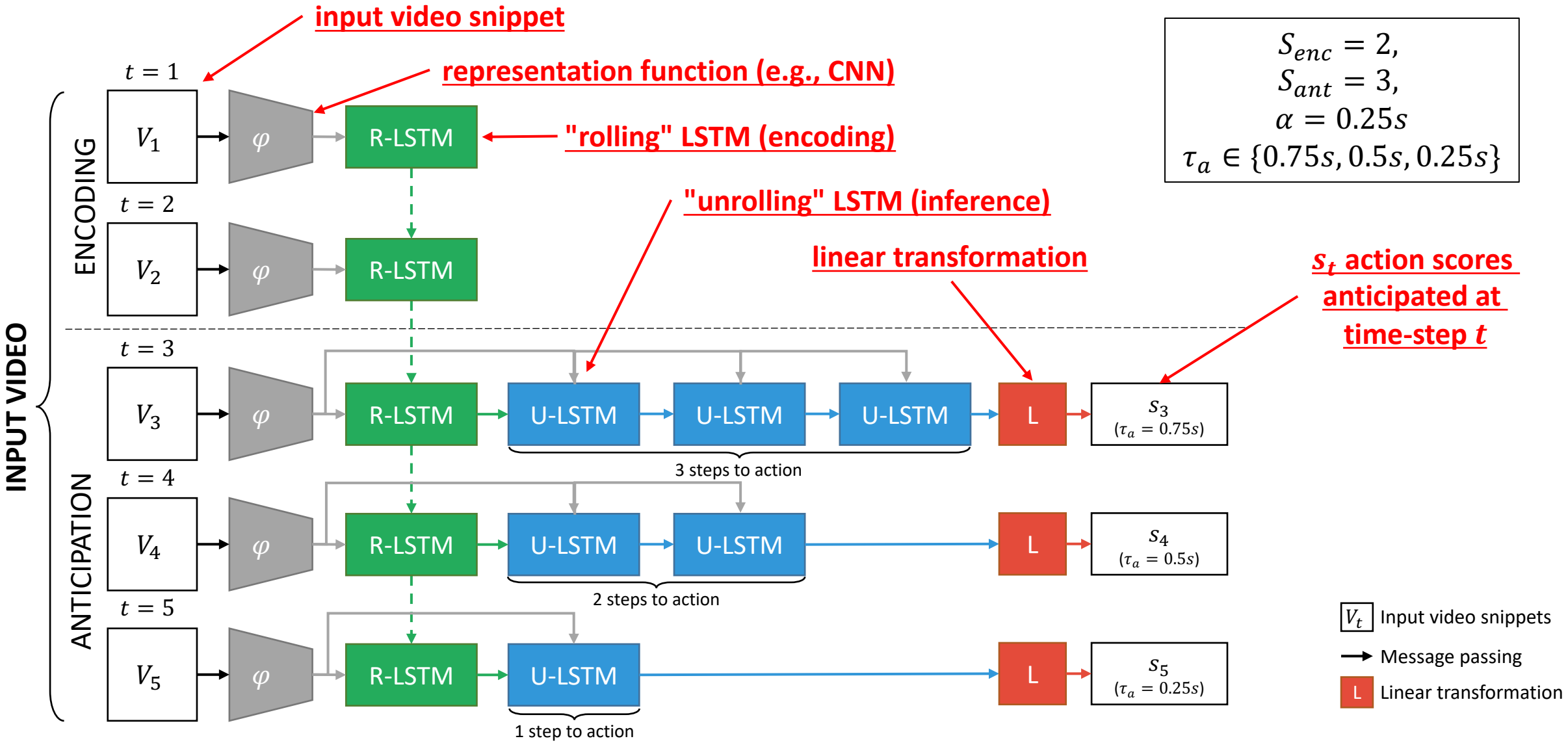
Classic Scheme: Join Encoding and Inference



LSTM **is responsible for both encoding streaming observations and infer future actions**

