CSCI 566: Deep Learning and its Applications

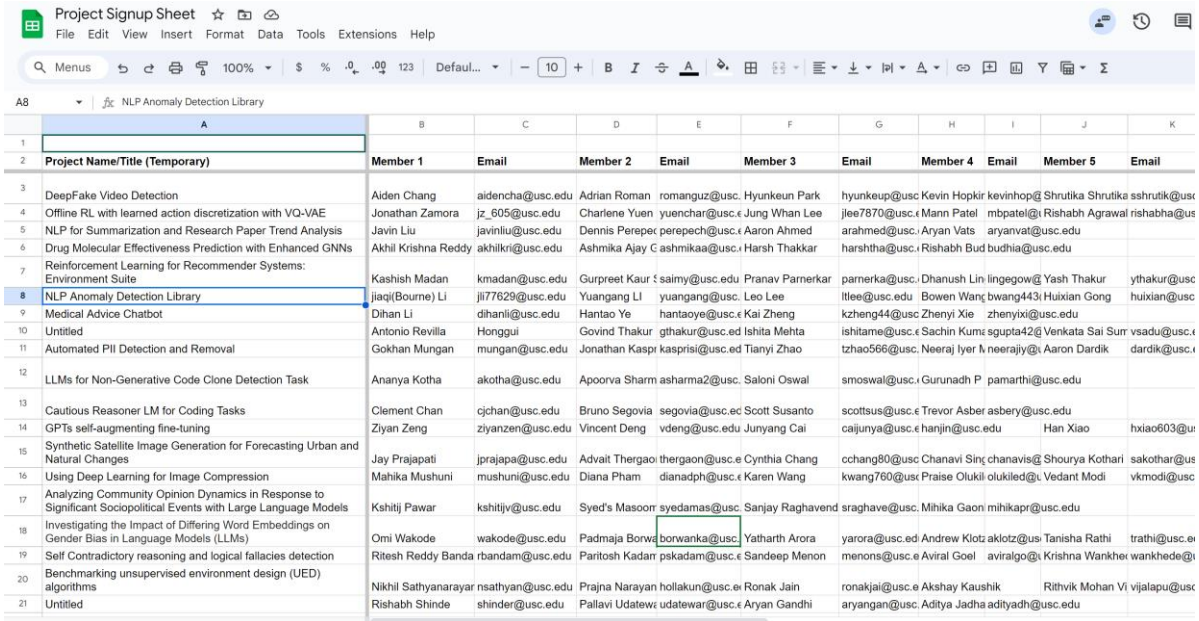
Yue Zhao
Thomas Lord Department of Computer Science
University of Southern California

Credits to previous versions of USC CSCI566, CMU 10601/701, Stanford CS 229, 231n, 224w

Special thanks to Prof. Jure Leskovec on GNN notes.

Logistics

Projects



Project Signup Sheet

File Edit View Insert Format Data Tools Extensions Help

A8 NLP Anomaly Detection Library

	Project Name/Title (Temporary)	Member 1	Email	Member 2	Email	Member 3	Email	Member 4	Email	Member 5	Email
3	DeepFake Video Detection	Aiden Chang	aidencha@usc.edu	Adrian Roman	romanguz@usc.	Hyunkeun Park	hyunkeup@usc	Kevin Hopkir	kevinhop@	Shrutika Shrutika	sshrutik@usc
4	Offline RL with learned action discretization with VQ-VAE	Jonathan Zamora	jz_605@usc.edu	Charlene Yuen	yuenchar@usc.e	Jung Whan Lee	jlee7870@usc.e	Mann Patel	mbpatel@u	Rishabh Agrawal	rishabha@us
5	NLP for Summarization and Research Paper Trend Analysis	Javin Liu	javinliu@usc.edu	Dennis Perepec	perepech@usc.e	Aaron Ahmed	arahmed@usc.	Aryan Vats	aryanvat@usc.edu		
6	Drug Molecular Effectiveness Prediction with Enhanced GNNs	Akhil Krishna Reddy	akhilkri@usc.edu	Ashmika Ajay G	ashmikaa@usc.e	Harsh Thakkar	harshtha@usc.e	Rishabh Bud	budhia@usc.edu		
7	Reinforcement Learning for Recommender Systems: Environment Suite	Kashish Madan	kmadan@usc.edu	Gurpreet Kaur S	saimy@usc.edu	Pranav Parnerkar	parnerka@usc.	Dhanush Lin	lingegow@	Yash Thakur	ythakur@usc
8	NLP Anomaly Detection Library	jiaqi(Bourne) Li	jli77629@usc.edu	Yuangang Li	yuangang@usc.	Leo Lee	ltlee@usc.edu	Bowen Wang	bwang443(	Huixian Gong	huixian@usc
9	Medical Advice Chatbot	Dihan Li	dihanli@usc.edu	Hantao Ye	hantaoye@usc.e	Kai Zheng	kzheng44@usc	Zhenyi Xie	zhenyixi@usc.edu		
10	Untitled	Antonio Revilla	Honggui	Govind Thakur	gthakur@usc.ed	Ishita Mehta	ishitame@usc.e	Sachin Kumz	sgupta42@	Venkata Sai Sum	vsadu@usc.e
11	Automated PII Detection and Removal	Gokhan Mungan	mungan@usc.edu	Jonathan Kaspr	kasprisi@usc.ed	Tianyi Zhao	tzhao566@usc.	Neeraj Iyer N	neerajiy@u	Aaron Dardik	dardik@usc.e
12	LLMs for Non-Generative Code Clone Detection Task	Ananya Kotha	akotha@usc.edu	Apoorva Sharm	asharma2@usc.	Saloni Oswal	smoswal@usc.e	Gurunadh P	pamarthi@usc.edu		
13	Cautious Reasoner LM for Coding Tasks	Clement Chan	cjchan@usc.edu	Bruno Segovia	segovia@usc.ed	Scott Susanto	scottsus@usc.e	Trevor Asber	asbery@usc.edu		
14	GPTs self-augmenting fine-tuning	Ziyan Zeng	ziyanzen@usc.edu	Vincent Deng	vdeng@usc.edu	Junyang Cai	caijunya@usc.e	hanjin@usc.edu		Han Xiao	hxiao603@us
15	Synthetic Satellite Image Generation for Forecasting Urban and Natural Changes	Jay Prajapati	jprajapa@usc.edu	Advait Thergao	thergaon@usc.e	Cynthia Chang	cchang80@usc	Chanavi Sing	chanavis@	Shourya Kothari	sakothar@us
16	Using Deep Learning for Image Compression	Mahika Mushuni	mushuni@usc.edu	Diana Pham	dianadph@usc.e	Karen Wang	kwang760@usc	Praise Olukil	olukiled@u	Vedant Modi	vkmodi@usc
17	Analyzing Community Opinion Dynamics in Response to Significant Sociopolitical Events with Large Language Models	Kshitij Pawar	kshitijv@usc.edu	Syed's Masoom	syedamas@usc.	Sanjay Raghavend	sraghave@usc.	Mihika Gaon	mihikapr@usc.edu		
18	Investigating the Impact of Differing Word Embeddings on Gender Bias in Language Models (LLMs)	Omi Wakode	wakode@usc.edu	Padmaja Borwa	borwanka@usc.	Yatharth Arora	yarora@usc.ed	Andrew Klotz	aklotz@usc	Tanisha Rathi	trathi@usc.e
19	Self Contradictory reasoning and logical fallacies detection	Ritesh Reddy Banda	rbandam@usc.edu	Paritosh Kadam	pskadam@usc.e	Sandeep Menon	menons@usc.e	Aviral Goel	aviralgo@u	Krishna Wankhe	wankhede@u
20	Benchmarking unsupervised environment design (UED) algorithms	Nikhil Sathyanarayar	nsathyan@usc.edu	Prajna Narayan	hollakun@usc.e	Ronak Jain	ronakjai@usc.e	Akshay Kaushik		Rithvik Mohan Vi	vijalapu@usc
21	Untitled	Rishabh Shinde	shinder@usc.edu	Pallavi Udatewa	udatewar@usc.e	Aryan Gandhi	aryangan@usc.	Aditya Jadha	adityadh@usc.edu		

Logistics

https://docs.google.com/spreadsheets/d/1IEisSea4CF86ikrhrUKtiTDqQQ-SkctIWU5XTHu8cr8/edit#gid=0

1					zizhao Hu
2	**Project Name/Title (Temporary)**	**Member 6**	**Email**	**Comment (e.g., still looking for people)**	**Assigned TA (TBA)**
3	DeepFake Video Detection	Tom Yang	cyang513@usc.edu	Looking for a TA who can support us on video and audio processing	Pengda Xiang
4	Offline RL with learned action discretization with VQ-VAE				Ayush Jain
5	NLP for Summarization and Research Paper Trend Analysis				Zihao He
6	Drug Molecular Effectiveness Prediction with Enhanced GNNs				Yuehan Qin
7	Reinforcement Learning for Recommender Systems: Environment Suite	Harshita Poojary	hpoojary@usc.edu		Ayush Jain
8	NLP Anomaly Detection Library	Zhaotian Weng	wengzhao@usc.edu		Yue Zhao
9	Medical Advice Chatbot				Yuehan Qin
10	CAT-RAG: Conversational ASL Translator using LLMS with RAG				Ayush Jain
11	Automated PII Detection and Removal				Ziyi Liu

Project Check-in (comments released this week)

Each project has 2 mandatory check-in with your TA

- Potentially one before the mid-report due (March 22nd)
- Potentially another one before final report is due (May 3rd)
- We will work out the logistic then

We have 55 projects – each TA is in charge of 7-8 projects

- They will be your point of contact for project questions
- Use **Piazza**, **office hours**, or **scheduled appointment via email** with them to make sure your questions are addressed

Assignments

Due in 2 weeks…

- The project should still be ongoing
- Questions on piazza will be addressed by TA
    - Yuehan, Varun, and Ziyi will help
- Also TA hours will be helpful for all the Tas

Do not do it in the last minute – recall we do not do late days due to the size of the course

Interaction on Piazza

Please post general questions to all students:

- You will get your answer much faster!
- Again – do not post your assignment answer, but the general questions are fine!

If you actively answer others' questions, we will take note of your name – if you need a bump by the end of the semester, we will get you there automatically due to your contribution to the course!

TA Office Hours

We have a slightly easier way to handle office hours (not that ideal)

Zihao He 4 days ago

Actions

Hi everyone, we will share our TA OHs in here in the future.

	Varun Bhatt	Zihao He	Zizhao Hu	Ayush Jain	Ziyi Liu	Yuehan Qin	Pengda Xiang
Week of Feb 5	Tuesday 14:00 - 15:00, RTH 419 (preferred) or Zoom,	Thu, 2-3pm Leavey 202C, Zoom Link	Wed, 10-11am Leavey, 201F Zoom	Wed, 2-3pm RTH 4th floor lobby		Tuesday 10:00-11:00 Zoom	
Week of Feb 12							

good comment | 2

Midterm – Note there is no Final!

Feb 23rd – **in-class** – **paper-based** – open books (although I have no idea which books to use)

- 50 + 6 (bonus) multiple choice questions + single answer questions, e.g., a single number
- there is no make-up unless you have medical emergency with proof ☺
- 1 bonus for each quiz and three bonus for the exam (0.5 each so it is 6 questions)

Midterm – Note there is no Final!

- Overfitting, regularization (L1, L2, dropout, pruning, etc)
- Details of CNN
- Basic calculation of NN and CNN parameters
- The functionalities of each step of GNN
- Concepts of decision tree, kNN, anomaly detection

Midterm – Note there is no Final!

- Print your name
- Student ID
- Put everything on the first page
- We will only grade the first page

Student Name (Please print):

Student ID:

Student Email:

This exam contains 50+6 bonus questions with equal weights (each worth ½ point of your total grade). Please only provide your answer in the answer sheet below. We will only grade the **first page** and not consider the information on other pages. Please make sure your answer is readable.

This exam is open book, open notes, but no computers or other electronic devices.

The total time for this exam is 112 minutes (2 min/question; 1:00 pm to 2:52 pm). Students with OSAS permission can take 1.75 times the duration and submit by 4:20 pm.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
17	18	19	20
21	22	23	24
25	26	27	28
29	30	31	32
33	34	35	36
37	38	39	40
41	42	43	44
45	46	47	48
49	50	51	52
53	54	55	56

What is Happening? Italian Pup Open AI

What is Happening? Gold Rush Open AI

Who Takes my Job?

Recurrent Neural Networks (RNN)

Motivation

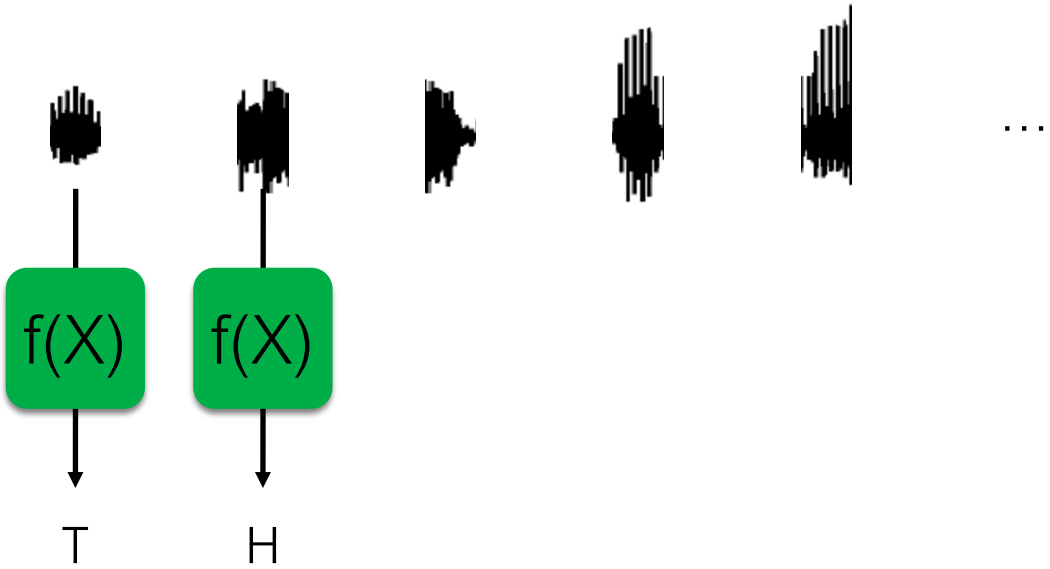
What about speech recognition?

Slide credit: Dhruv Batra
Image credit: Alex Graves and Kevin Gimpel

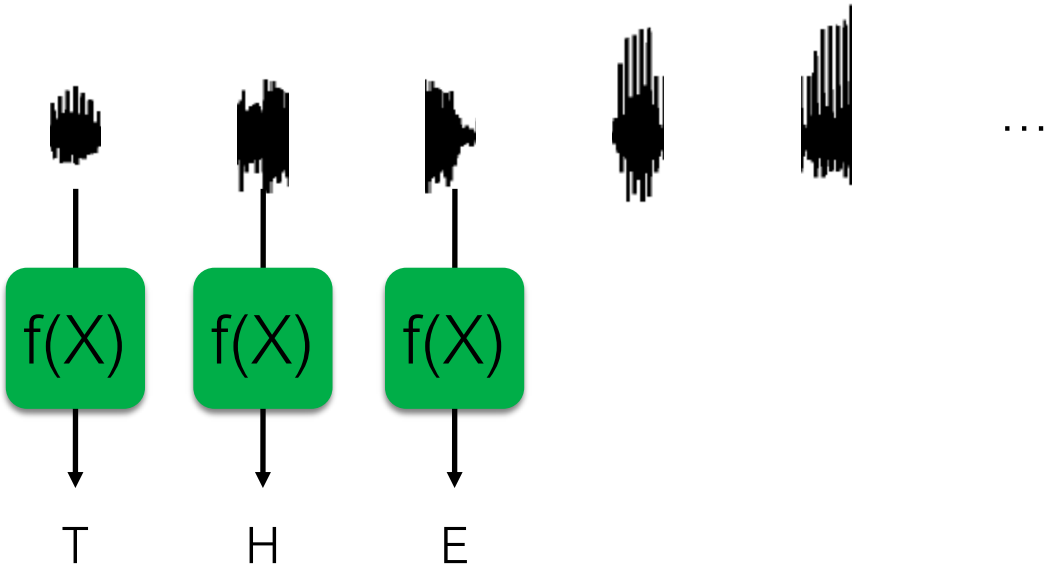
What about speech recognition?

What about speech recognition?

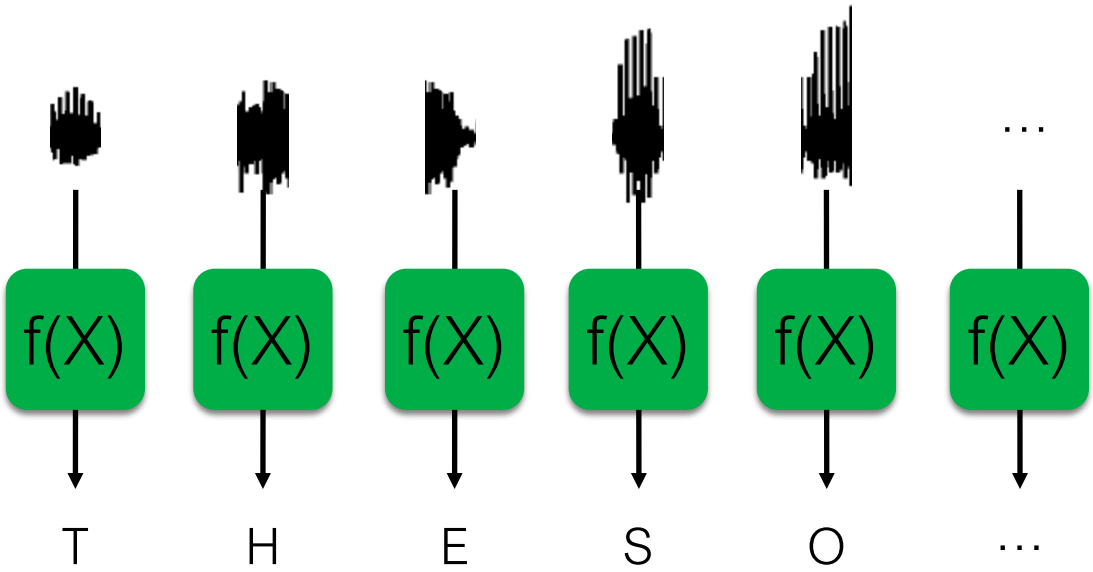
What about speech recognition?



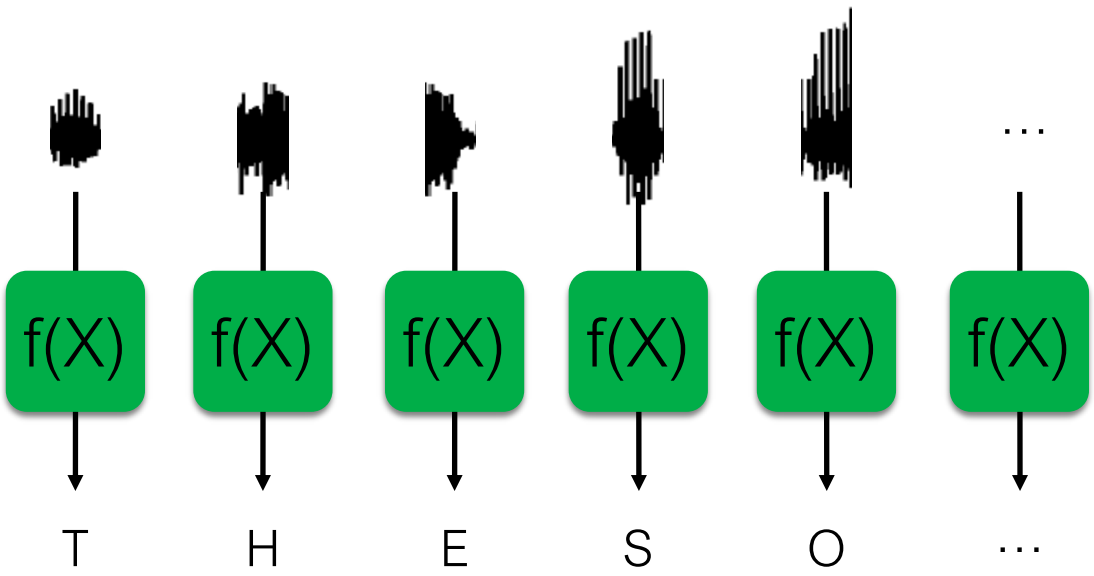
What about speech recognition?



What about speech recognition?

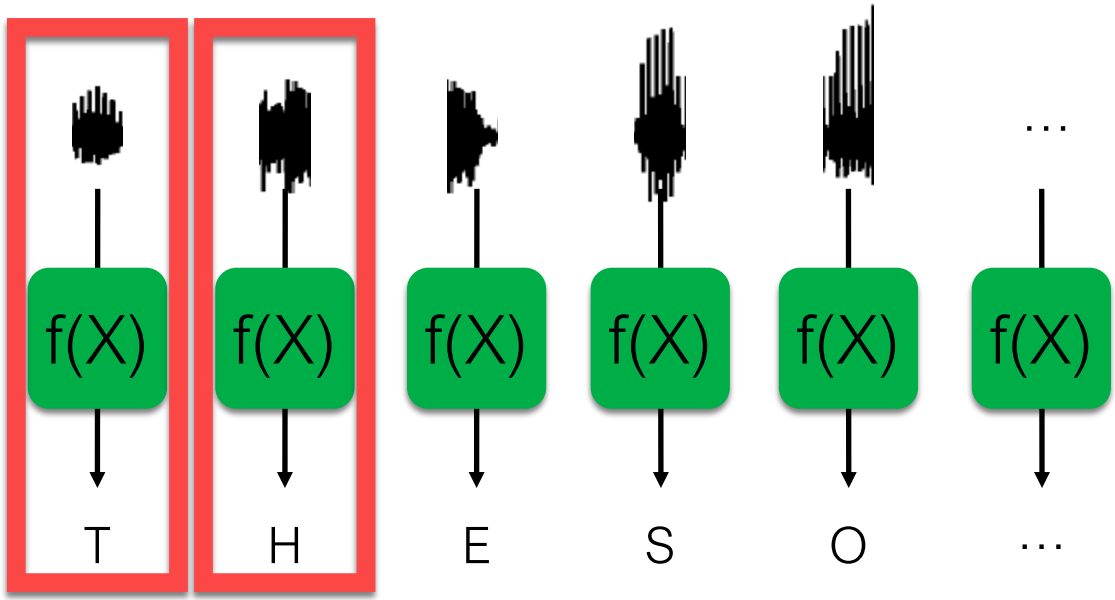


What about speech recognition?

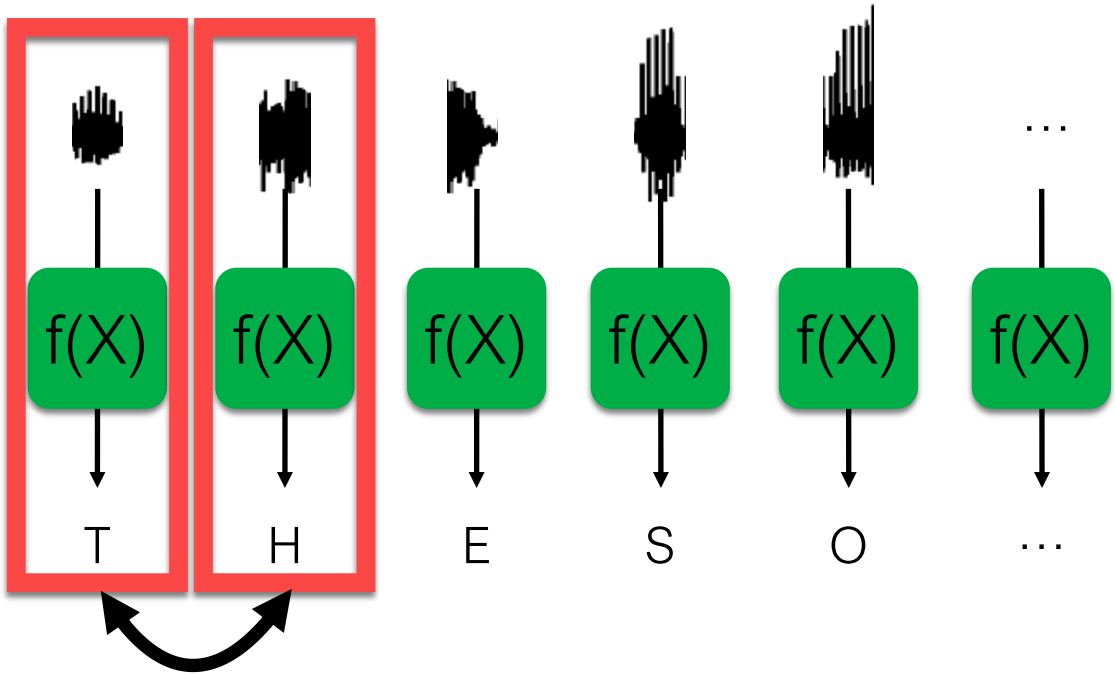


Any limitation on this approach?