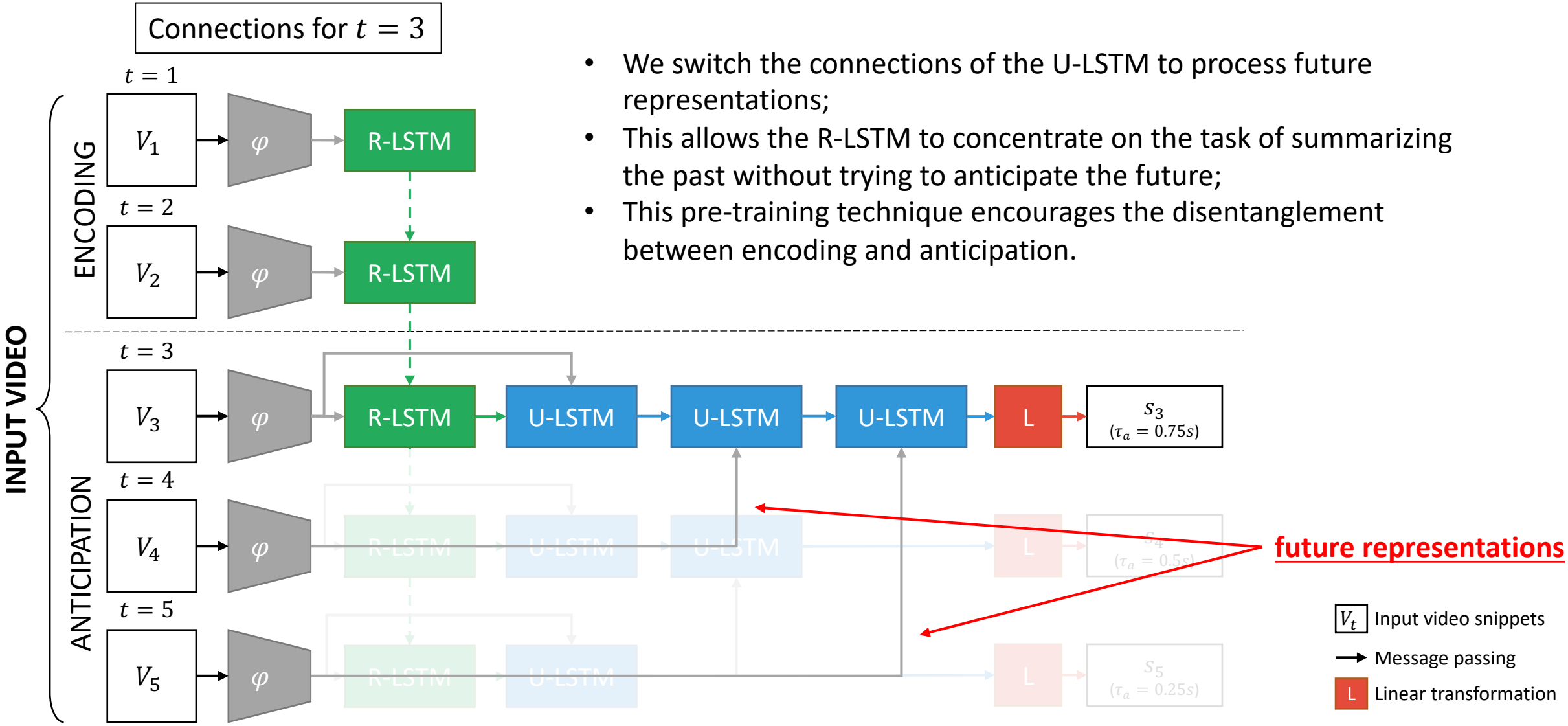
A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