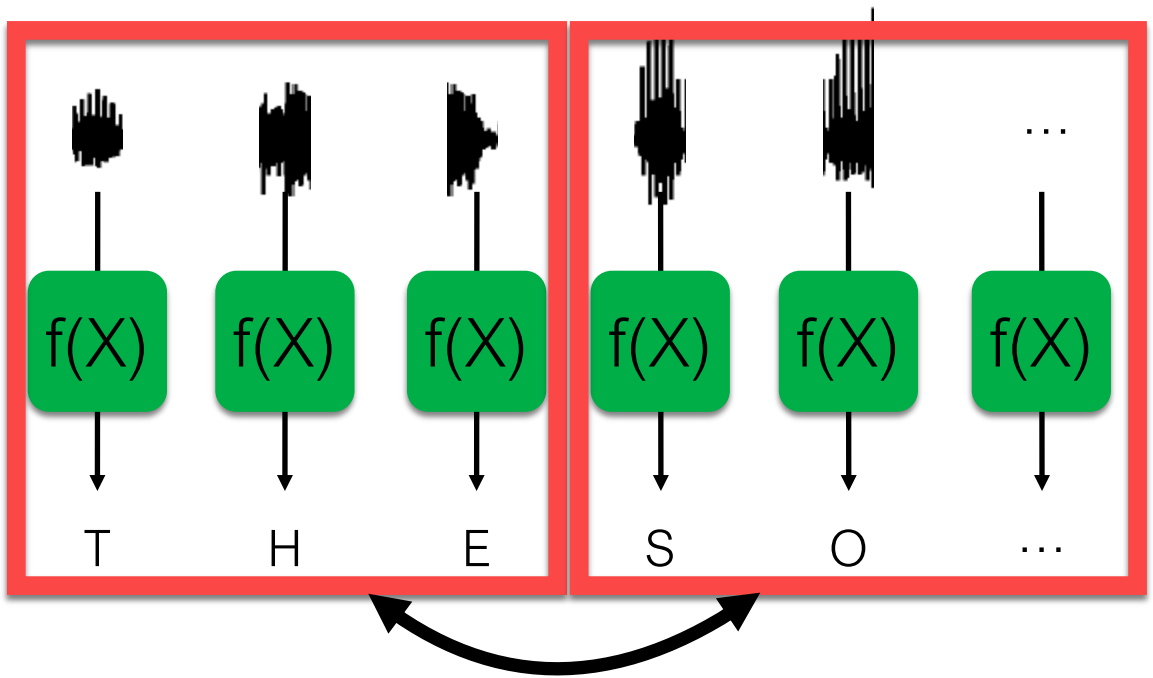
A limitation of “normal” CNNs



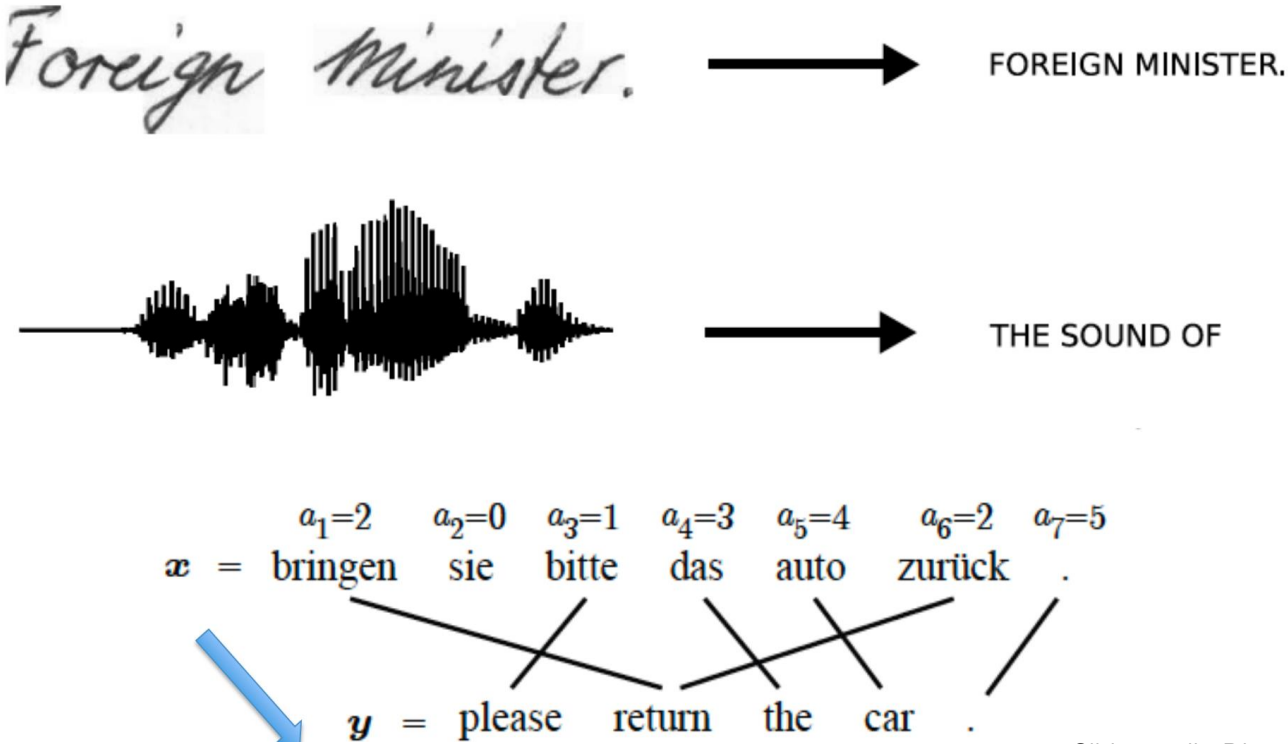
A limitation of “normal” CNNs



A limitation of “normal” CNNs

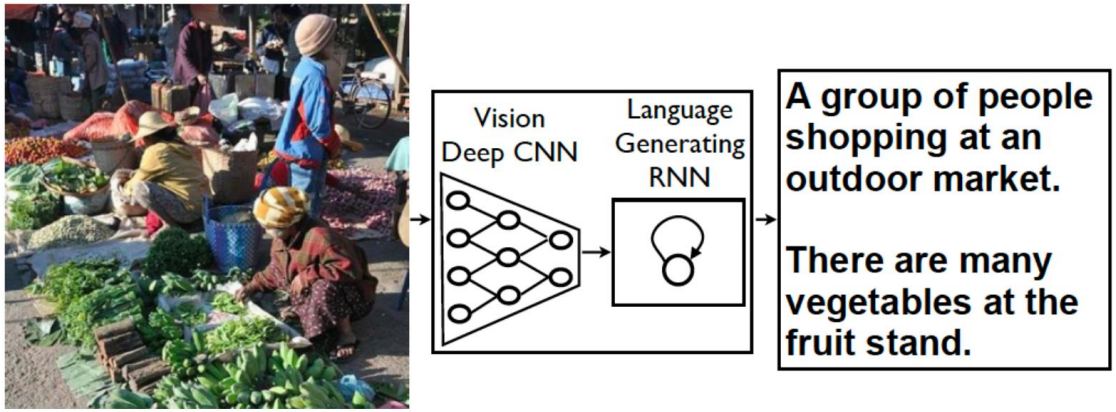


Sequential data is everywhere



Slide credit: Dhruv Batra
Image credit: Alex Graves and Kevin Gimpel

Sequential data is everywhere



John has a dog . → (S (NP NNP)$_{NP}$ (VP VBZ (NP DT NN)$_{NP}$)$_{VP}$.)$_{S}$

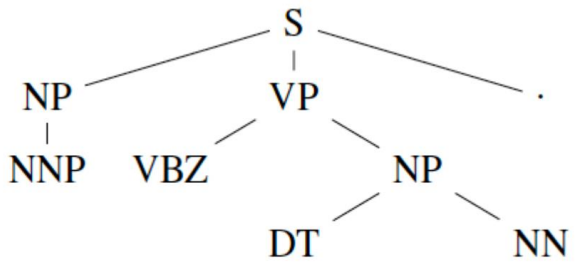
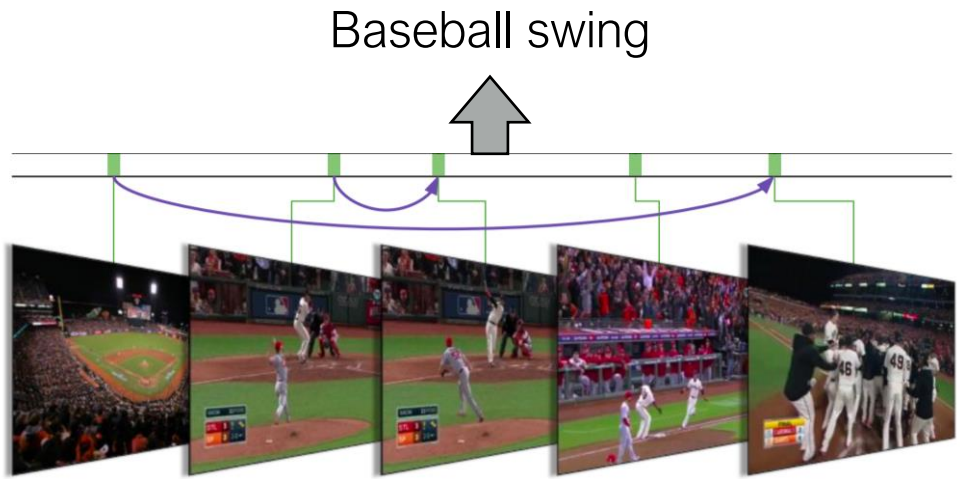


Image credit: Vinyals et. al.

Sequential data is everywhere

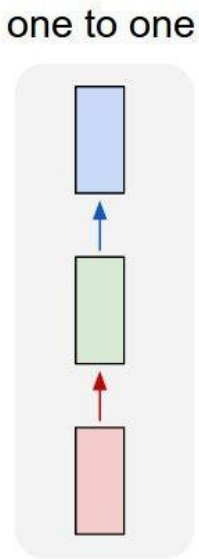


Yeung, et. al. End-to-end Learning of Action Detection from Frame Glimpses in Videos. CVPR 2017.

Recurrent Neural Networks (RNN)

Basics

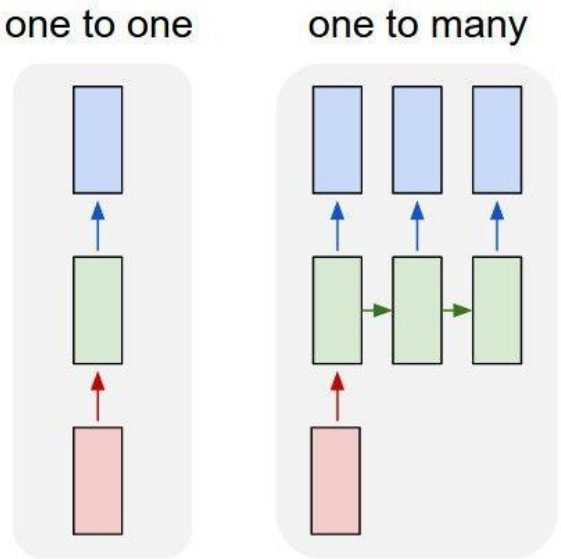
“Vanilla” Neural Network



Vanilla Neural Networks

Credit to Stanford cs231n lecture slides

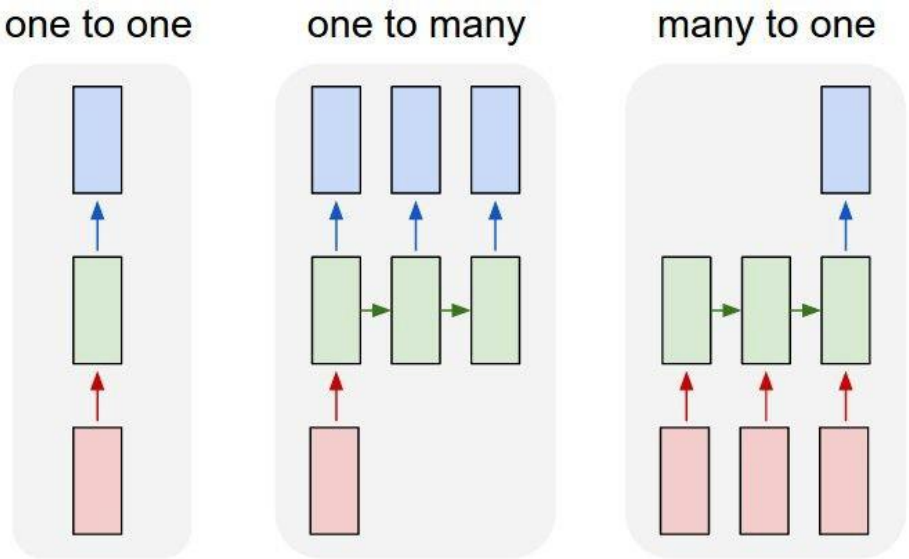
Recurrent Neural Networks: Process Sequences



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

Credit to Stanford cs231n lecture slides

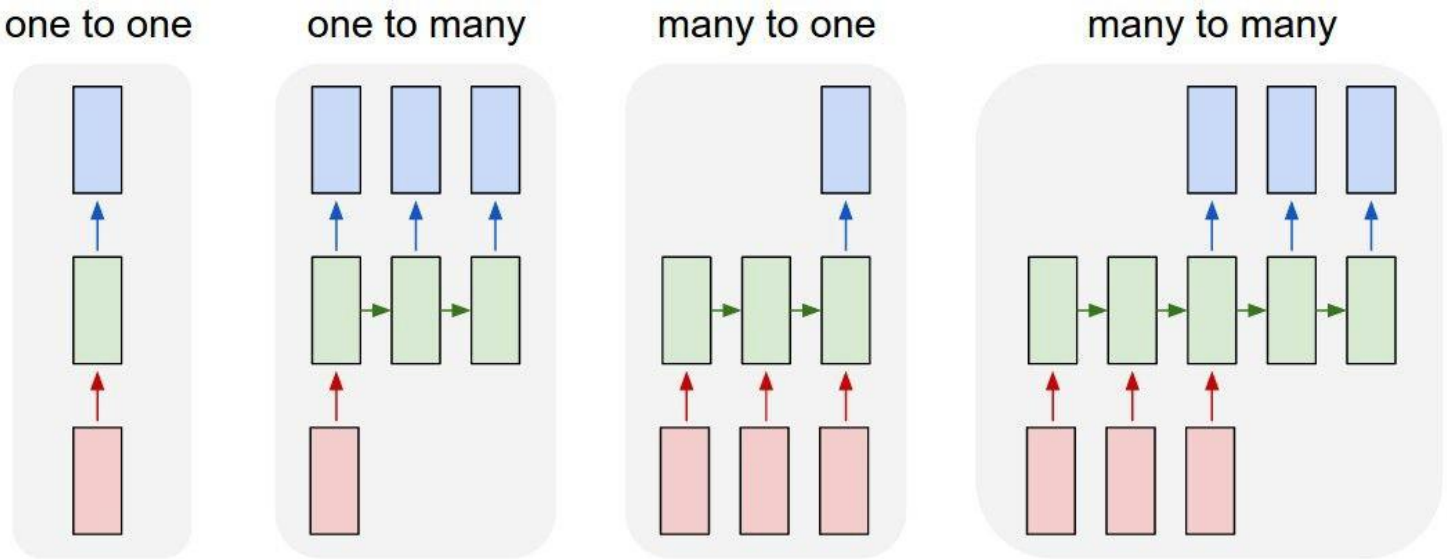
Recurrent Neural Networks: Process Sequences



e.g. **action prediction**
sequence of video frames -> action class

Credit to Stanford cs231n lecture slides

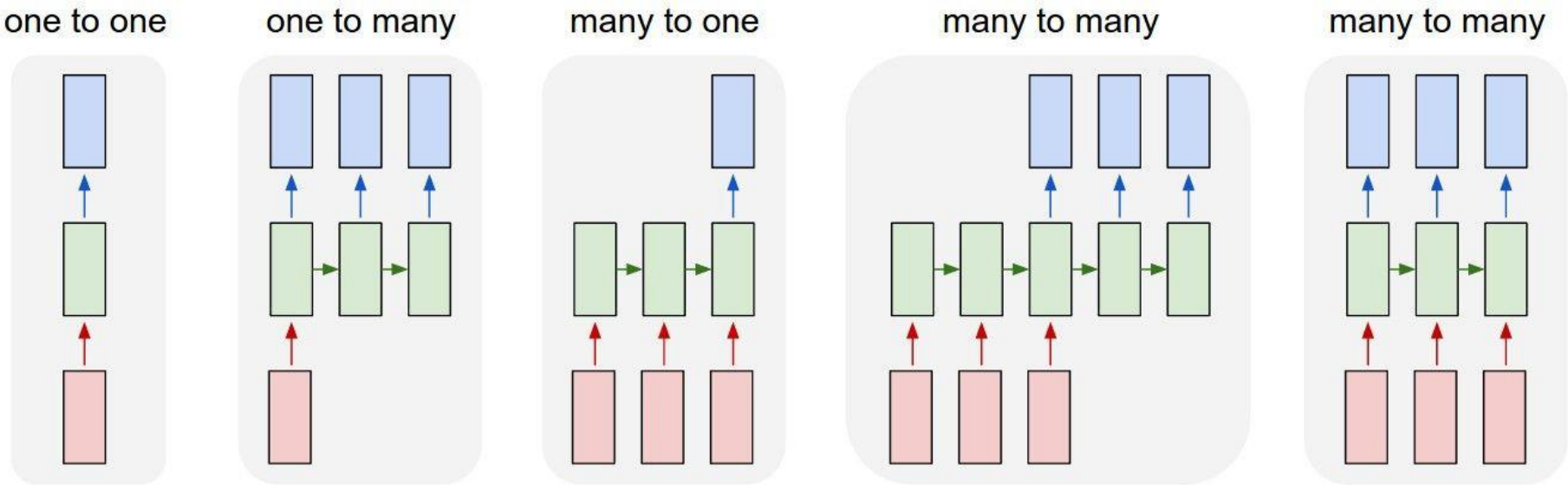
Recurrent Neural Networks: Process Sequences



E.g. **Video Captioning**
Sequence of video frames -> caption

Credit to Stanford cs231n lecture slides

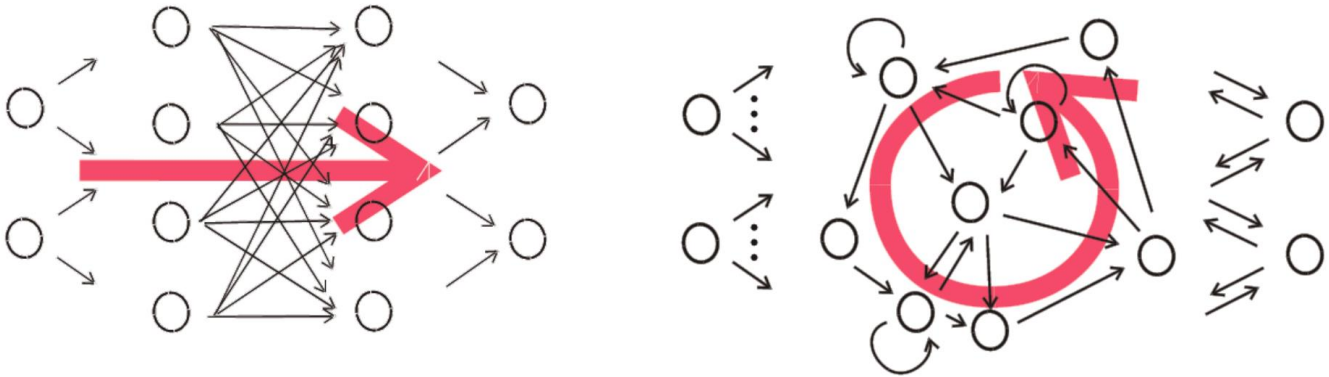
Recurrent Neural Networks: Process Sequences



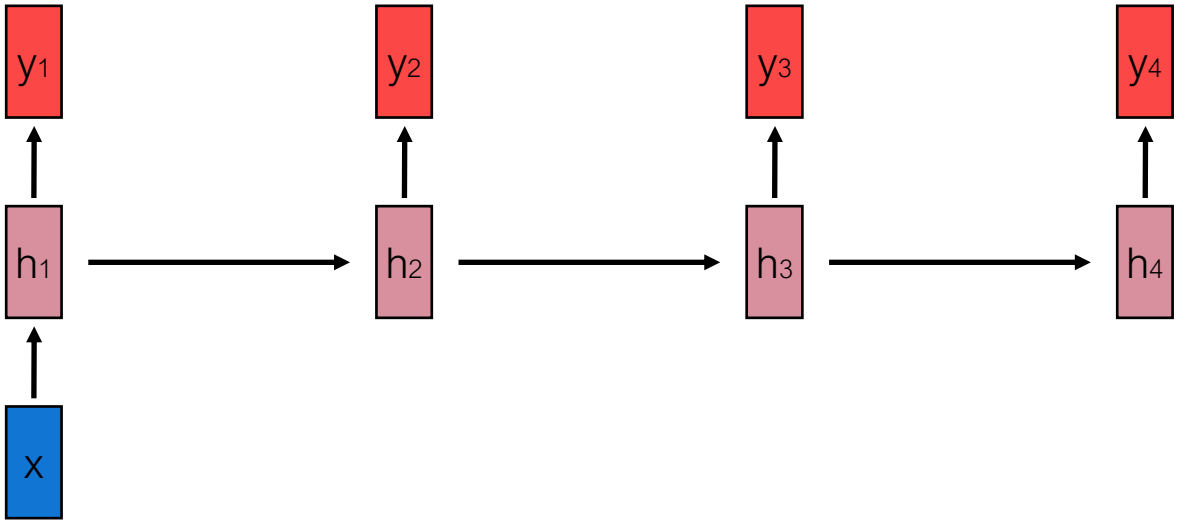
e.g. **Video classification on frame level**

Credit to Stanford cs231n lecture slides

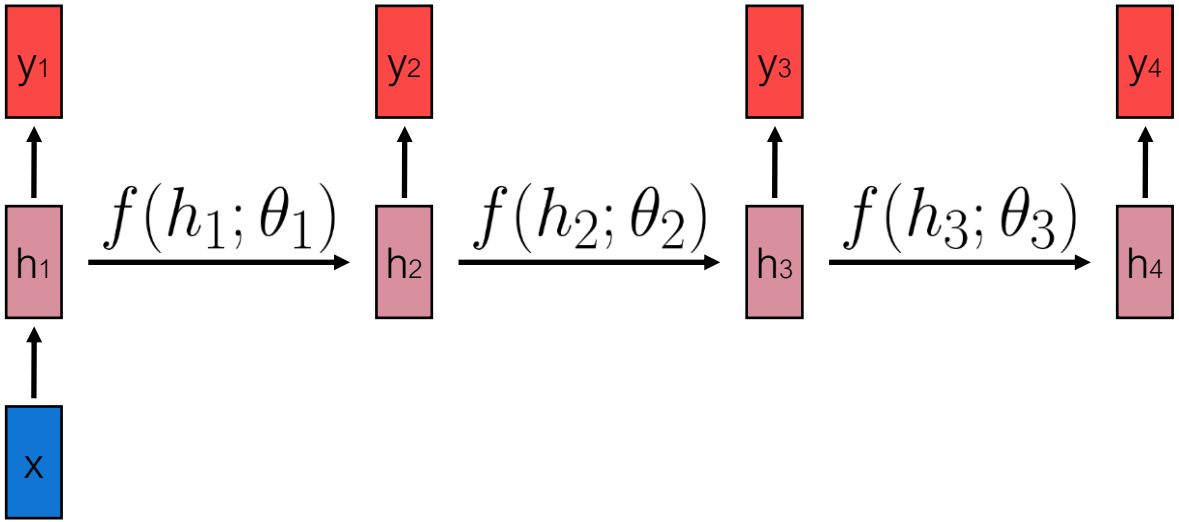
It can get tricky…



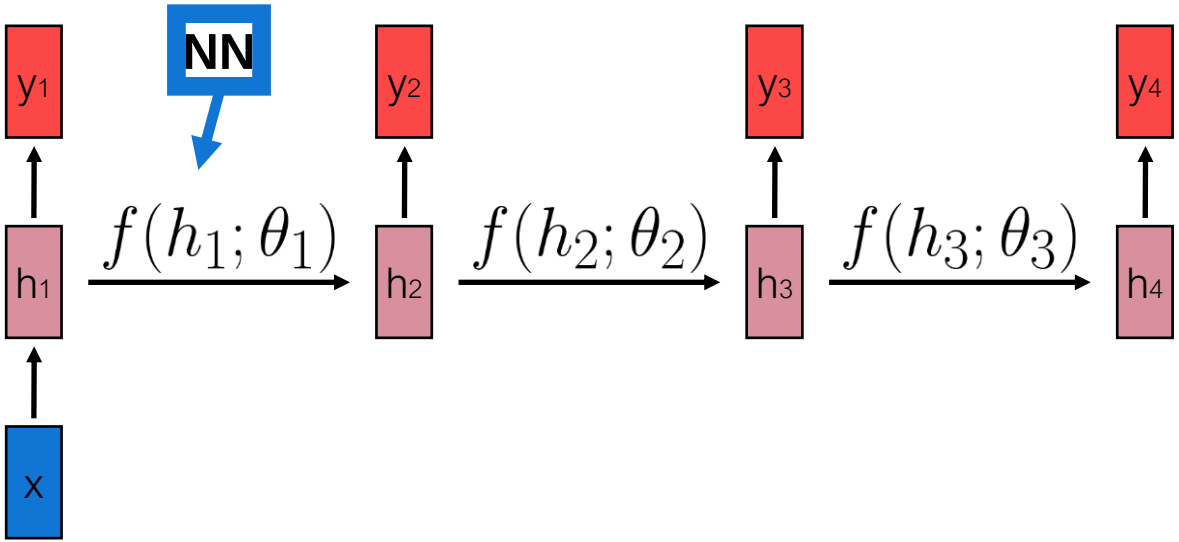
Let's zoom in!