Proposed Approach: Rolling-Unrolling LSTM



A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

Sequence Completion Pre-Training

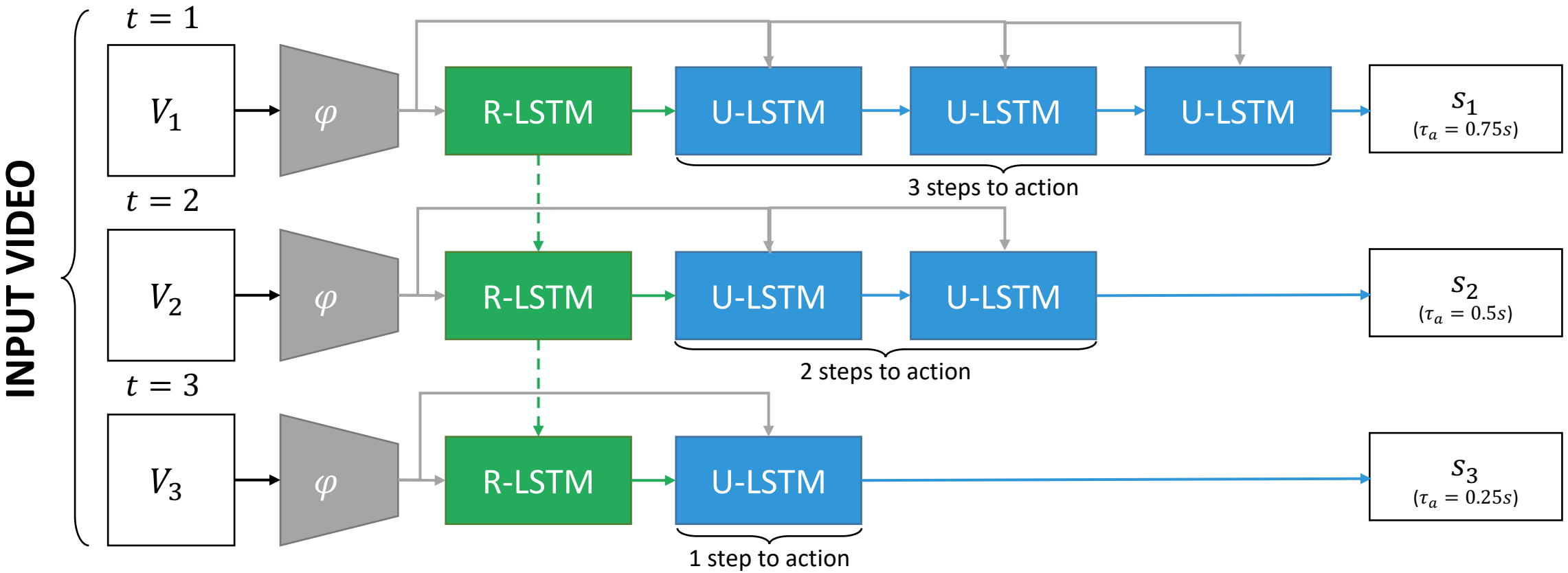


- We switch the connections of the U-LSTM to process future representations;
- This allows the R-LSTM to concentrate on the task of summarizing the past without trying to anticipate the future;
- This pre-training technique encourages the disentanglement between encoding and anticipation.

A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

Fine-Tuning for Action Anticipation

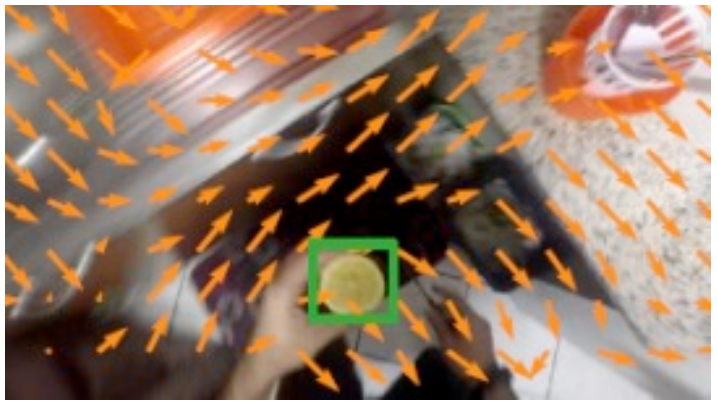
Rolling-Unrolling LSTMS, regular connections for action anticipation:



Modalities

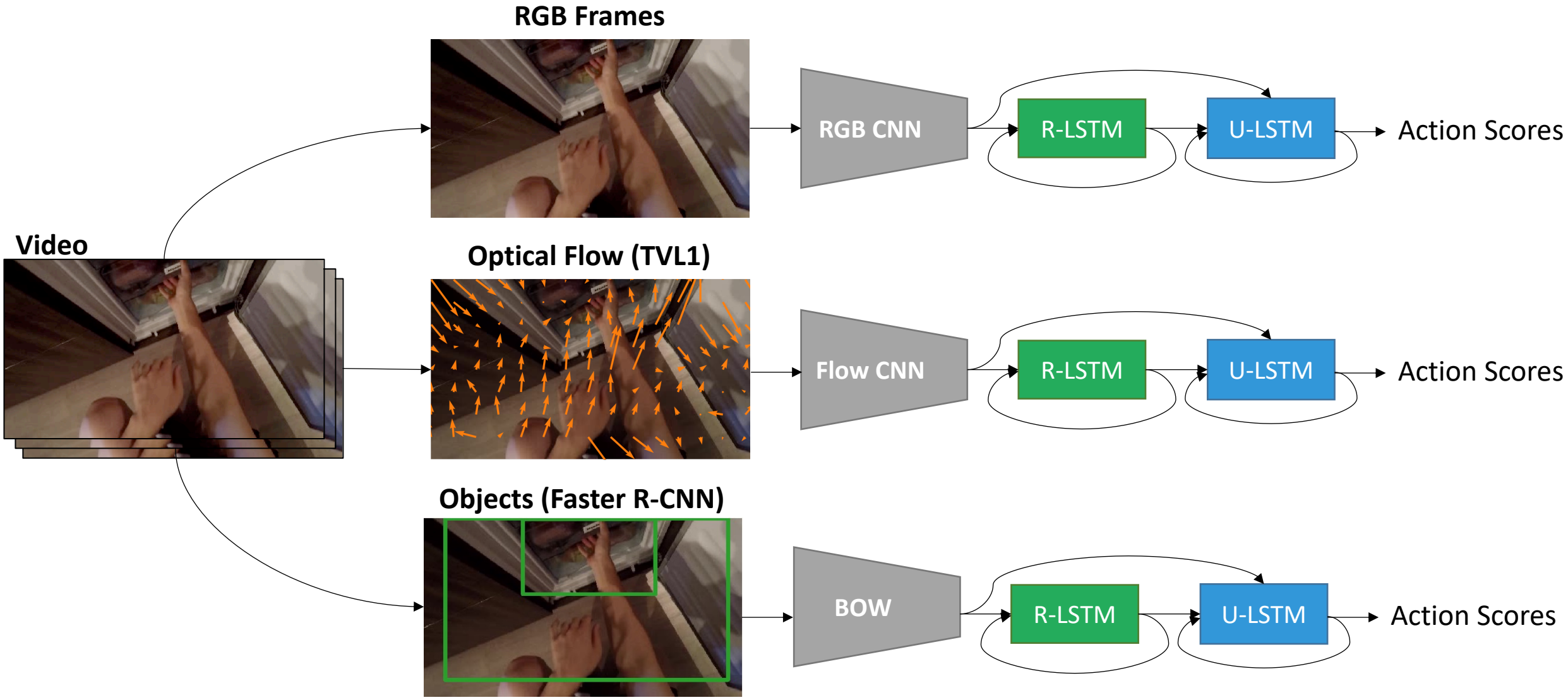
Our model has three branches:

- RGB: process RGB frames;
- Flow: processes stacks of 5 optical flows;
- OBJ: processes object-based features obtained by running an object detector.