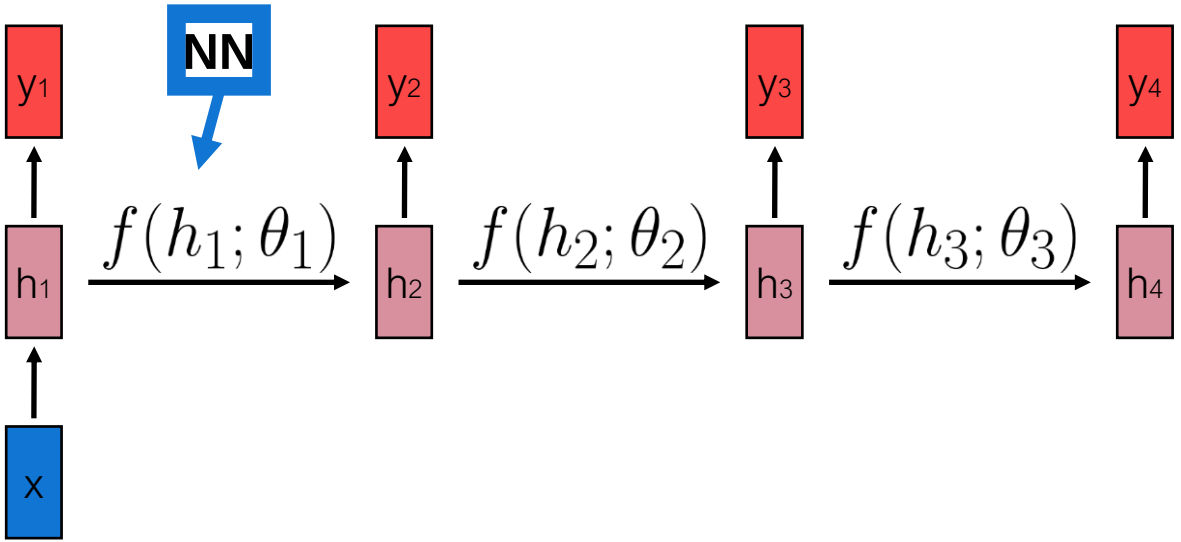
Let's zoom in!



Let's zoom in!



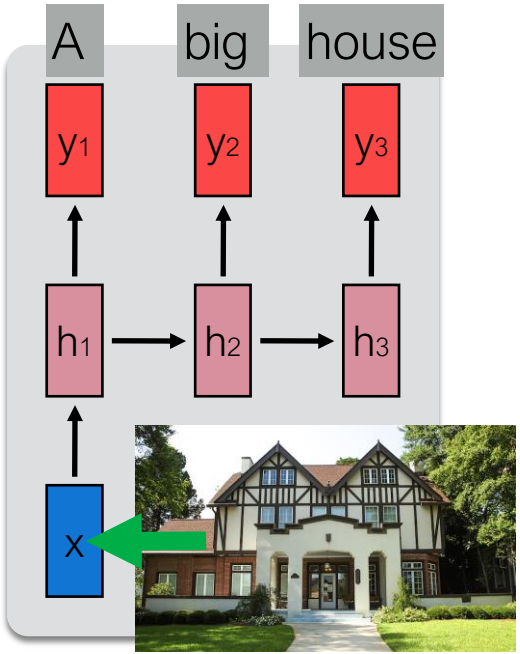
Let's zoom in!



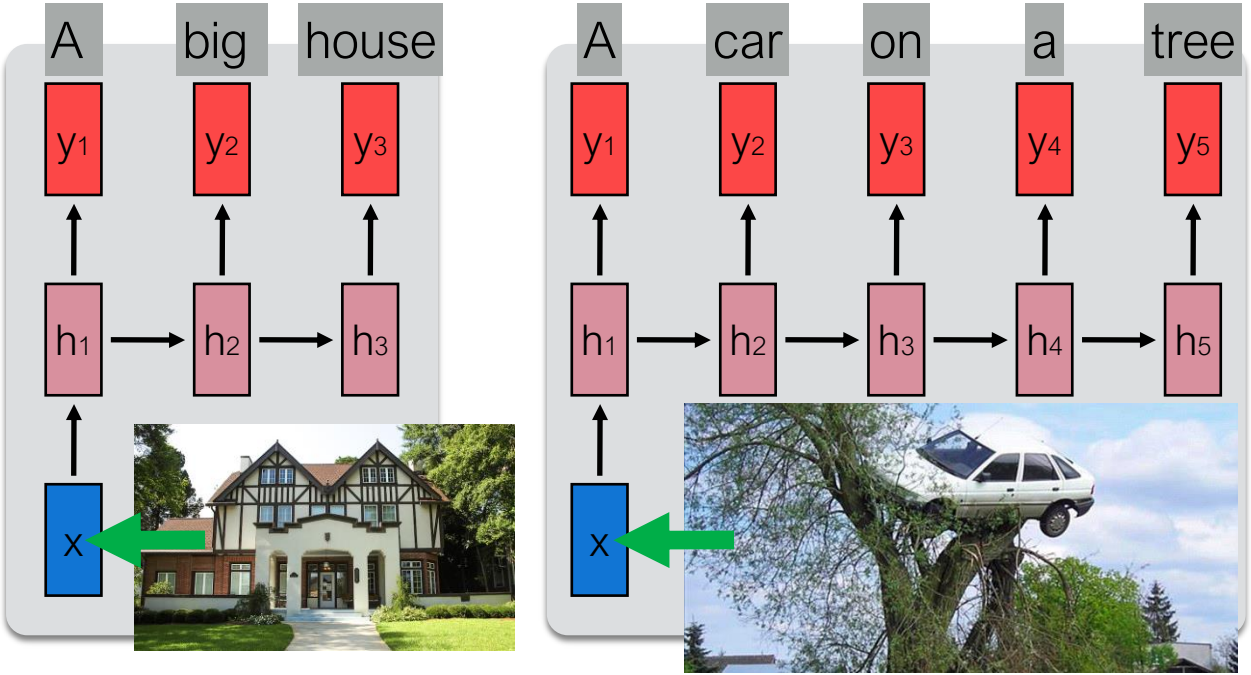
Any issue with this?

How do we model sequences?

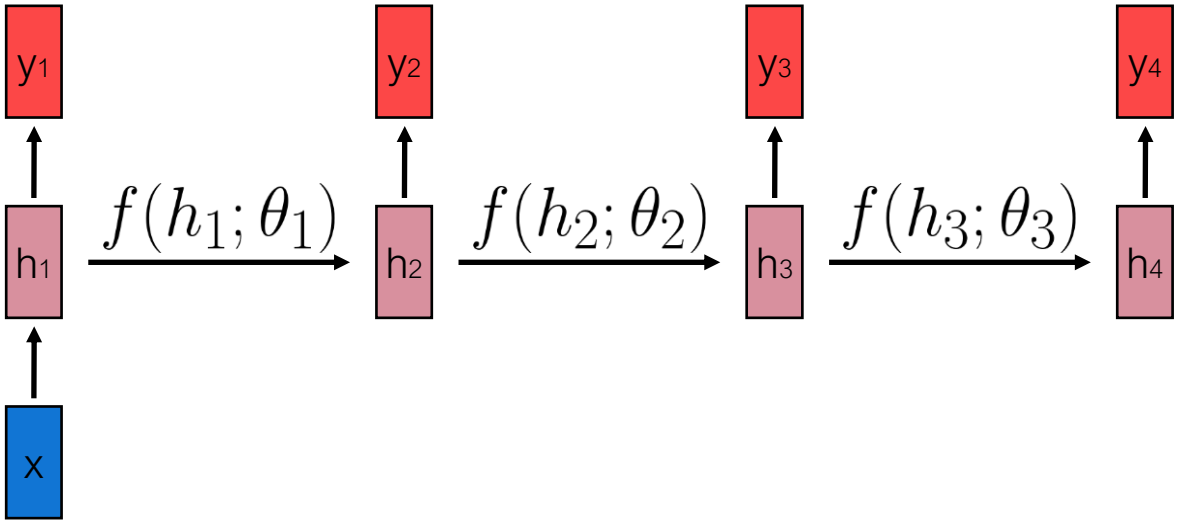
How do we model sequences?



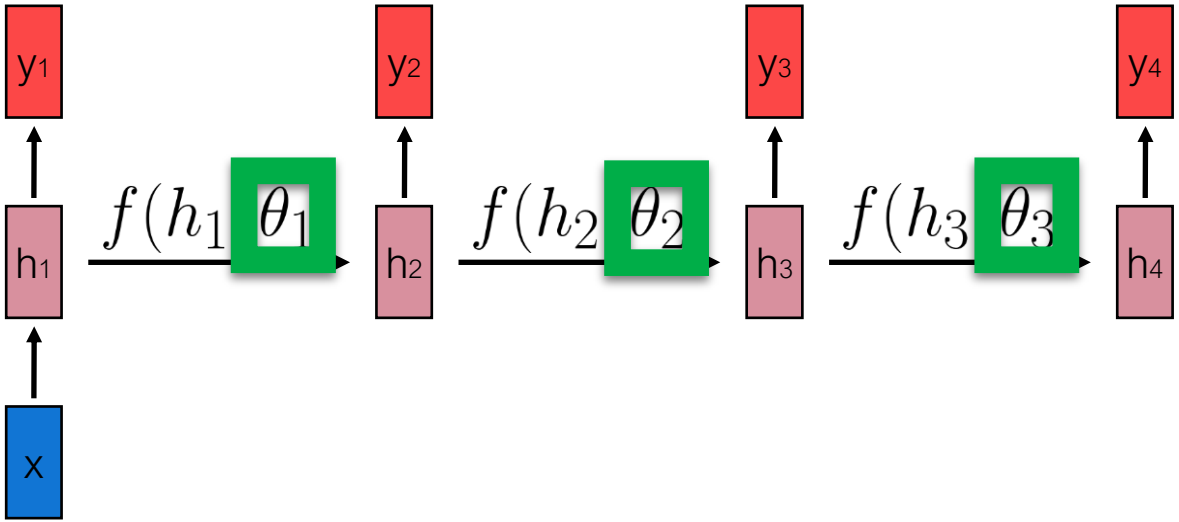
How do we model sequences?



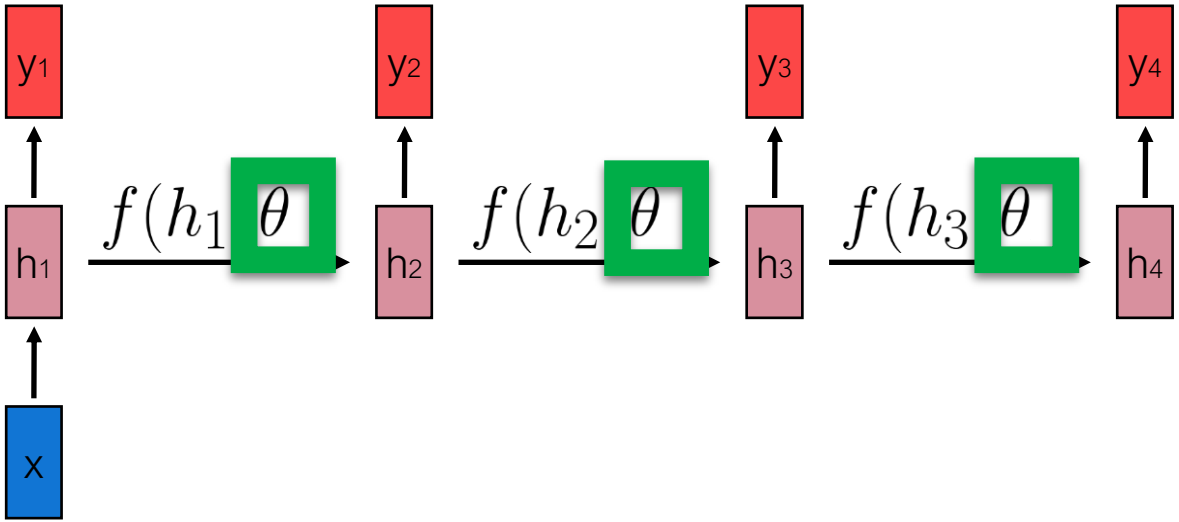
Recurrent Neural Network



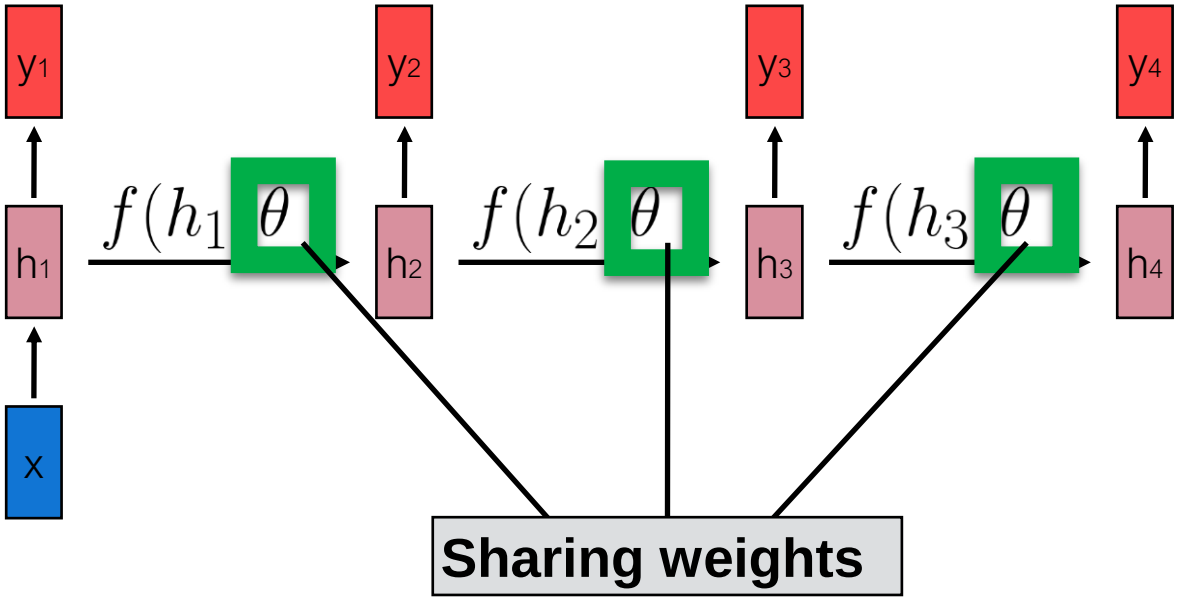
Recurrent Neural Network



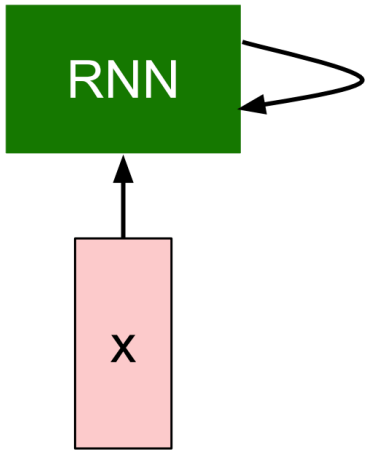
Recurrent Neural Network



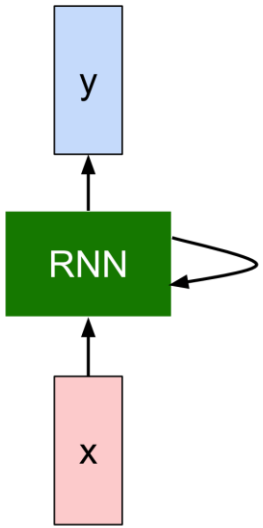
Recurrent Neural Network



Recurrent Neural Network



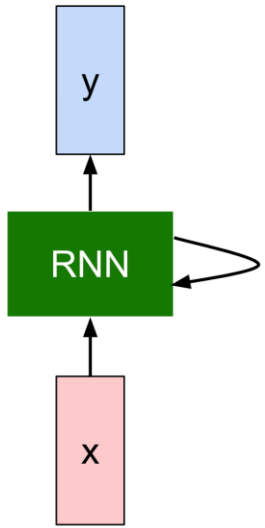
Recurrent Neural Network



Slide credit: Stanford CS231n

Recurrent Neural Network

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

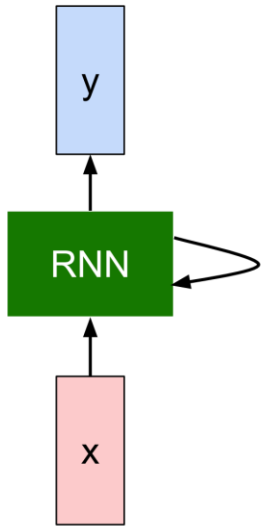


Slide credit: Stanford CS231n

Recurrent Neural Network

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

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



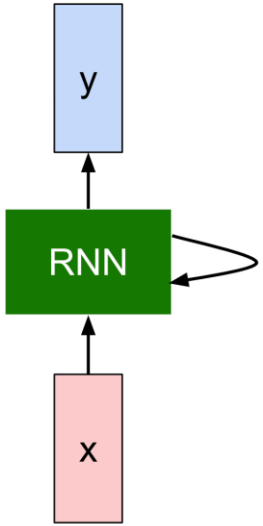
Slide credit: Stanford CS231n

Recurrent Neural Network

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

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

old state



Slide credit: Stanford CS231n

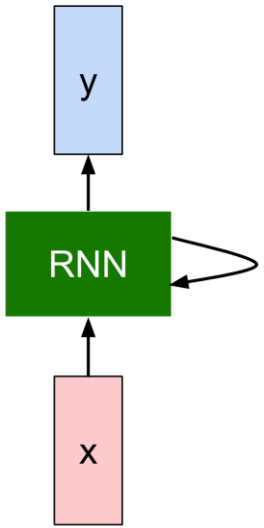
Recurrent Neural Network

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

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

old state

input vector at some time step



Slide credit: Stanford CS231n

Recurrent Neural Network

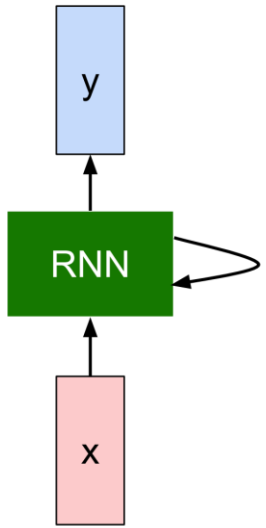
We can process a sequence of vectors **x** by applying a **recurrence formula** at every time step:

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

old state

input vector at some time step

some function with parameters W



Slide credit: Stanford CS231n

Recurrent Neural Network

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

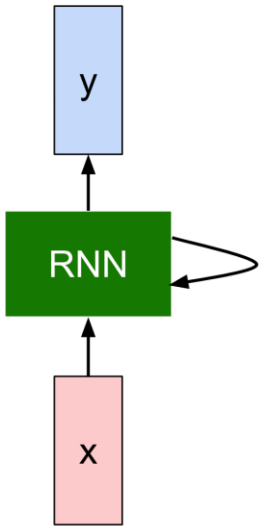
$$h_t = f_W(h_{t-1}, x_t)$$

new state

some function with parameters W

old state

input vector at some time step



Recurrent Neural Network

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

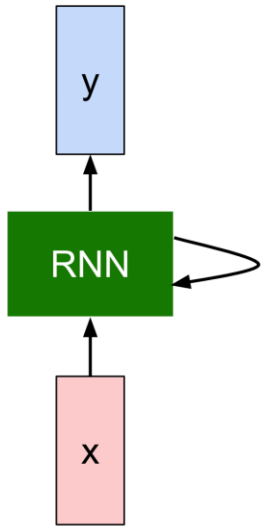
$$h_t = f_W(h_{t-1}, x_t)$$

new state

old state

input vector at some time step

some function with parameters W

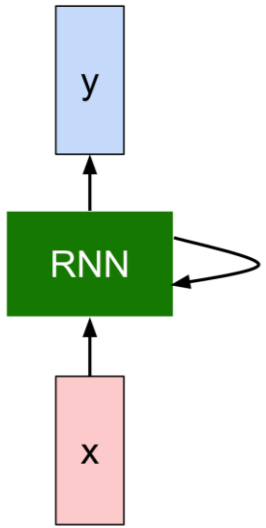


Note: the same function with the same parameters

Slide credit: Stanford CS231n

Recurrent Neural Network

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



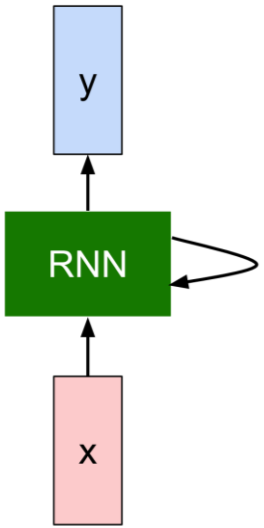
Slide credit: Stanford CS231n

(Vanilla) Recurrent Neural Network

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

$$\downarrow$$

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

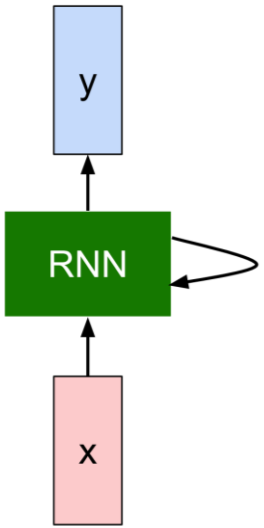


(Vanilla) Recurrent Neural Network

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

$$\downarrow$$

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



The state has a single “hidden” vector **h**

Slide credit: Stanford CS231n

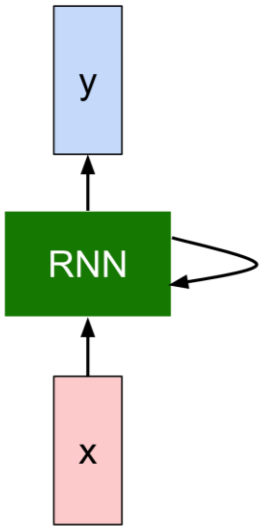
(Vanilla) Recurrent Neural Network

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

$$\downarrow$$

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

$$y_t = W_{hy}h_t$$



The state has a single "hidden" vector **h**

Slide credit: Stanford CS231n

The Math: Deep Encoder

- **Basic approach:** Average neighbor messages and apply a neural network

Initial 0-th layer embeddings are equal to node features

$$\mathbf{h}_v^0 = \mathbf{x}_v$$

embedding of v at layer k

$$\mathbf{h}_v^{(k+1)} = \sigma\left(\mathbf{W}_k \sum_{u \in \mathrm{N}(v)} \frac{\mathbf{h}_u^{(k)}}{|\mathrm{N}(v)|} + \mathbf{B}_k \mathbf{h}_v^{(k)}\right), \forall k \in \{0, \ldots, K-1\}$$

Total number of layers

Average of neighbor's previous layer embeddings

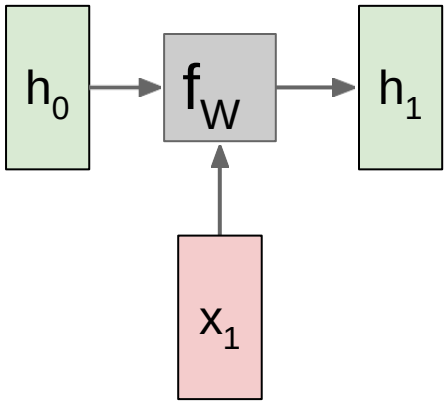
$$\mathbf{z}_v = \mathbf{h}_v^{(K)}$$

Embedding after K layers of neighborhood aggregation

Non-linearity (e.g., ReLU)

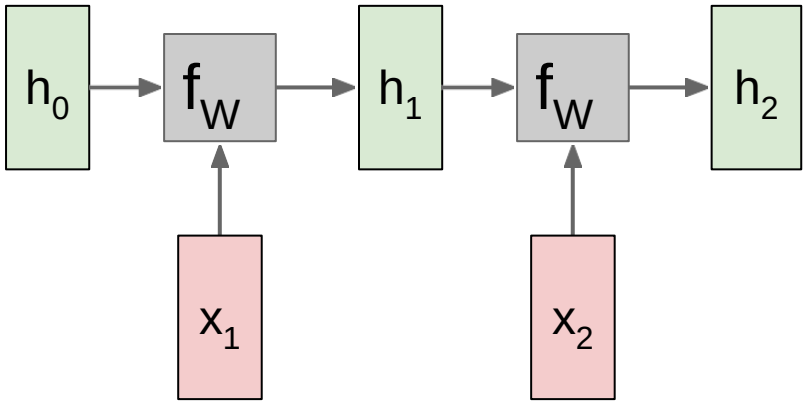
Notice summation is a permutation invariant pooling/aggregation.

RNN: Computational Graph



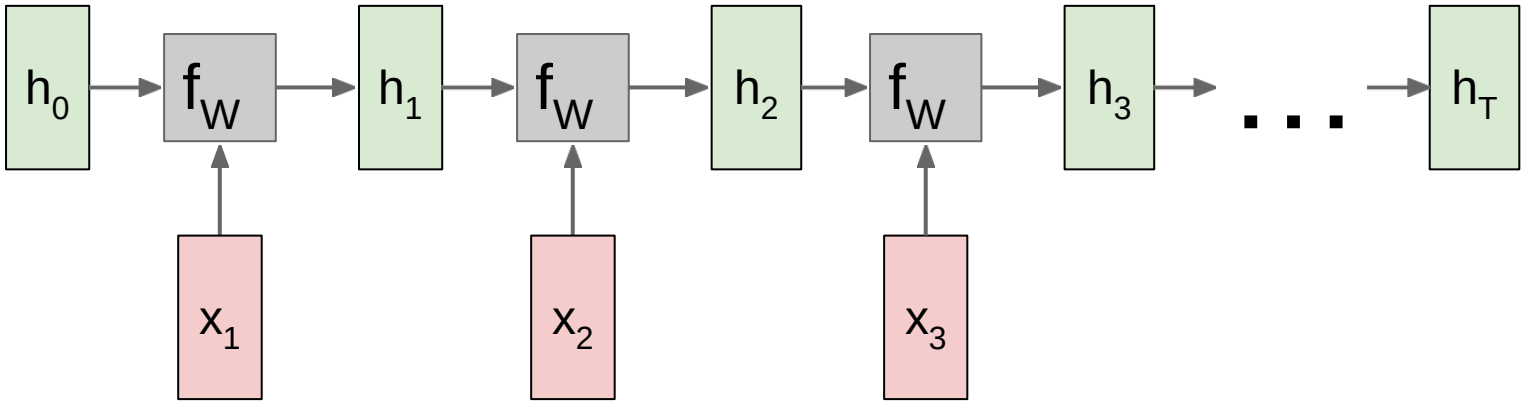
Credit to Stanford cs231n lecture slides

RNN: Computational Graph



Credit to Stanford cs231n lecture slides

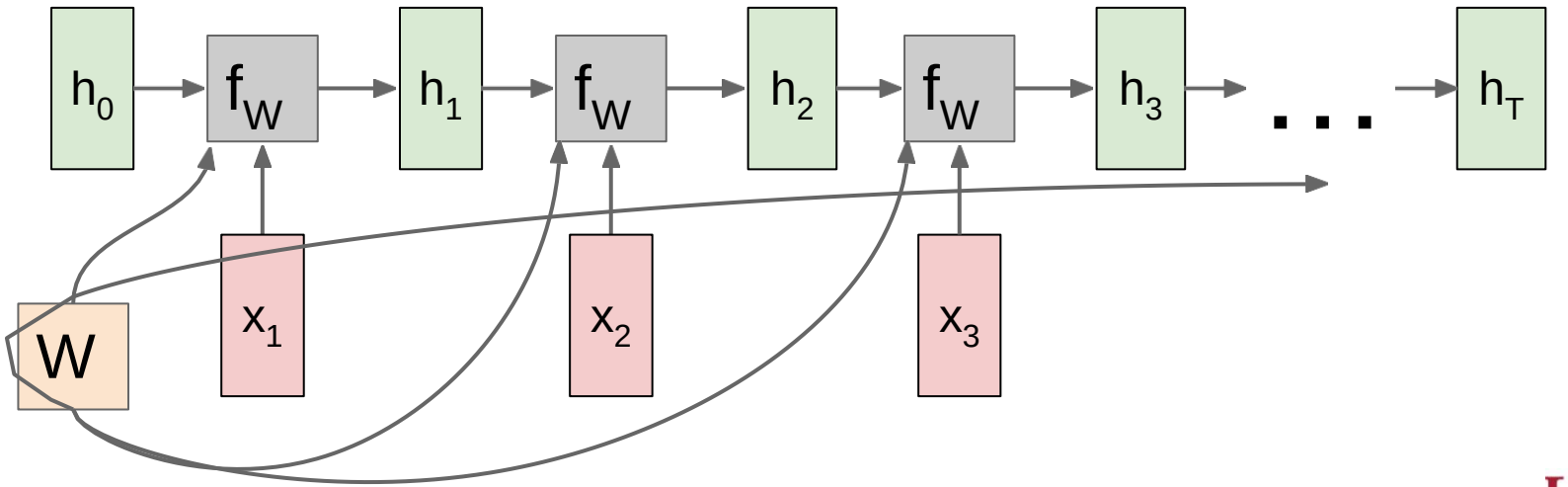
RNN: Computational Graph



Credit to Stanford cs231n lecture slides

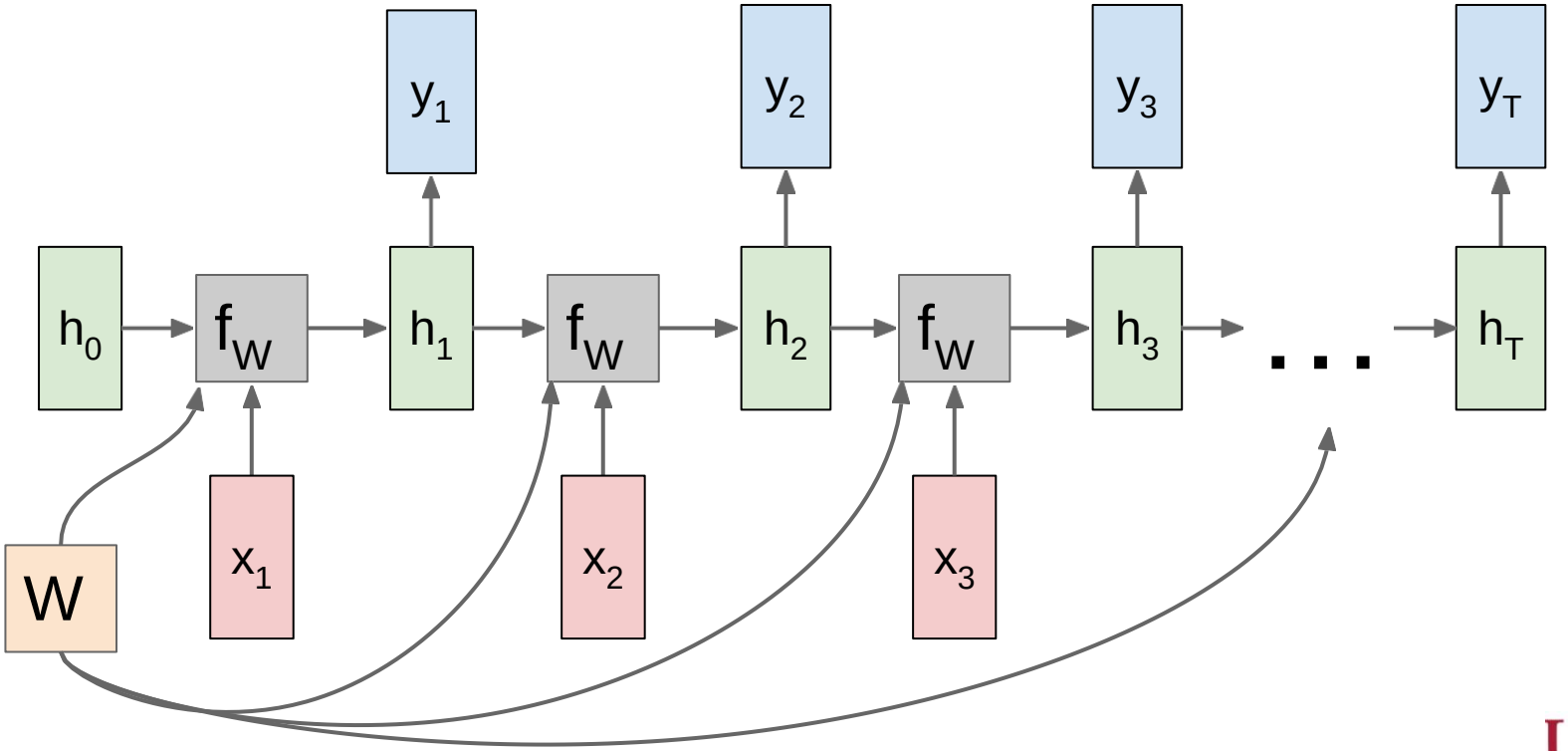
RNN: Computational Graph

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



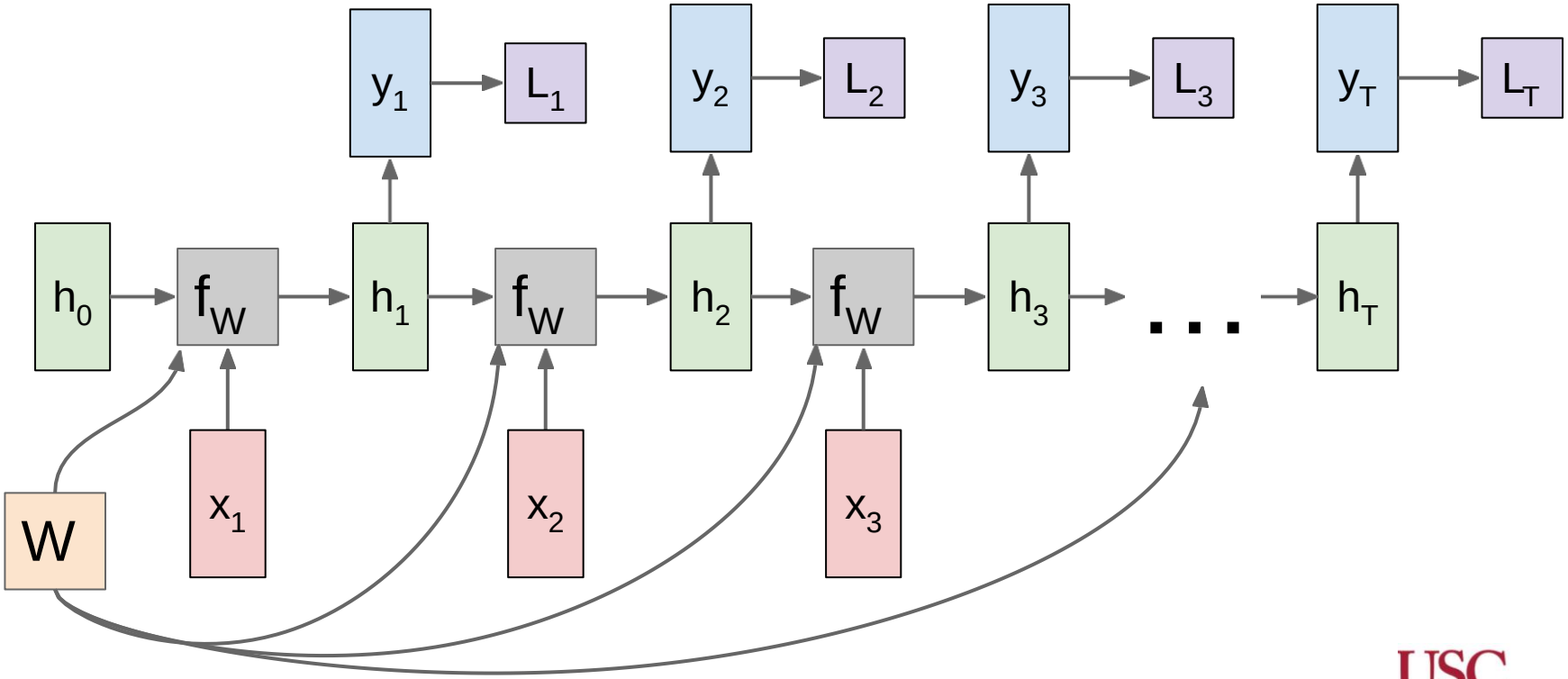
Credit to Stanford cs231n lecture slides

RNN: Computational Graph: Many to Many



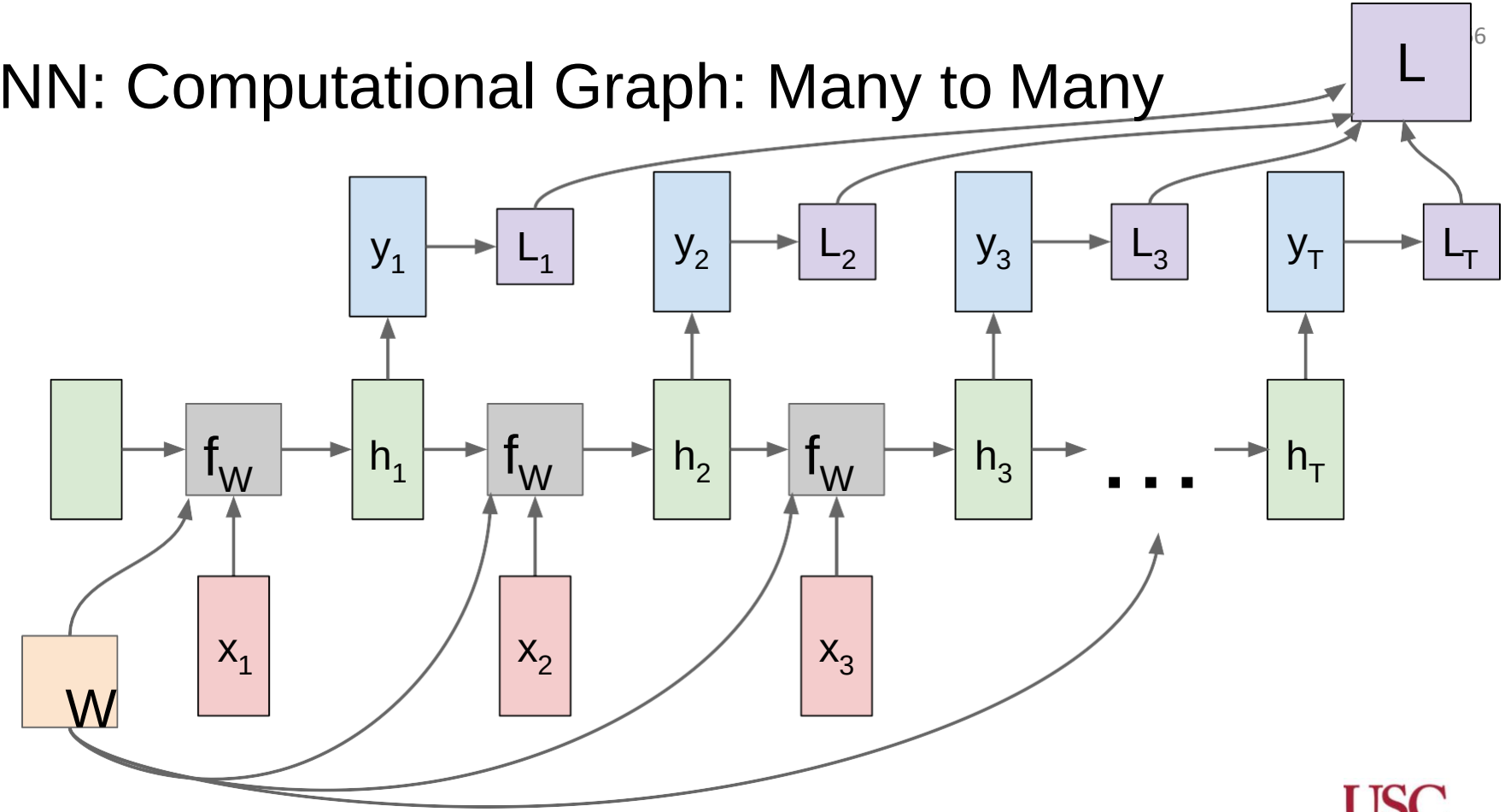
Credit to Stanford cs231n lecture slides

RNN: Computational Graph: Many to Many

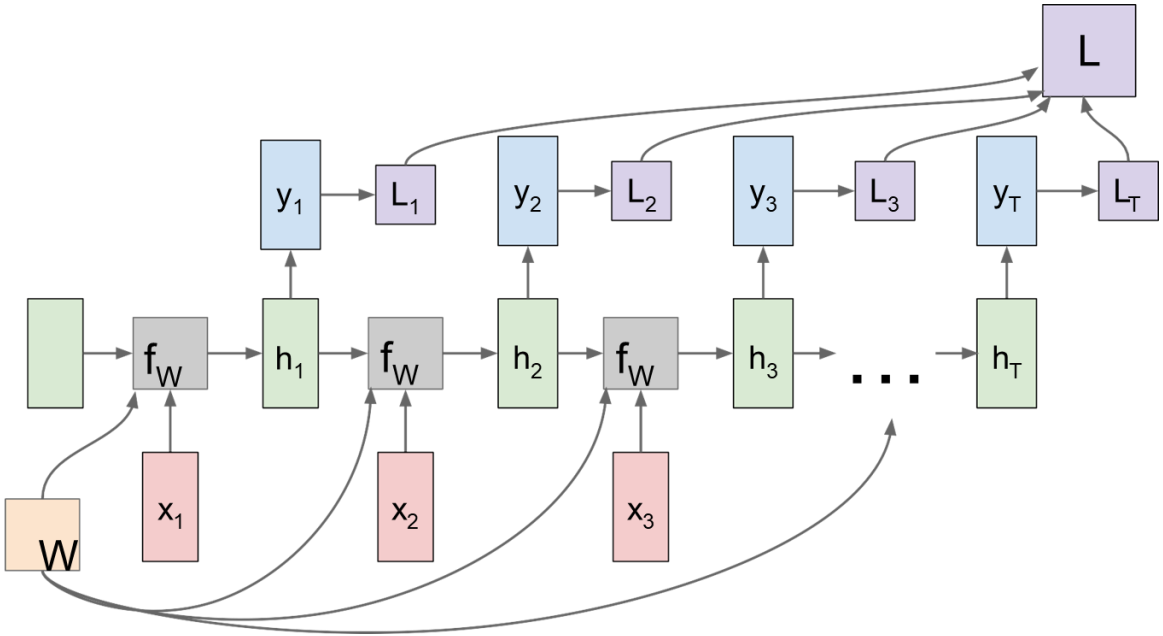


Credit to Stanford cs231n lecture slides

RNN: Computational Graph: Many to Many



Credit to Stanford cs231n lecture slides



f_W: This is the function that calculates the new hidden state

$y_1, y_2, y_3, \ldots y_t$: These are the outputs at each time step,

$L_1, L_2, L_3, \ldots L_t$: These are the loss values at each time step, calculated by comparing the output at each time step with the true value