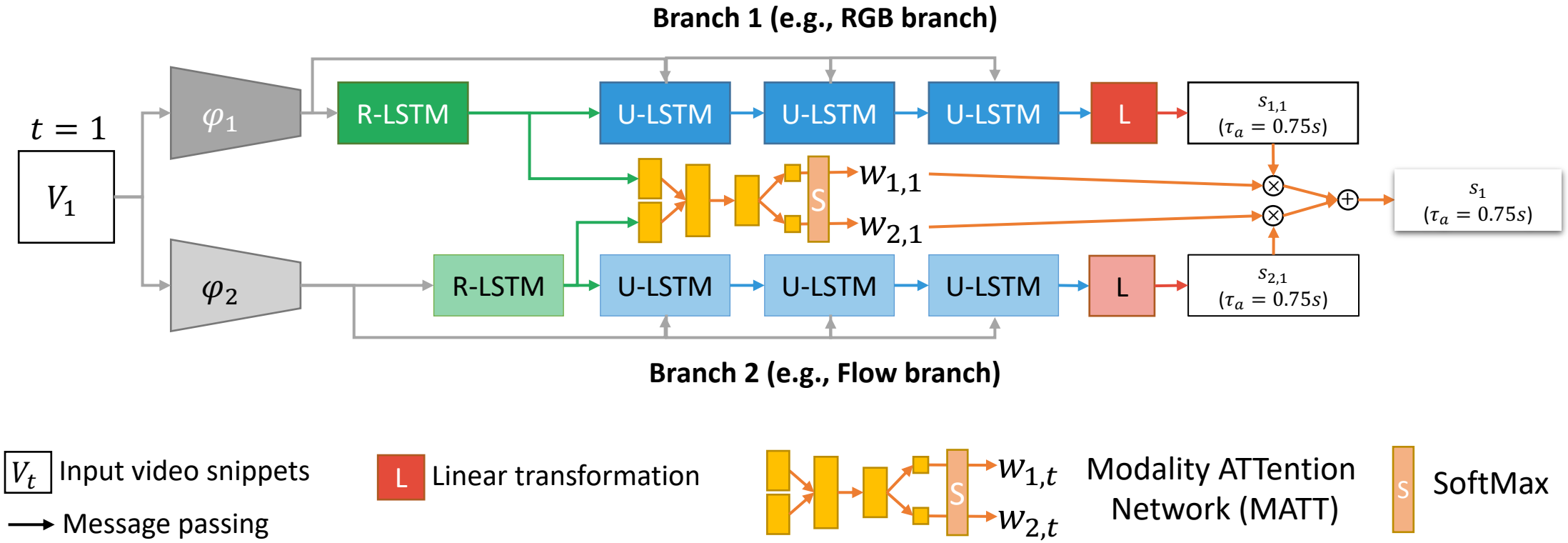
See all the details on the paper: http://iplab.dmi.unict.it/rulstm/