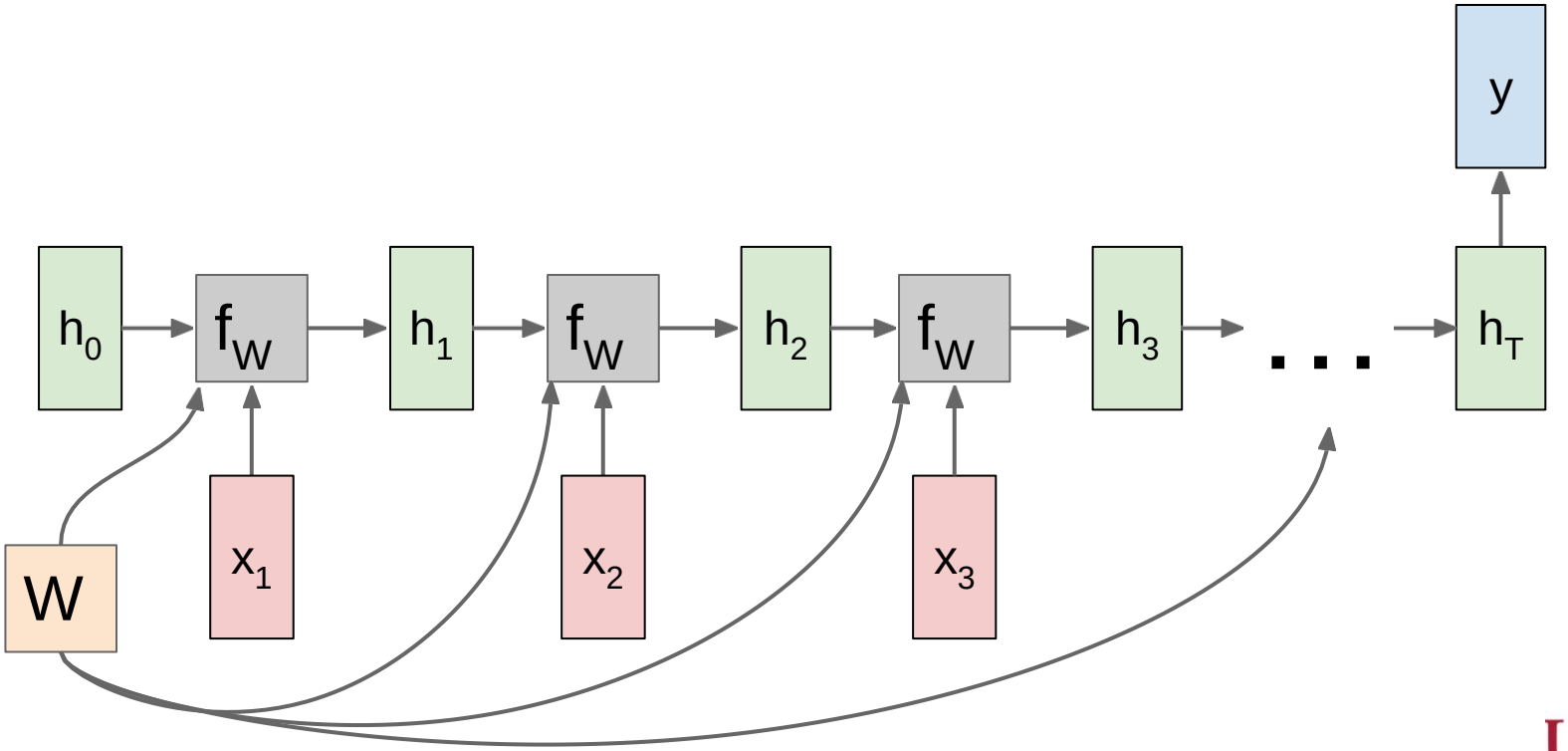
$x_1, x_2, x_3, \ldots x_t$: These are the input data points at each time step

$h_1, h_2, h_3, \ldots h_t$: These are the hidden states at each time step

W: This represents the **shared** weights that are learned during training and are applied at each time step to compute the hidden state

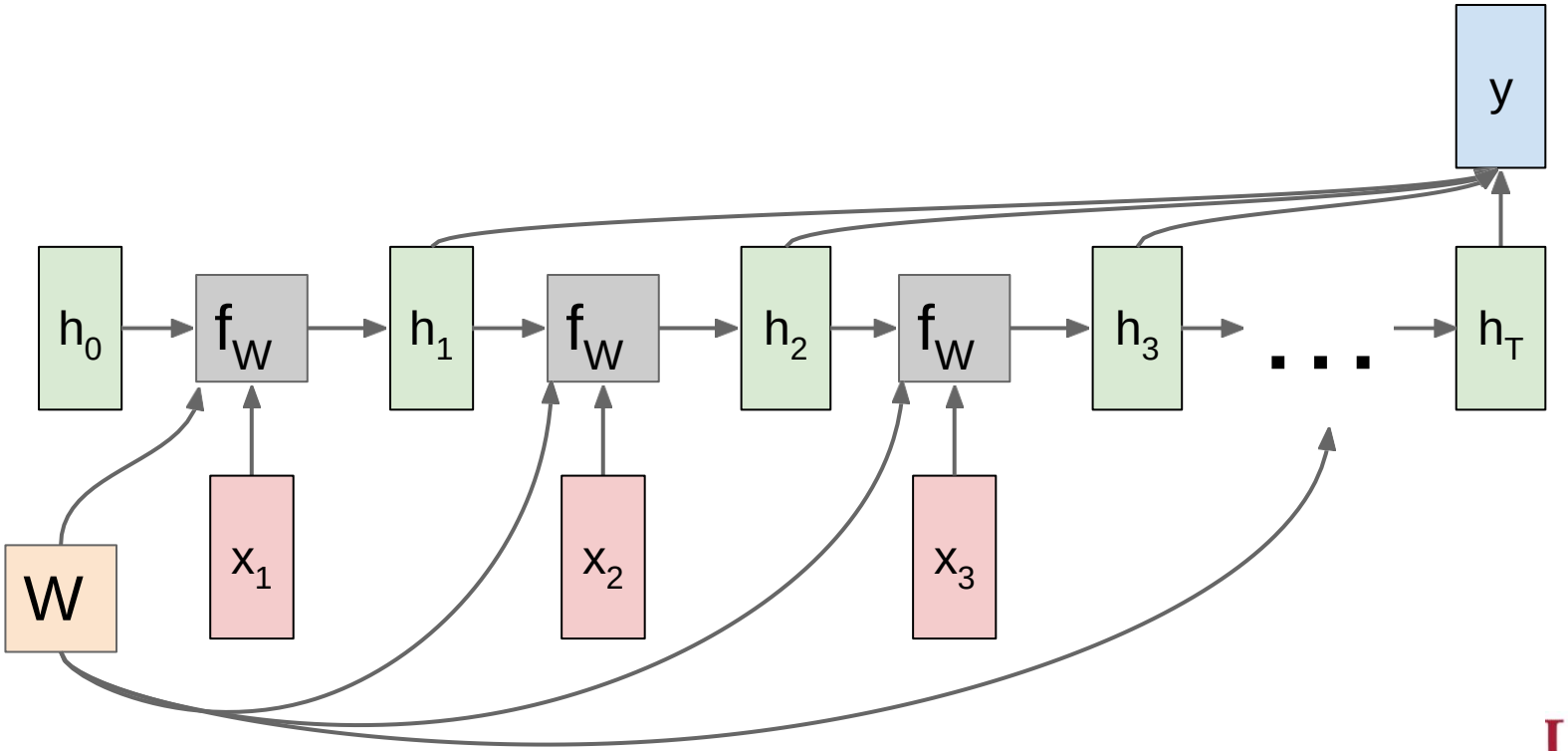
Credit to Stanford cs231n lecture slides

RNN: Computational Graph: Many to One



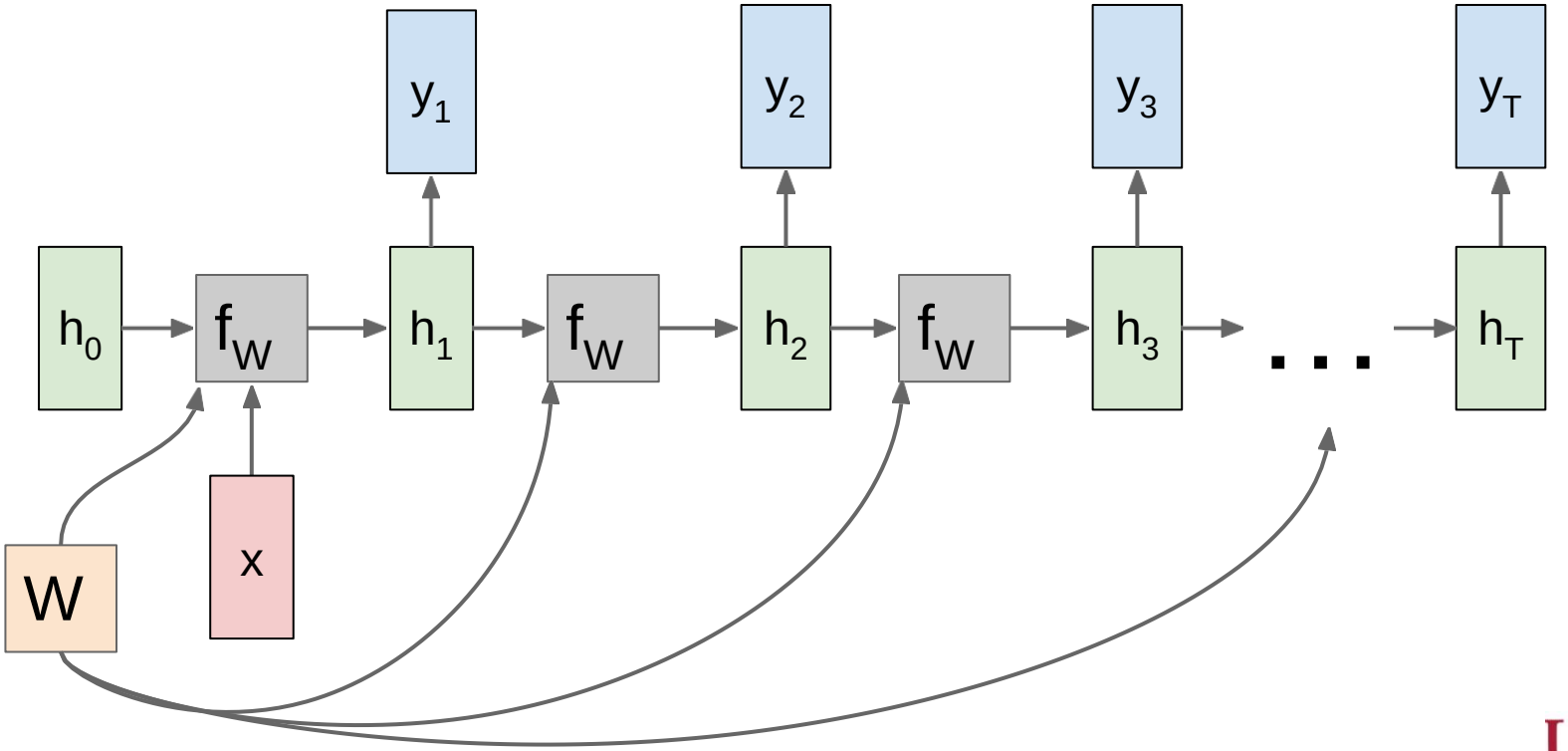
Credit to Stanford cs231n lecture slides

RNN: Computational Graph: Many to One



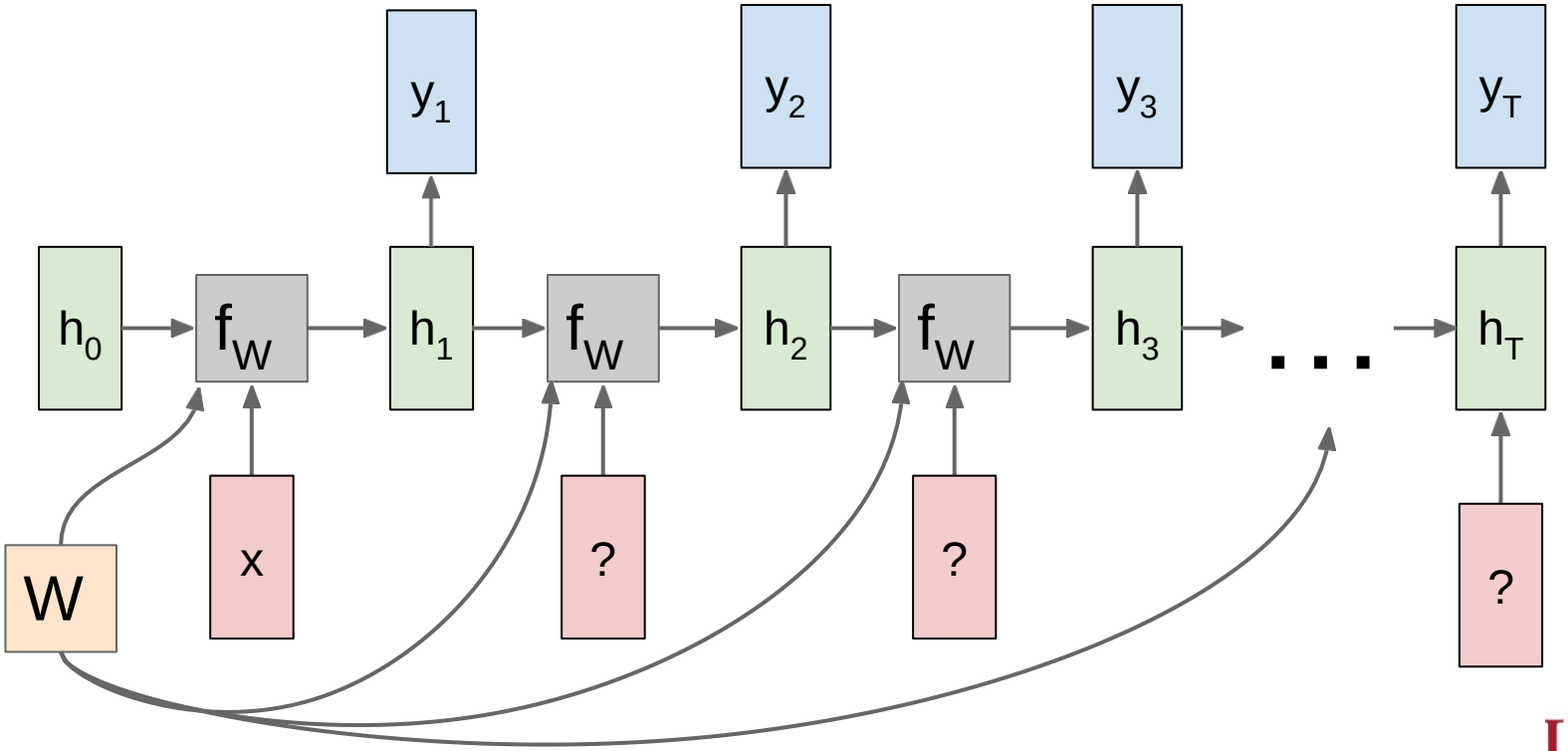
Credit to Stanford cs231n lecture slides

RNN: Computational Graph: One to Many



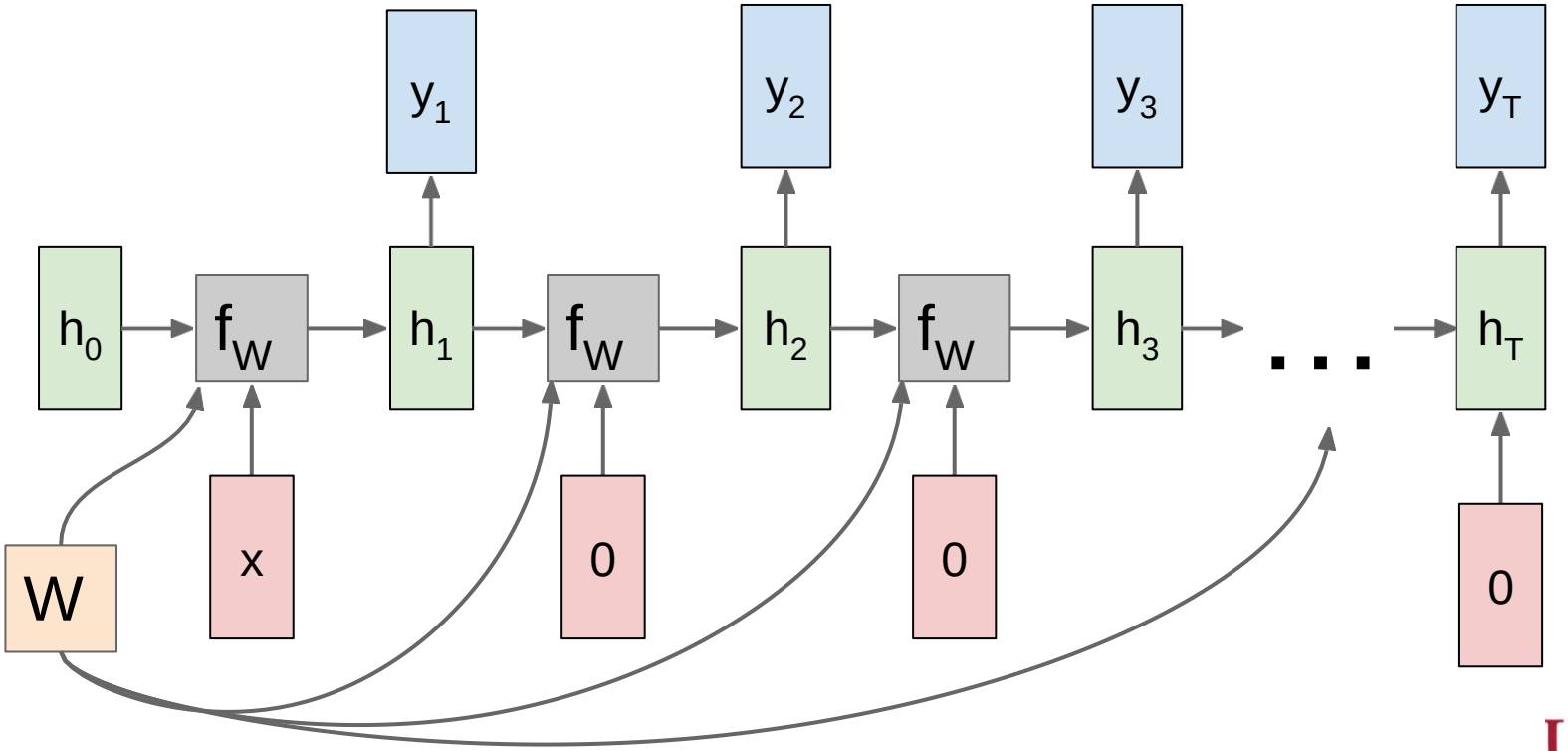
Credit to Stanford cs231n lecture slides

RNN: Computational Graph: One to Many



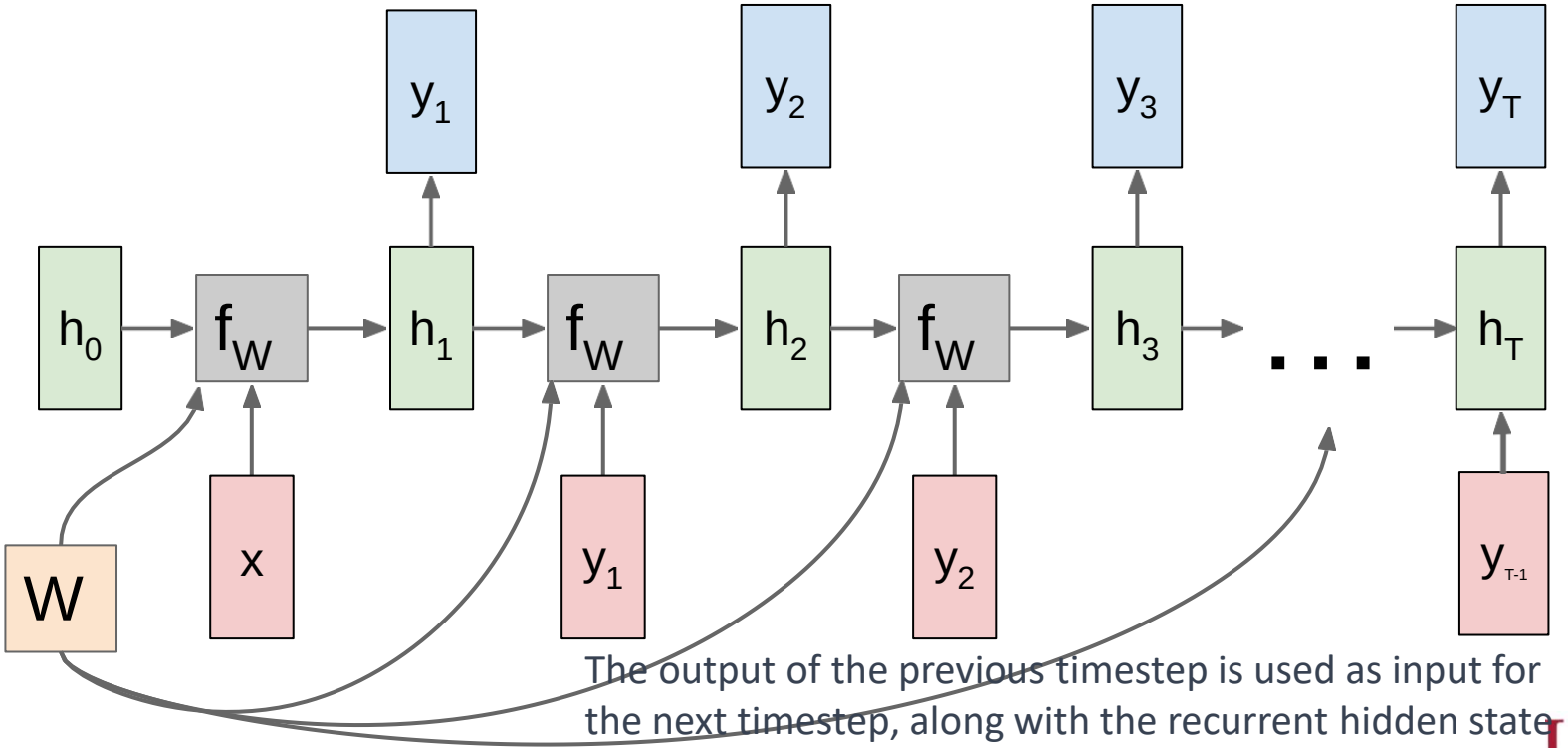
Credit to Stanford cs231n lecture slides

RNN: Computational Graph: One to Many



Credit to Stanford cs231n lecture slides

RNN: Computational Graph: One to Many

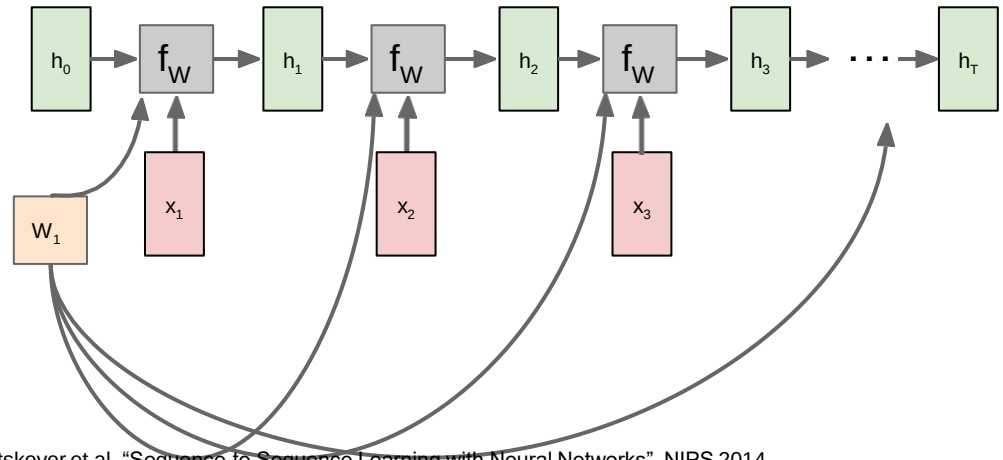


The output of the previous timestep is used as input for the next timestep, along with the recurrent hidden state

Credit to Stanford cs231n lecture slides

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

Many to one: Encode input sequence in a single vector



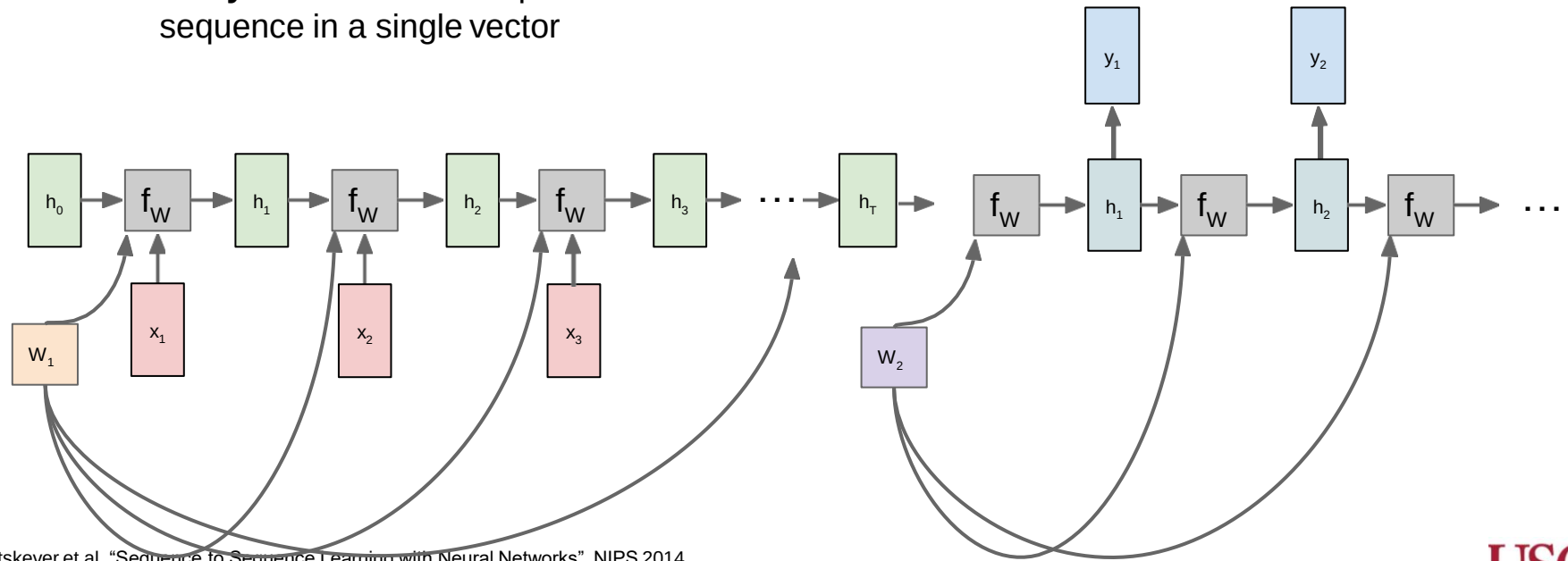
Sutskever et al, "Sequence to Sequence Learning with Neural Networks", NIPS 2014

Credit to Stanford cs231n lecture slides

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

Many to one: Encode input sequence in a single vector

One to many: Produce output sequence from single input vector



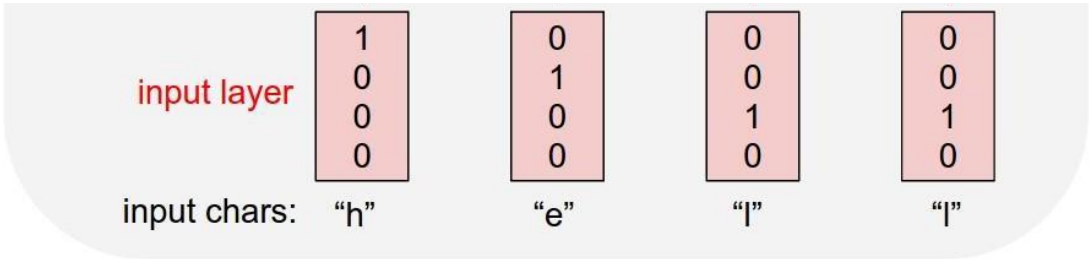
Sutskever et al, "Sequence to Sequence Learning with Neural Networks", NIPS 2014

Credit to Stanford cs231n lecture slides

Example: Character-level Language Model

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”



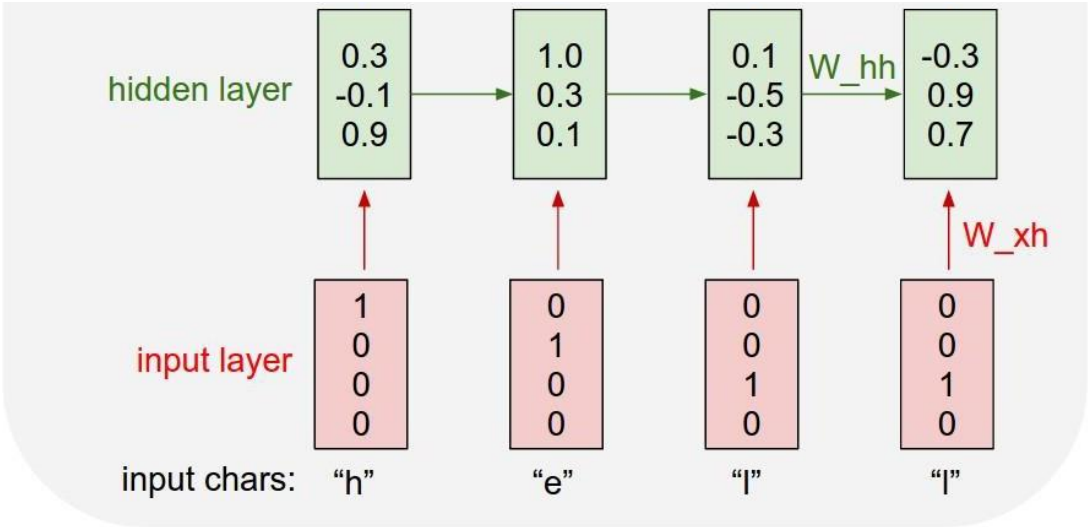
Credit to Stanford cs231n lecture slides

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”

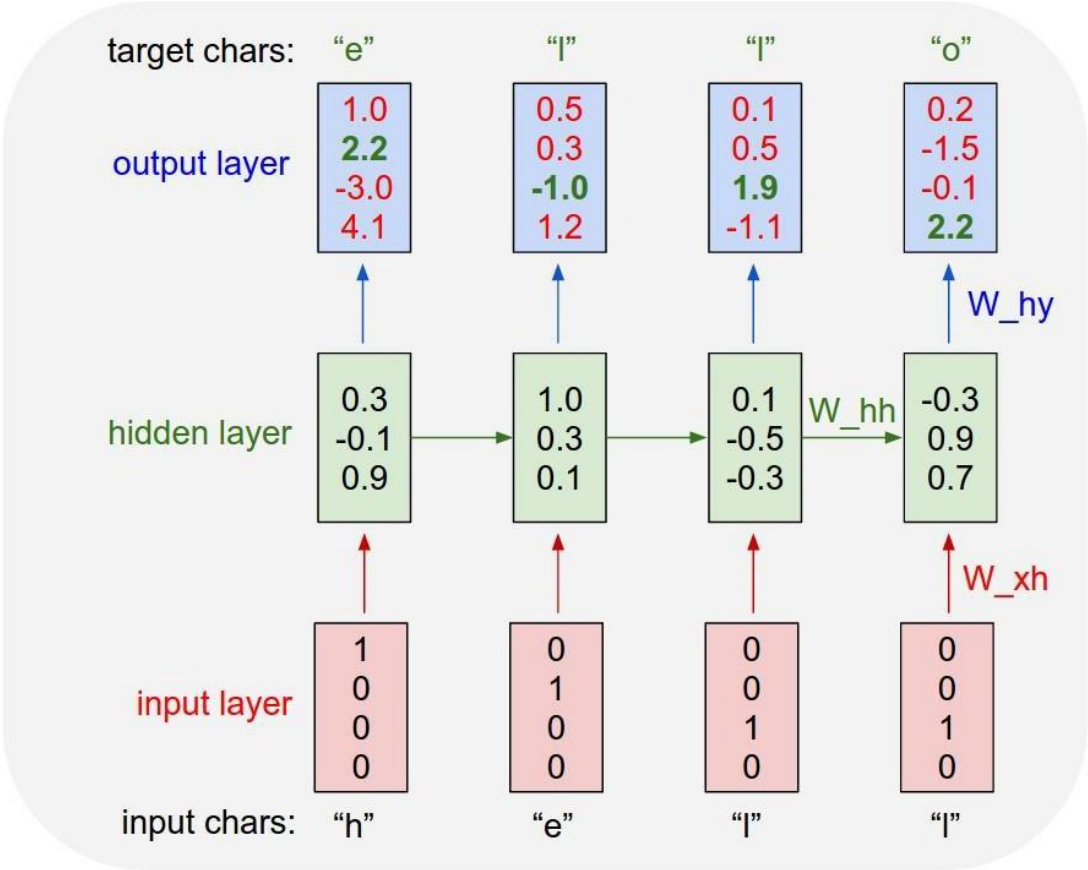


Credit to Stanford cs231n lecture slides

Example: Character-level Language Model

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

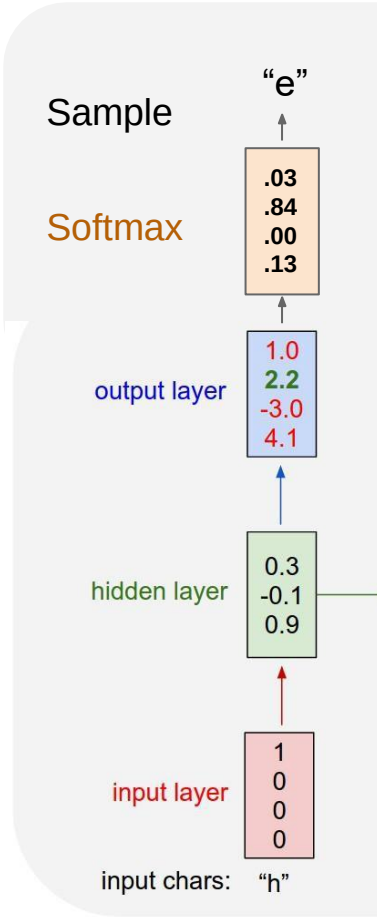


Credit to Stanford cs231n lecture slides

Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

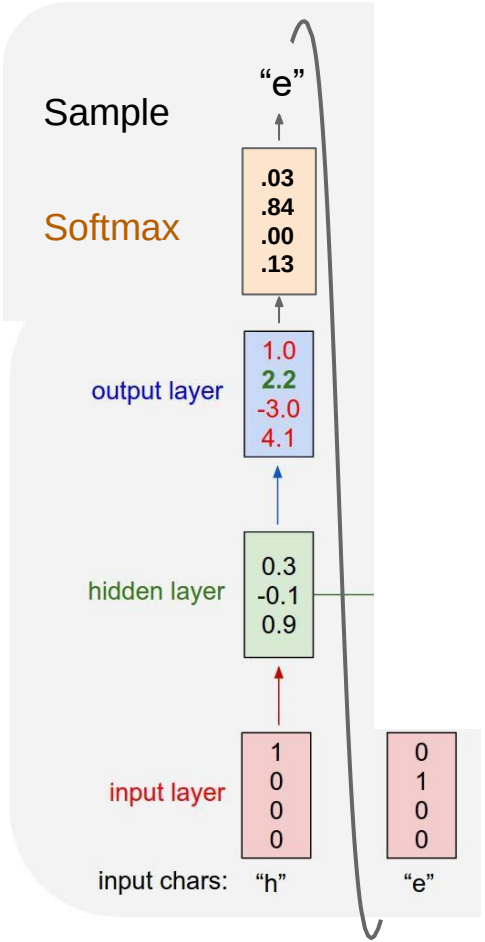
At test-time sample characters one at a time, feed back to model



Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

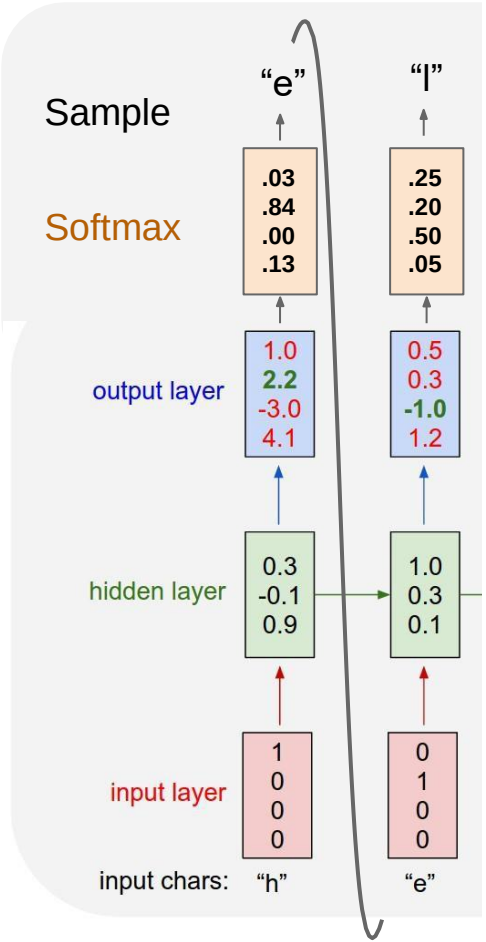
At test-time sample characters one at a time, feed back to model



Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

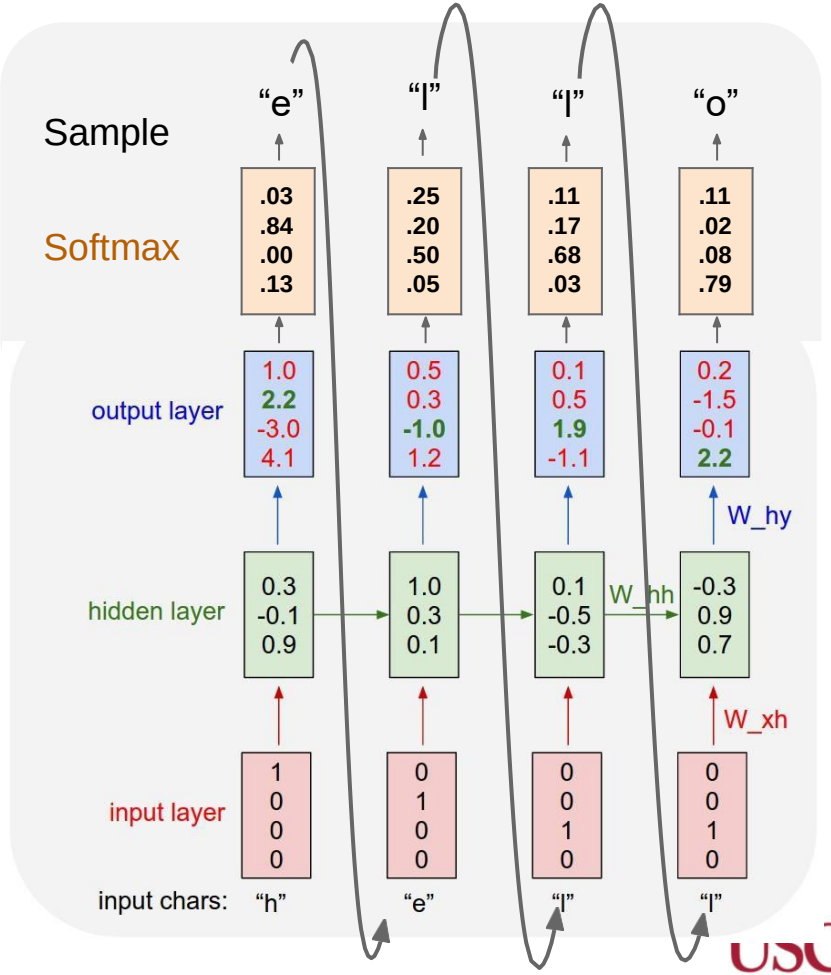
At test-time sample characters one at a time, feed back to model



Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

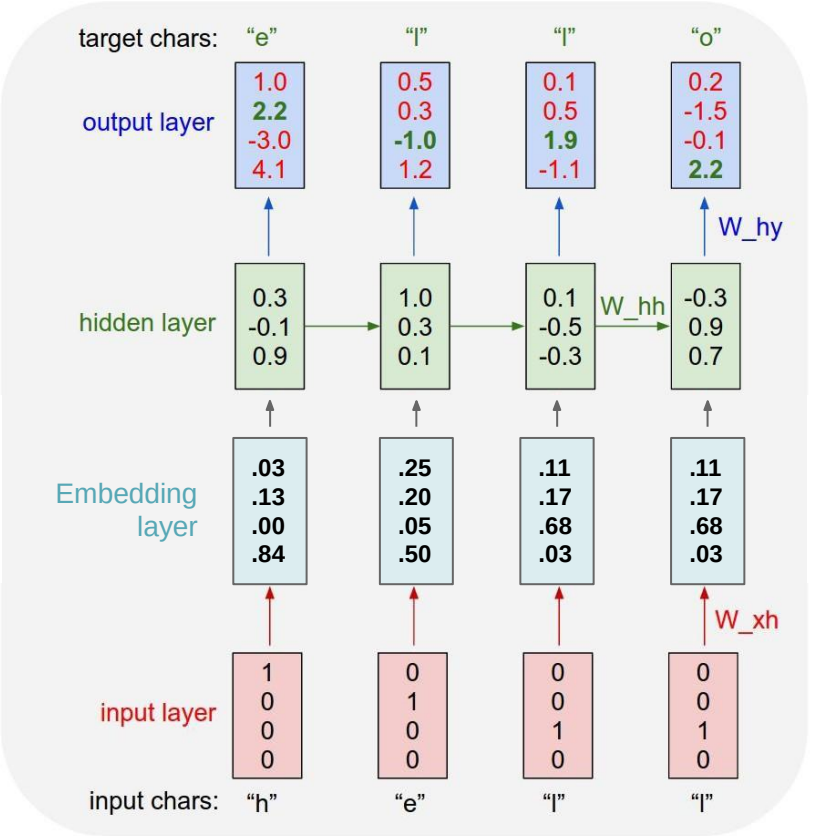
At test-time sample characters one at a time, feed back to model



Example: Character-level Language Model Sampling

$$\begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{14} \\ w_{31} & w_{32} & w_{33} & w_{14} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} w_{11} \\ w_{21} \\ w_{31} \end{bmatrix}$$

Matrix multiply with a one-hot vector just extracts a column from the weight matrix. We often put a separate **embedding** layer between input and hidden layers.

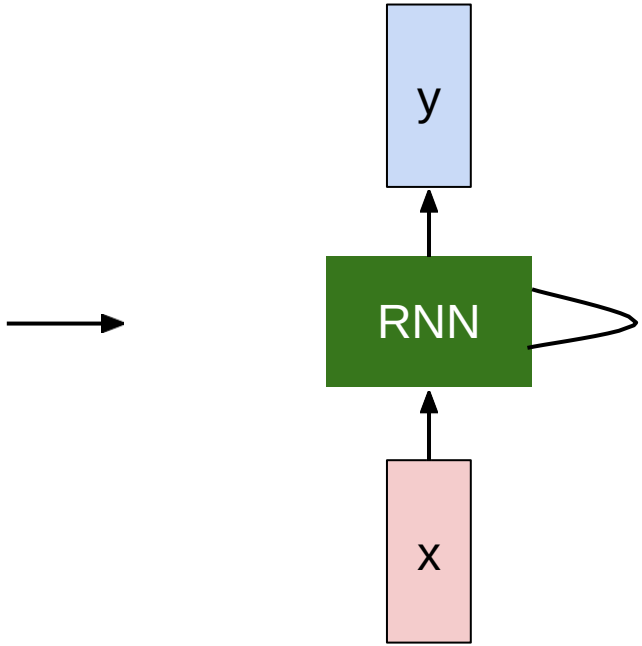


THE SONNETS

by William Shakespeare

From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as the riper should by time decease,
His tender heir might bear his memory:
But thou, contracted to thine own bright eyes,
Feed'st thy light's flame with self-substantial fuel,
Making a famine where abundance lies,
Thyself thy foe, to thy sweet self too cruel:
Thou that art now the world's fresh ornament,
And only herald to the gaudy spring,
Within thine own bud buriest thy content,
And tender churl mak'st waste in niggarding:
    Pity the world, or else this glutton be,
    To eat the world's due, by the grave and thee.

When forty winters shall besiege thy brow,
And dig deep trenches in thy beauty's field,
Thy youth's proud livery so gazed on now,
Will be a tatter'd weed of small worth held:
Then being asked, where all thy beauty lies,
Where all the treasure of thy lusty days;
To say, within thine own deep sunken eyes,
Were an all-eating shame, and thriftless praise.
How much more praise deserv'd thy beauty's use,
If thou couldst answer 'This fair child of mine
Shall sum my count, and make my old excuse,'
Proving his beauty by succession thine!
    This were to be new made when thou art old,
    And see thy blood warm when thou feel'st it cold.



at first:

```
tyntd-iafhatawiaoihrdemot  lytdws  e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e
plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng
```

train more

```
"Tmont thithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome
coaniogennc Phe lism thond hon at. MeiDimorotion in ther thize."
```

train more

```
Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
```

train more

```
"Why do what that day," replied Natasha, and wishing to himself the fact the
princess, Princess Mary was easier, fed in had oftened him.
Pierre aking his soul came to the packs and drove up his father-in-law women.
```

PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

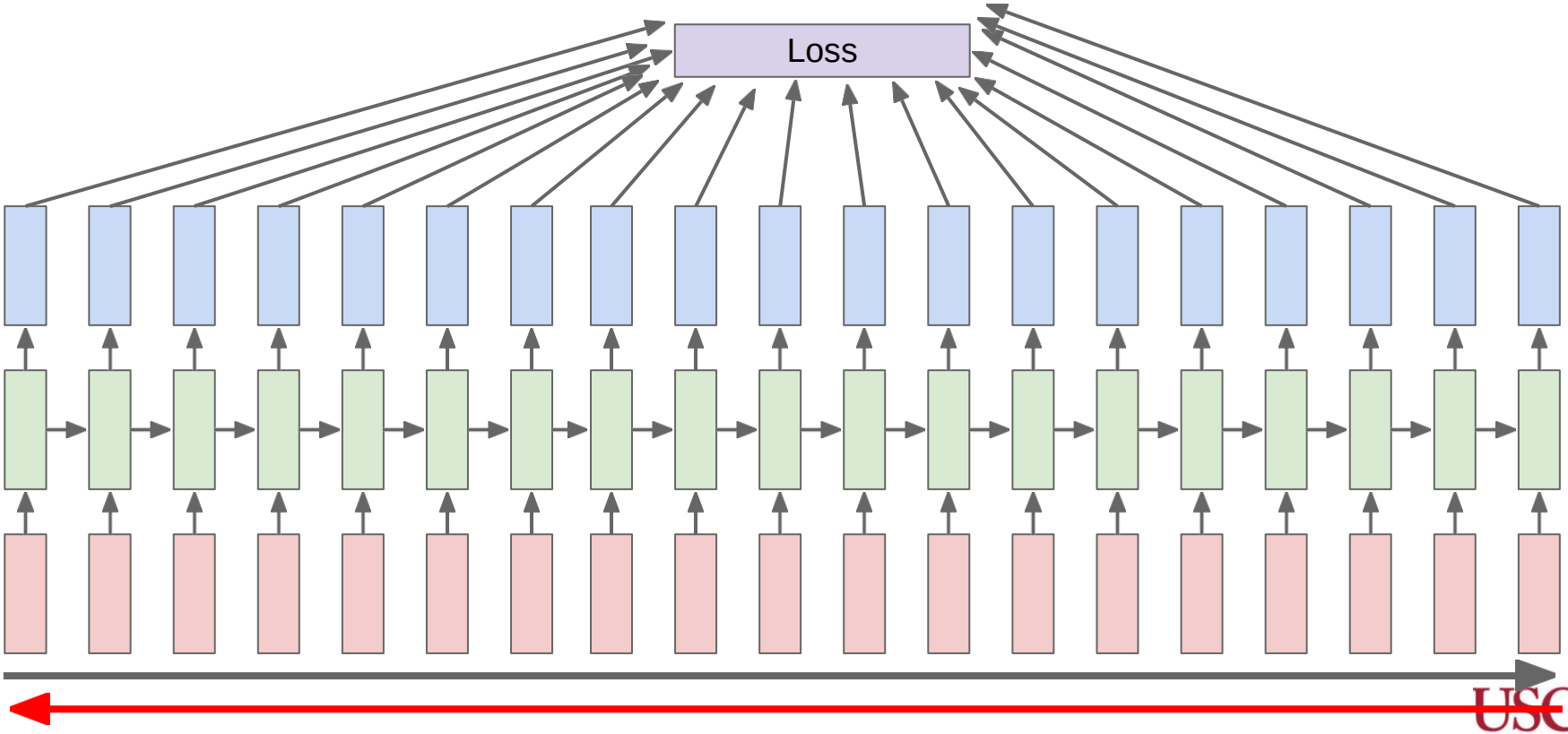
VIOLA:
I'll drink it.

VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

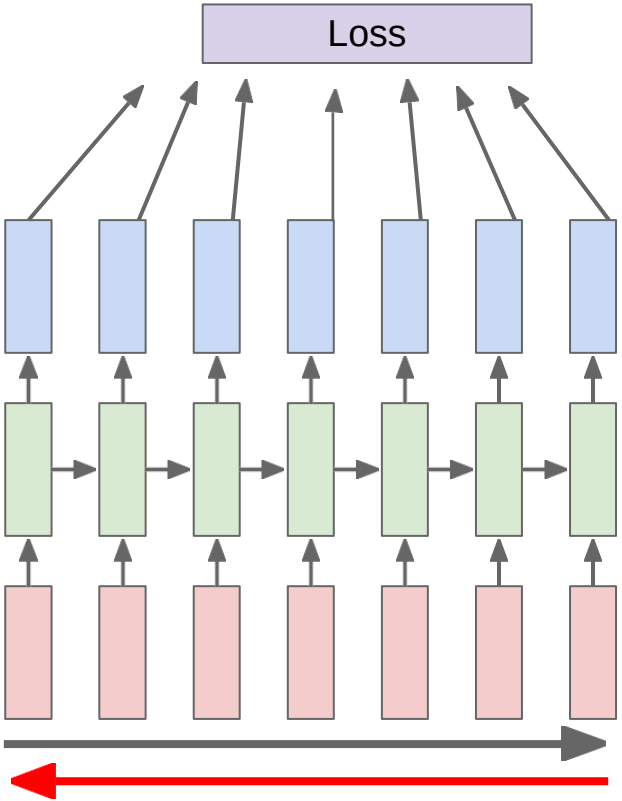
KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.

Backpropagation through time

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

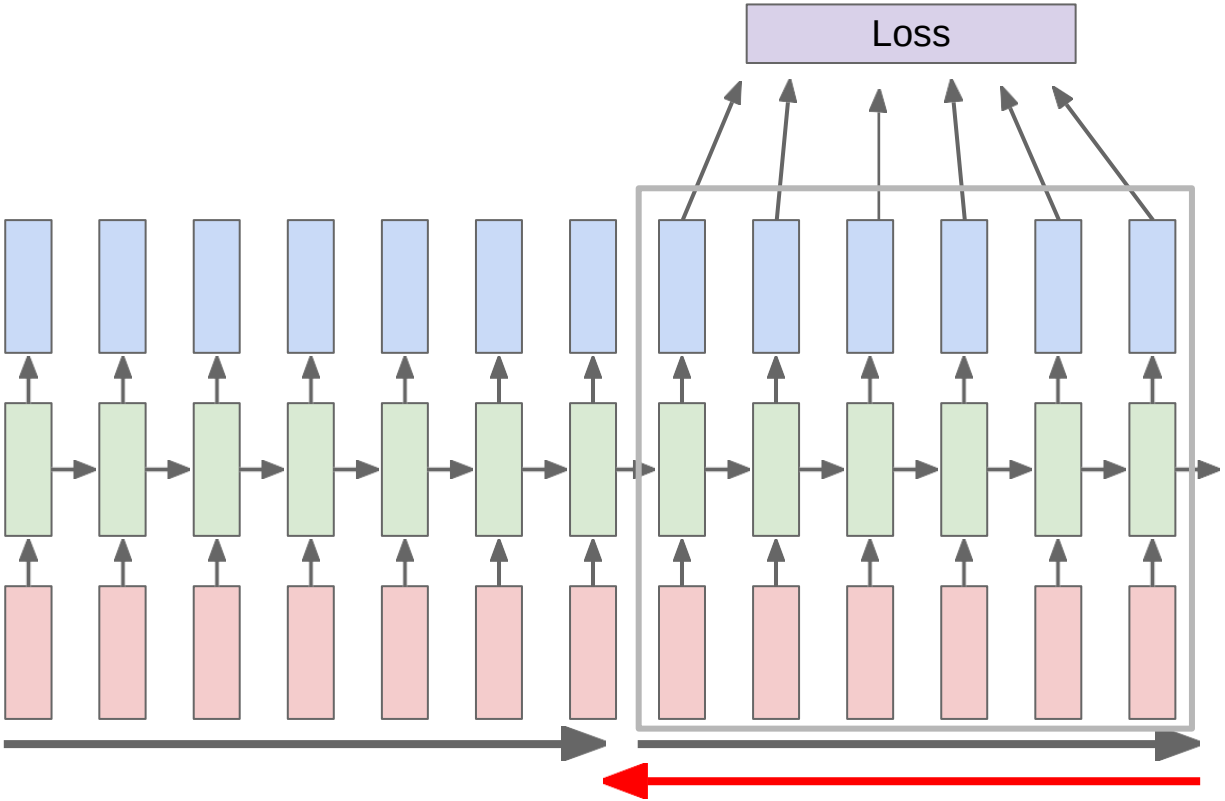


Truncated Backpropagation through time



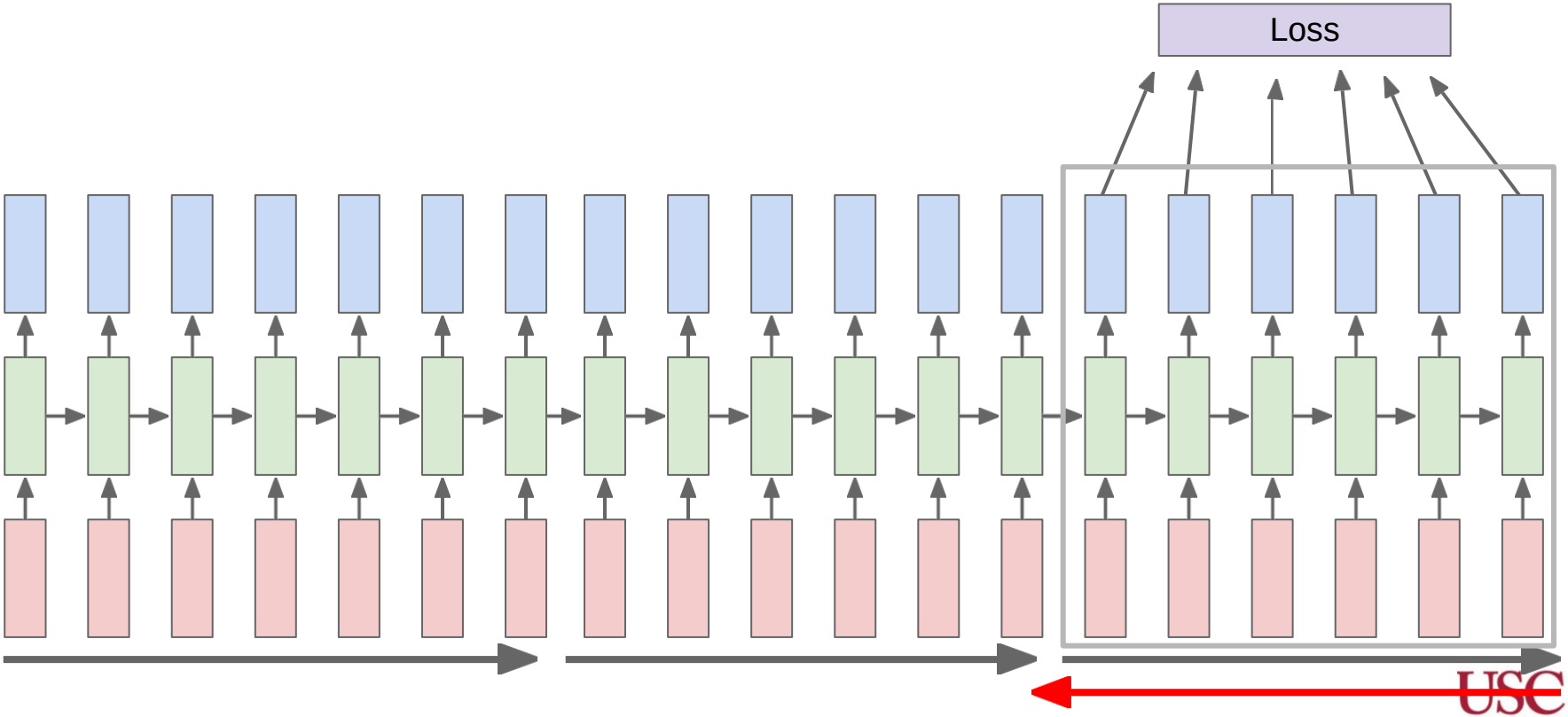
Run forward and backward through chunks of the sequence instead of whole sequence

Truncated Backpropagation through time



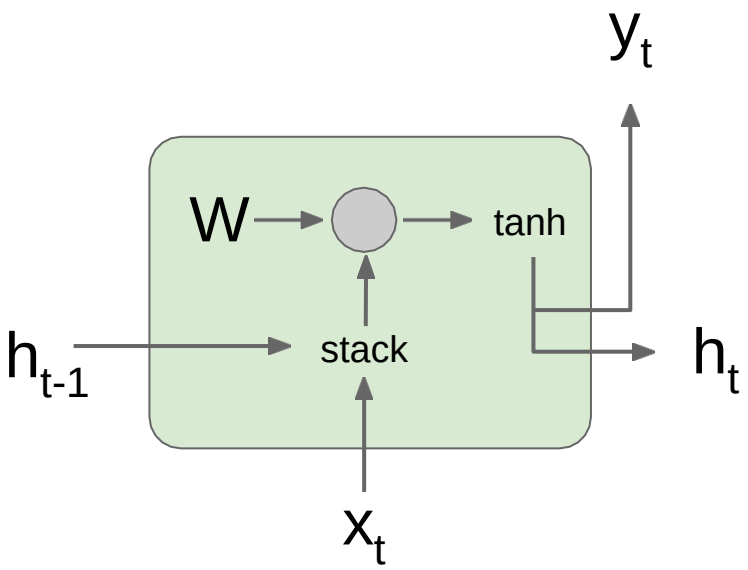
Carry hidden states forward in time forever, but only backpropagate for some smaller number of steps

Truncated Backpropagation through time



Vanilla RNN Gradient Flow

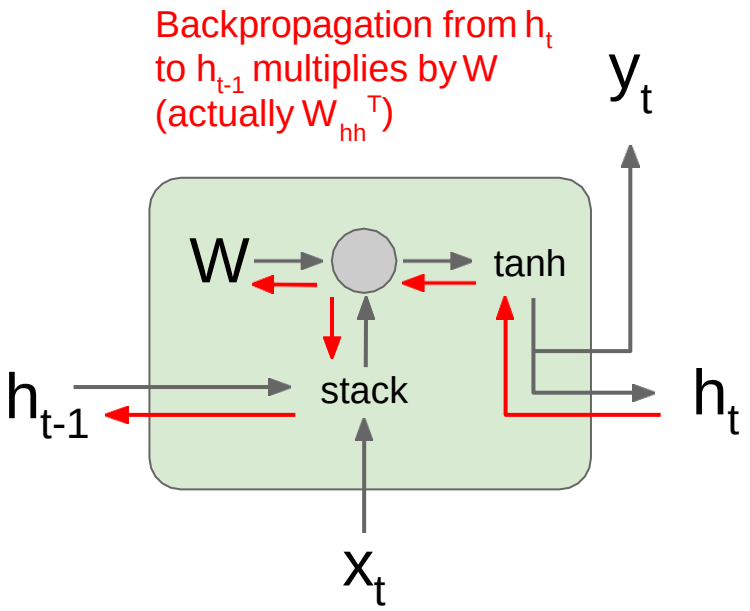
Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Pascanu et al, "On the difficulty of training recurrent neural networks", ICML 2013



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

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

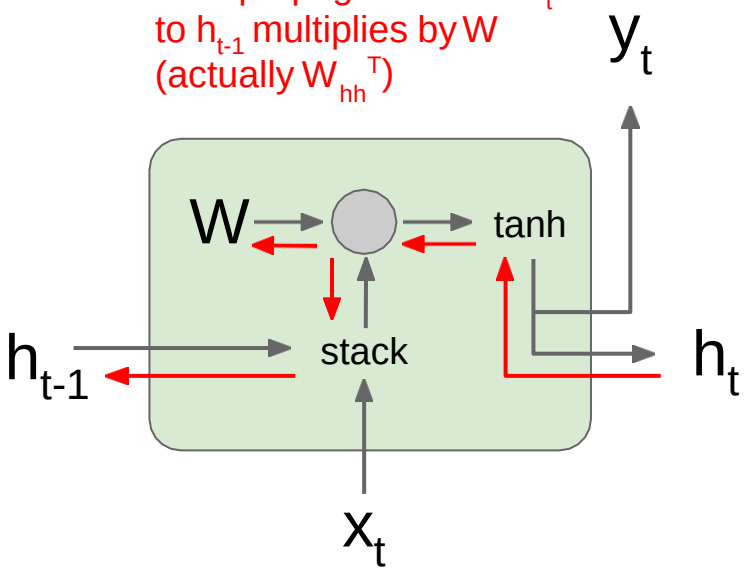


$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$
$$= \tanh\left((W_{hh} \quad W_{hx})\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$
$$= \tanh\left(W\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

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

Backpropagation from h_t to h_{t-1} multiplies by W (actually W_{hh}^T)

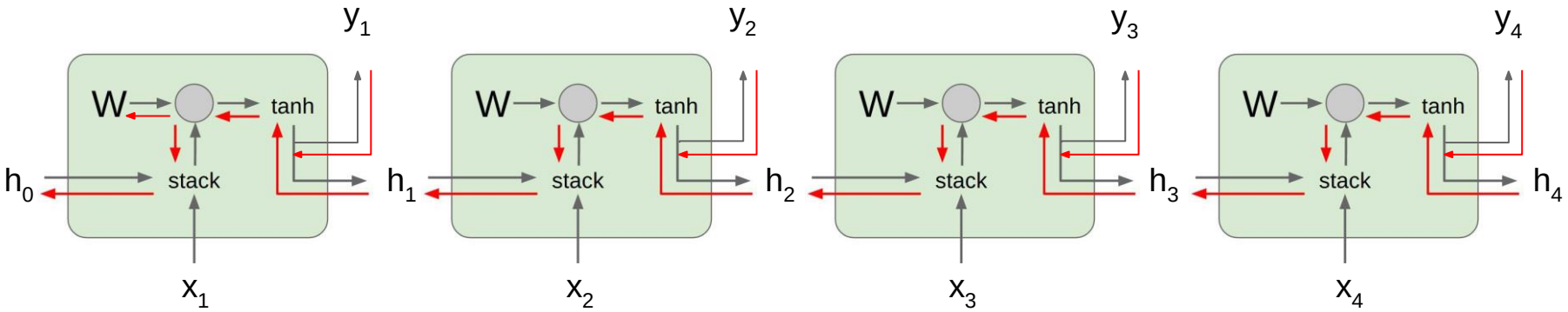


$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$
$$= \tanh\left((W_{hh} \quad W_{hx})\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$
$$= \tanh\left(W\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

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

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

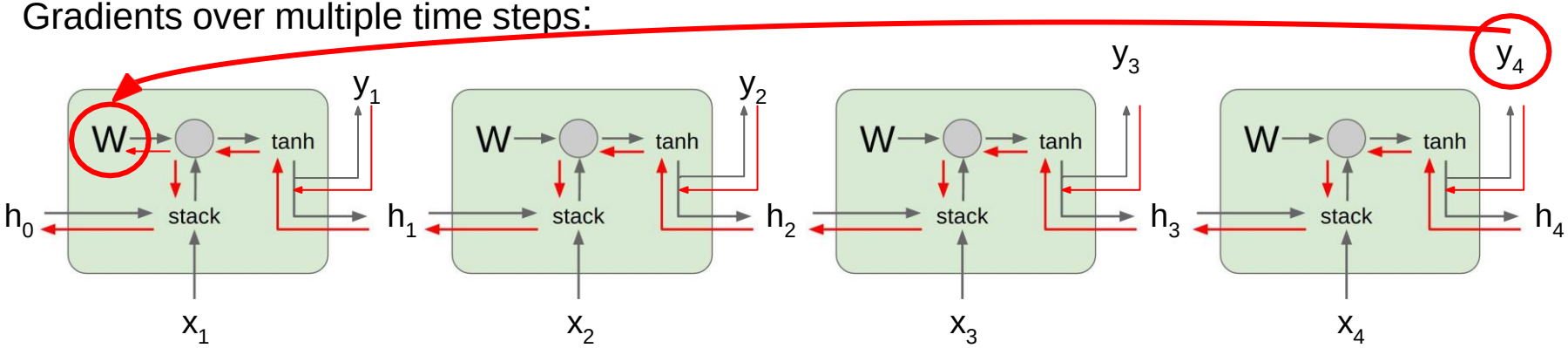


$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

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:



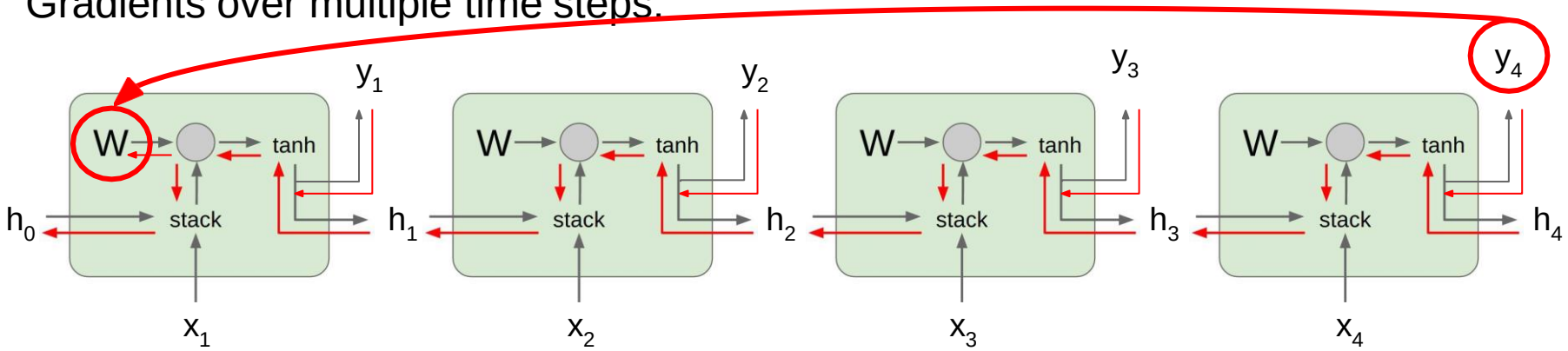
$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

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

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:



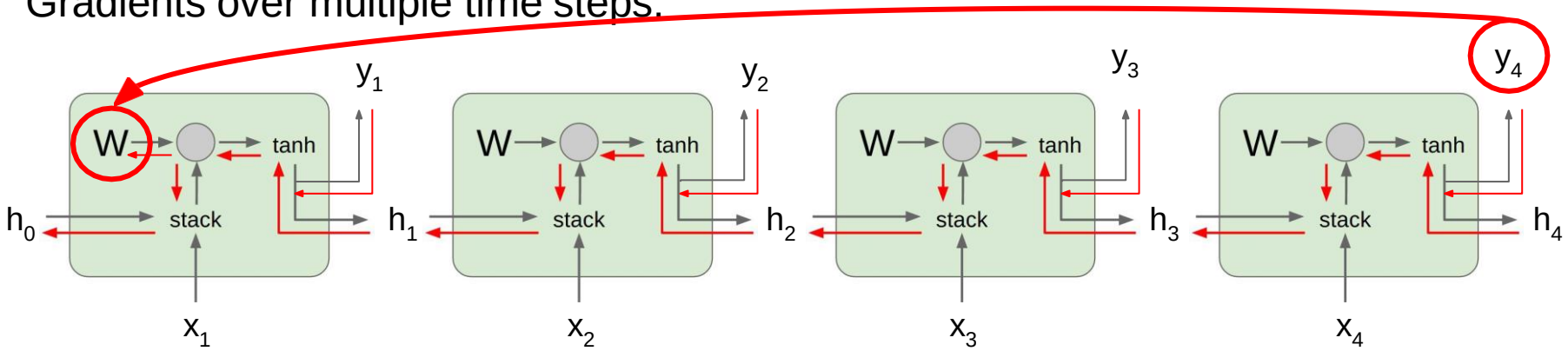
$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

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

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:



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

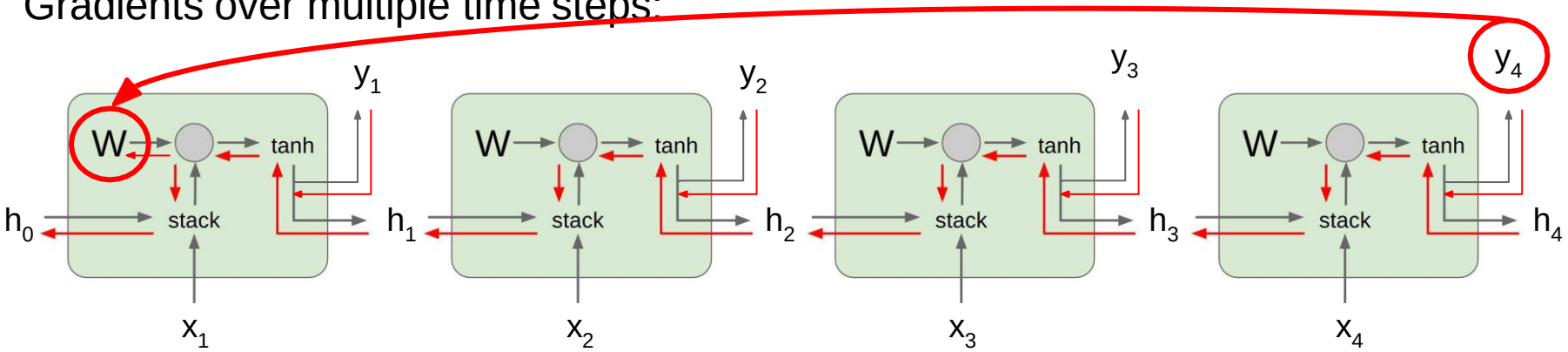
$$\frac{\partial h_t}{\partial h_{t-1}} = tanh'(W_{hh} h_{t-1} + W_{xh} x_t) W_{hh}$$

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

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

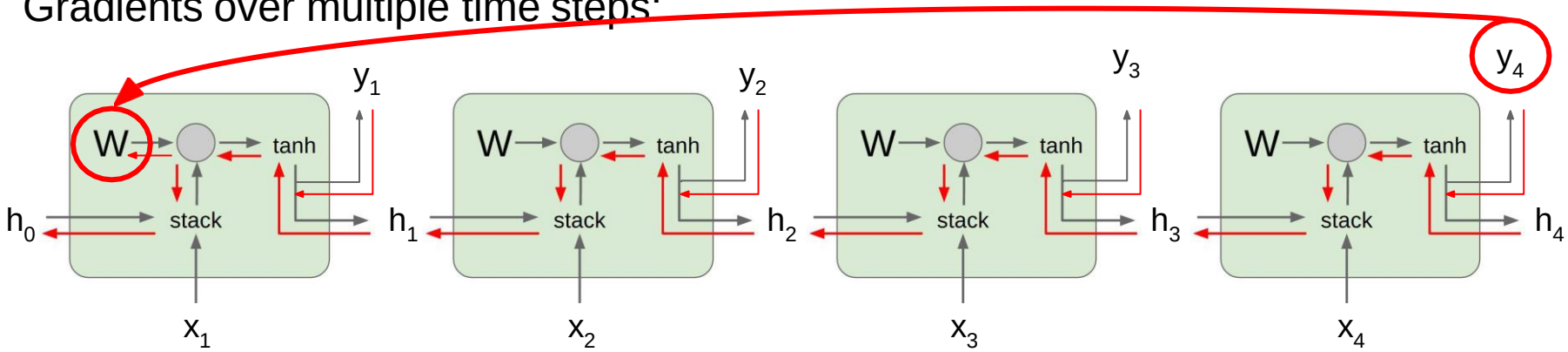
Vanishing gradients

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

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:



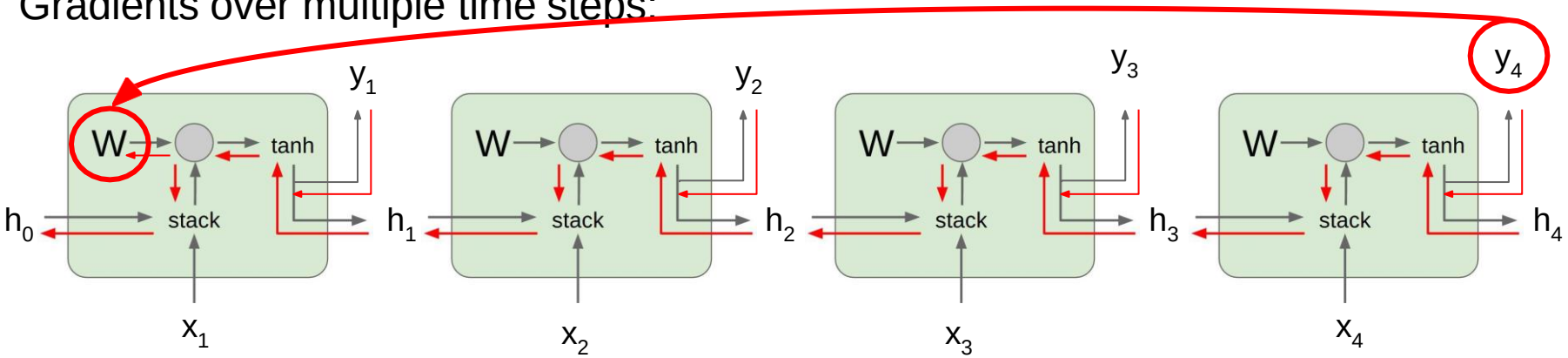
$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

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?