Three Modalities



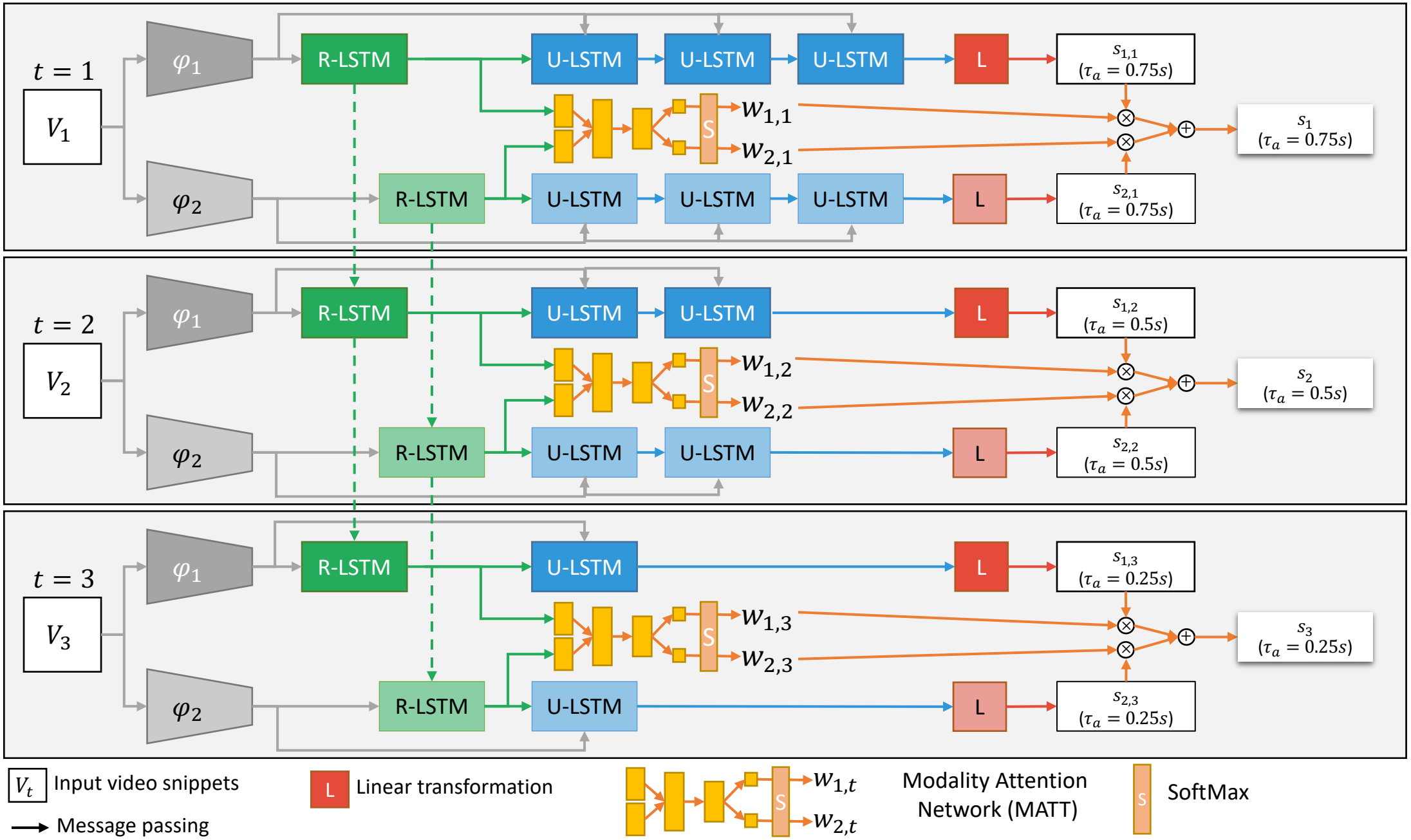
Modality ATTention (MATT)

The relative importance of each modality may depend on the observed sample.



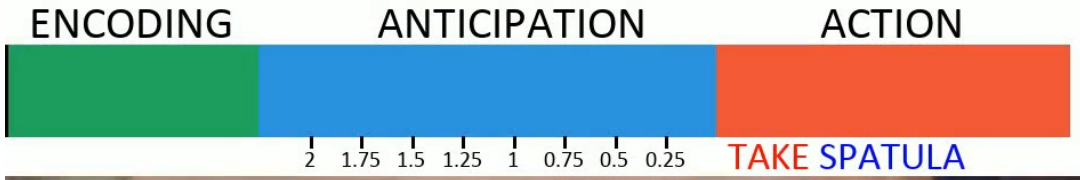
A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

OVERALL ARCHITECTURE (last three steps)



A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

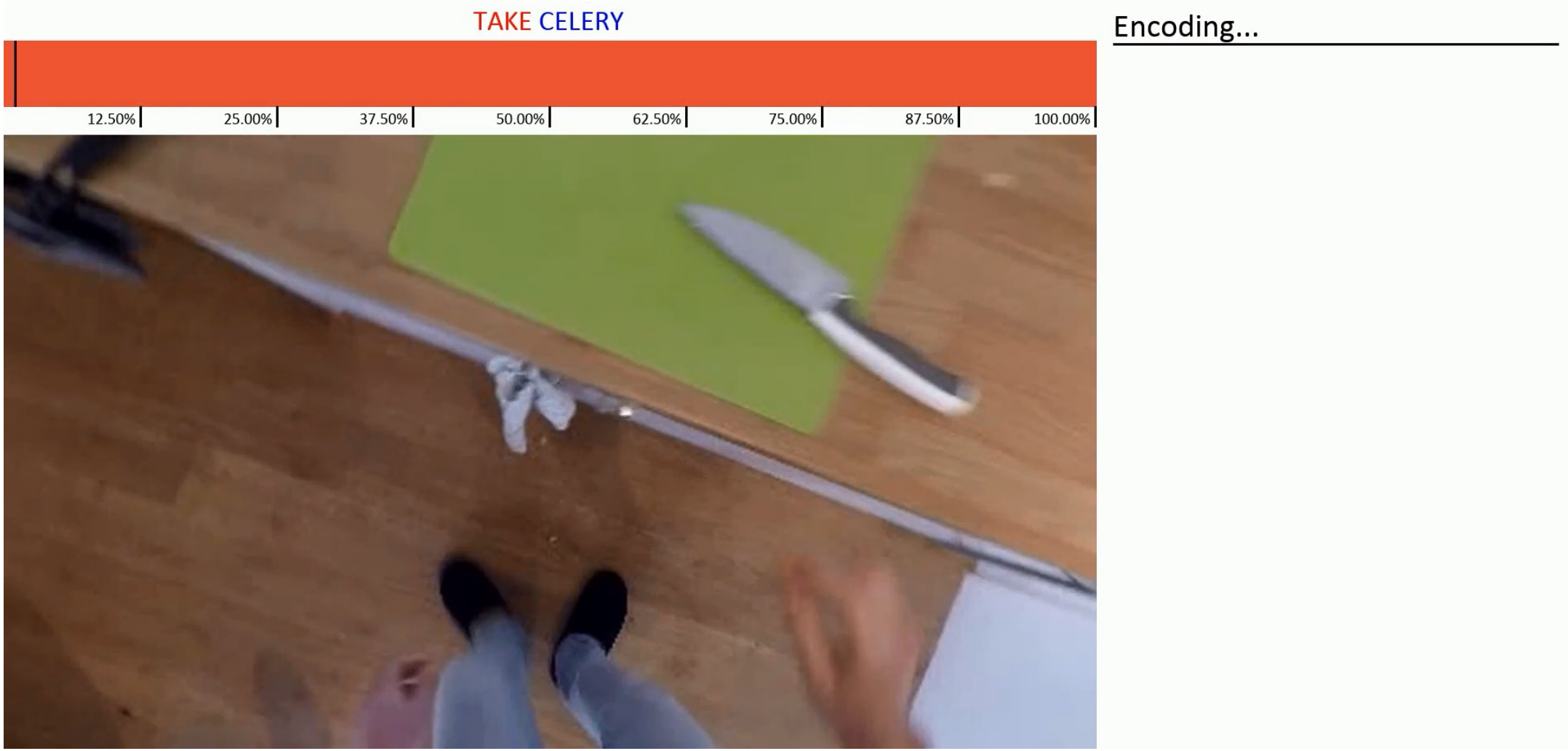
Demo Video: Egocentric Action Anticipation



Encoding...

A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

Demo Video: Early Action Recognition



A. Furnari, G. M. Farinella, Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2020. http://iplab.dmi.unict.it/rulstm/

Ablation on EPIC-KITCHENS