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

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

Largest singular value > 1:

Exploding gradients

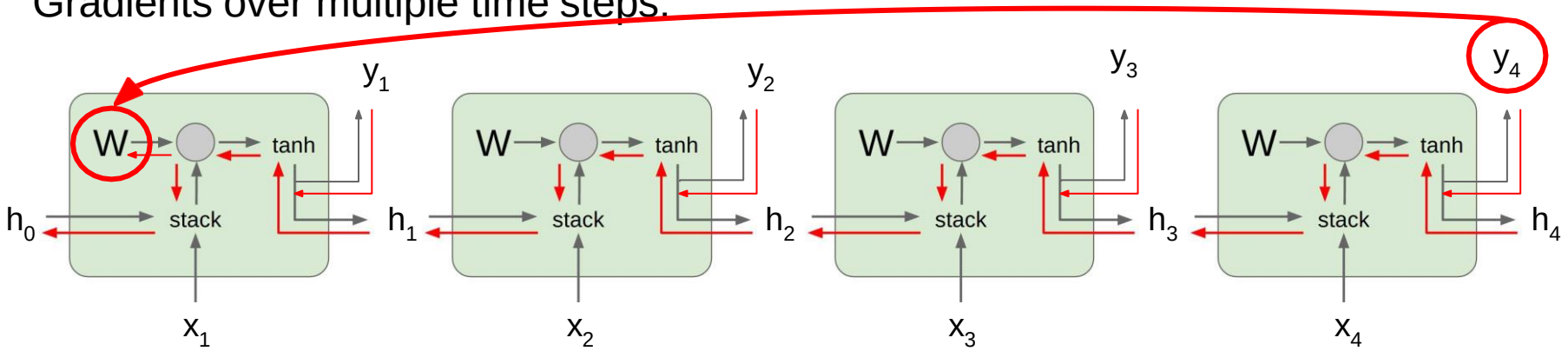
Largest singular value < 1:

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

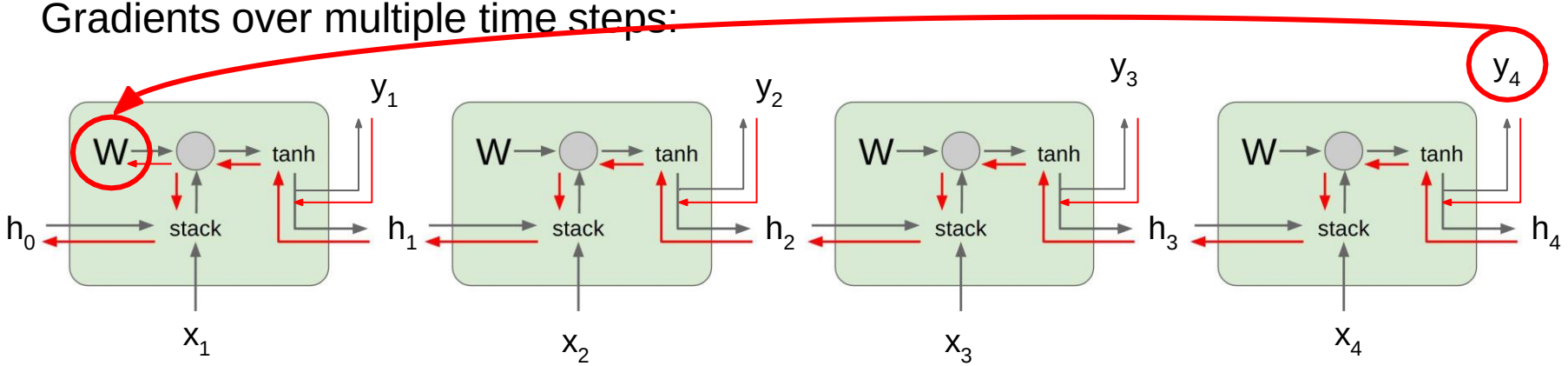
```
grad_norm = np.sum(grad * grad)
if grad_norm > threshold:
  grad *= (threshold / grad_norm)
```

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?

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

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

Largest singular value > 1:

Exploding gradients

Largest singular value < 1:
Vanishing gradients

→ Change RNN architecture

Long Short Term Memory (LSTM)

Details

Long Short Term Memory (LSTM)

- Learn **long-term** dependencies
  - Remember information for long periods of time is practically their default behavior

Hochreiter and Schmidhuber, “Long Short Term Memory”, Neural Computation 1997
Colah's blog: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short Term Memory (LSTM)

Vanilla RNN

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

LSTM

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

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

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

e.g., $g = tanh(W_{gh}h_{t-1} + W_{gx}x_t)$

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)

LSTM

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

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

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

$$W = \begin{bmatrix} W_i \\ W_f \\ W_o \\ W_g \end{bmatrix}$$

Long Short Term Memory (LSTM)

LSTM

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

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

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

$$W = \begin{bmatrix} W_i \\ W_f \\ W_o \\ W_g \end{bmatrix}$$

$$i = \sigma(W_i \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix})$$

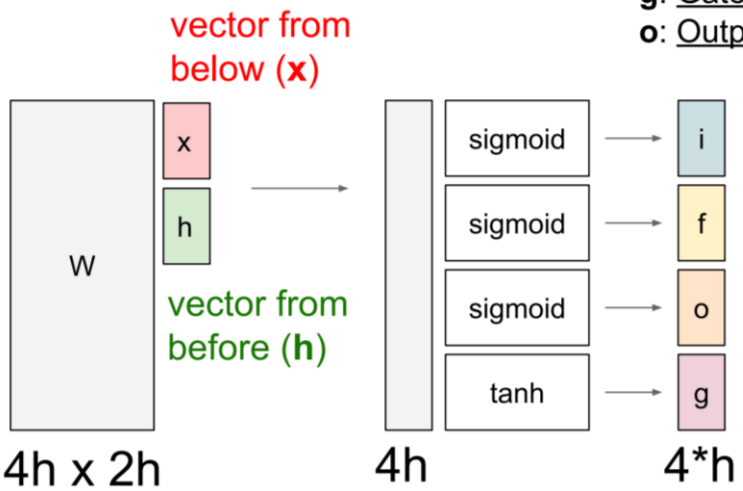
$$f = \sigma(W_f \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix})$$

$$o = \sigma(W_o \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix})$$

$$g = tanh(W_g \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix})$$

Long Short Term Memory (LSTM)

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



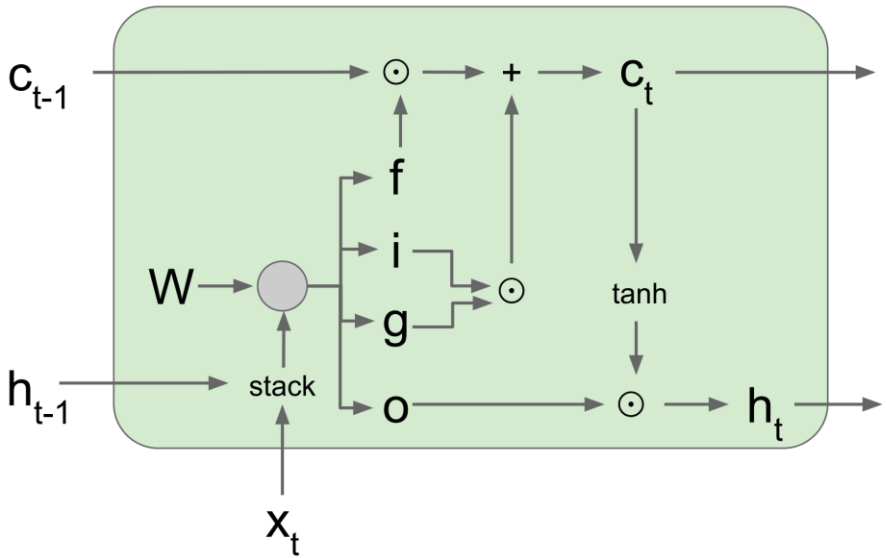
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



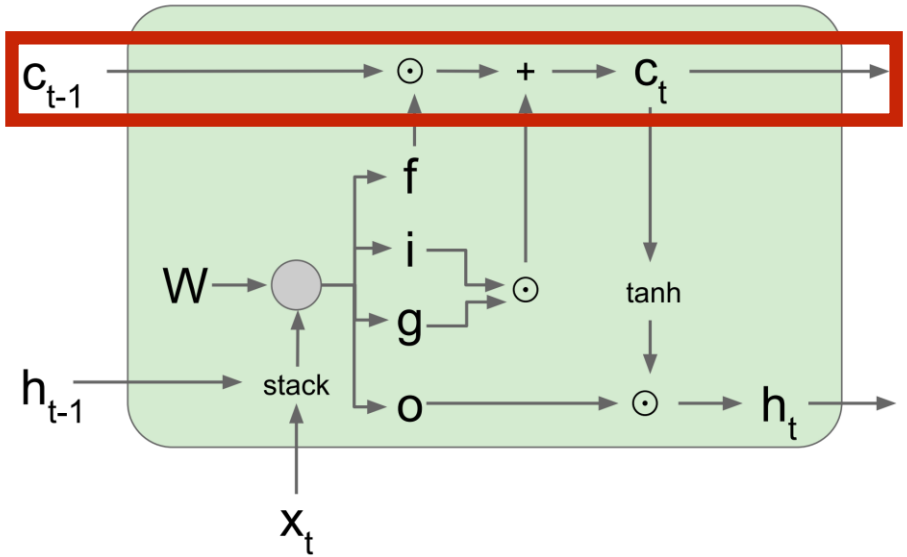
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



- Cell state: information to pass to the next state

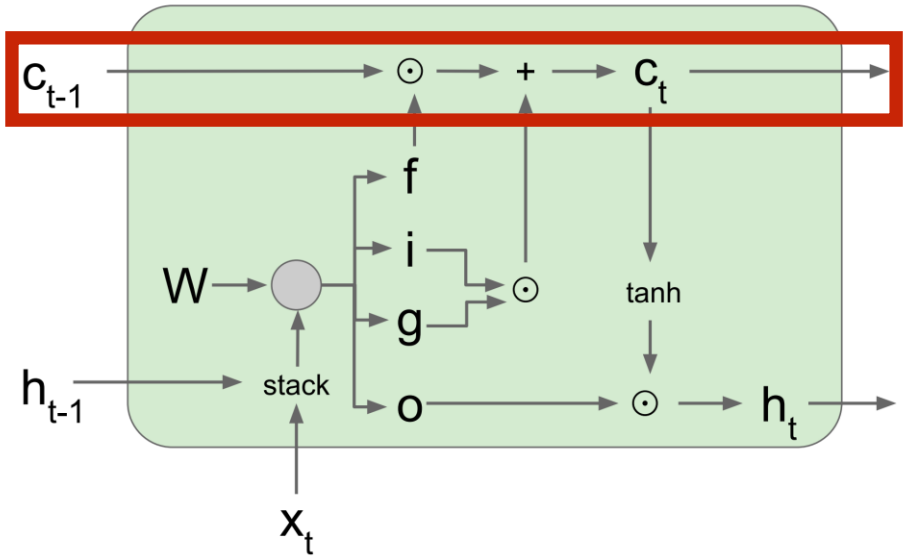
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



- Cell state: information to pass to the next state

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

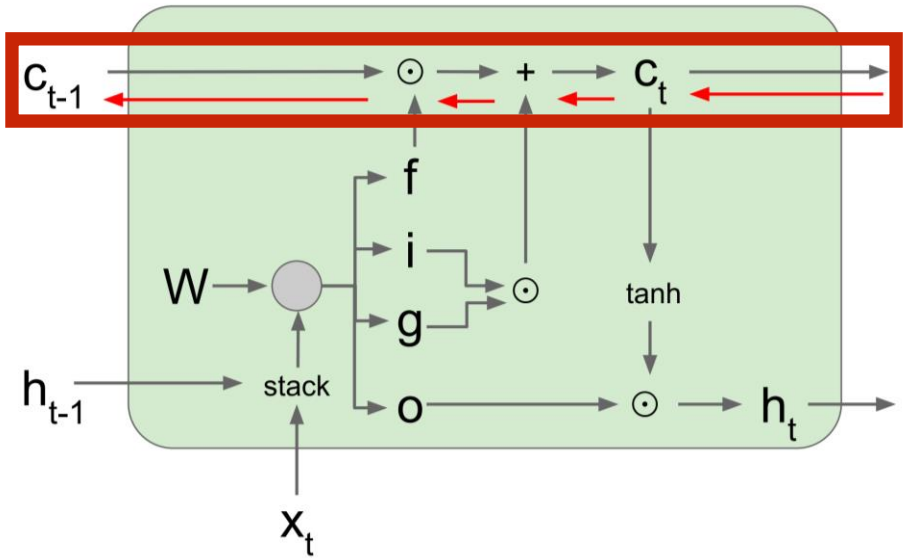
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



- Cell state: information to pass to the next state

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

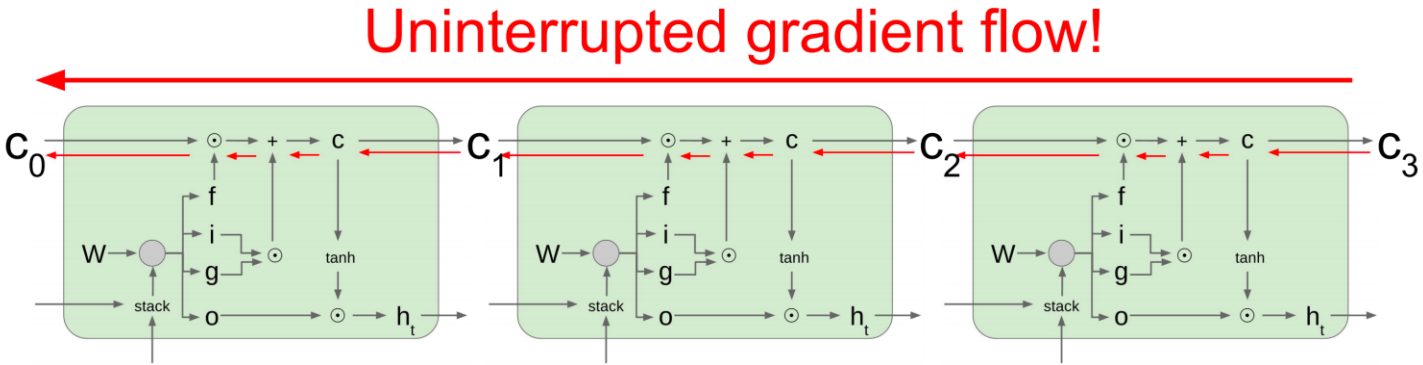
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

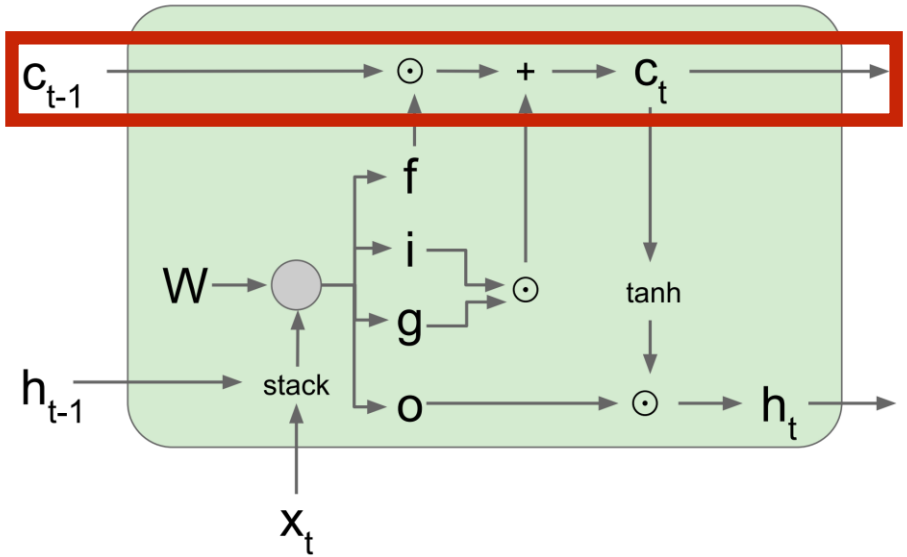
Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



- Cell state: information to pass to the next state

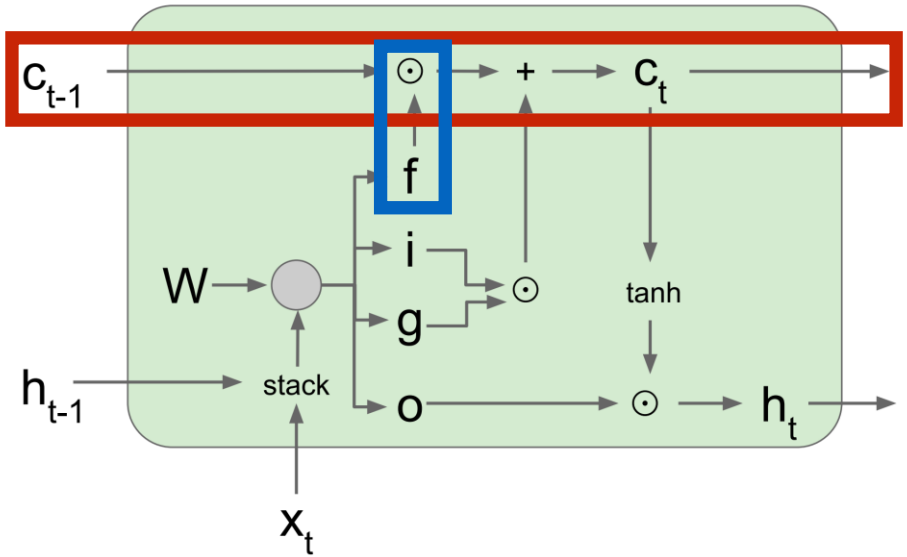
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



- Cell state: information to pass to the next state
- Forget gate: how much information to pass (sigmoid)

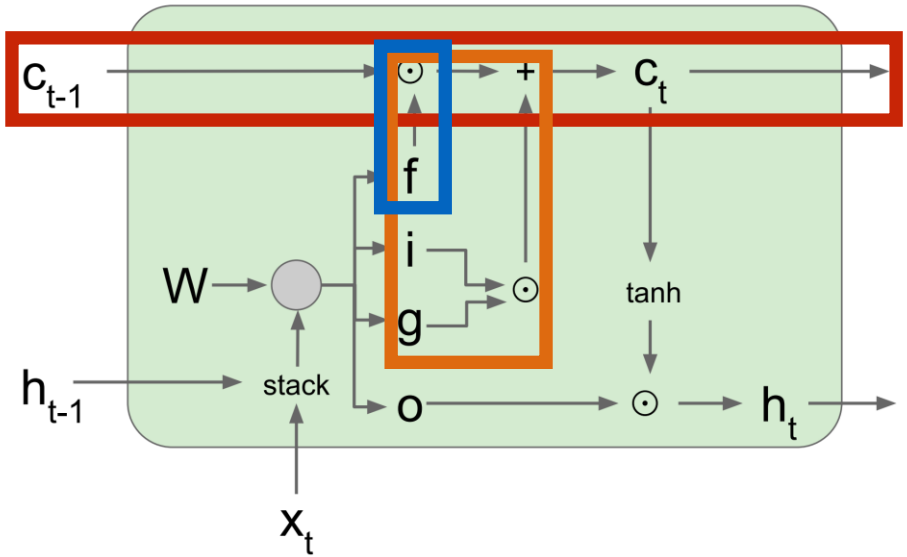
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



- Cell state: information to pass to the next state
- Forget gate: how much information to pass (sigmoid)
- **Input gate:** how much information to add (sigmoid)
- **Gate gate:** which information to add (tanh)

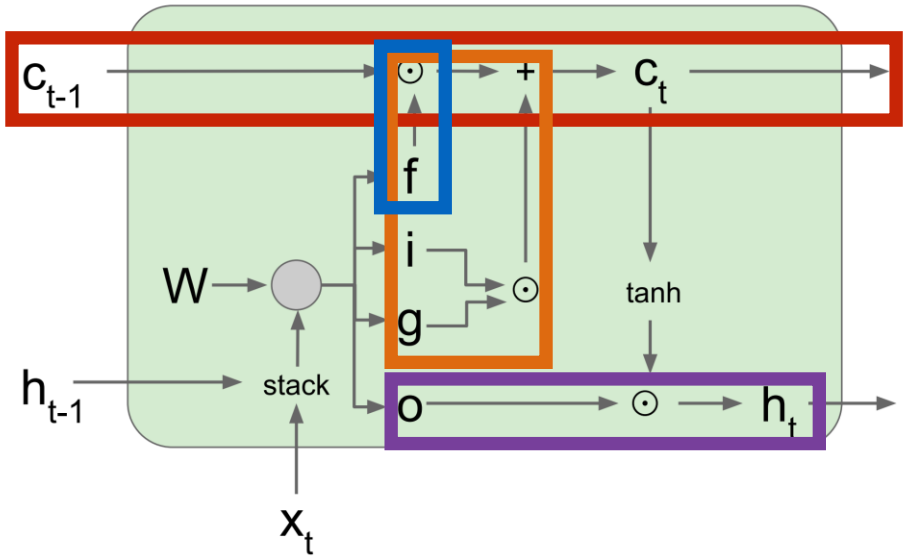
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Long Short Term Memory (LSTM)



- Cell state: information to pass to the next state
- Forget gate: how much information to pass (sigmoid)
- **Input gate:** how much information to add (sigmoid)
- **Gate gate:** which information to add (tanh)
- **Output gate:** what we are going to output (sigmoid)

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

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

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

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation 1997

Other variants

GRU [*Learning phrase representations using rnn encoder-decoder for statistical machine translation*, Cho et al. 2014]

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r)$$
$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z)$$
$$\tilde{h}_t = \tanh(W_{xh}x_t + W_{hh}(r_t \odot h_{t-1}) + b_h)$$
$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t$$

[*LSTM: A Search Space Odyssey*, Greff et al., 2015]

[*An Empirical Exploration of Recurrent Network Architectures*, Jozefowicz et al., 2015]

MUT1:

$$\begin{aligned} z &= \text{sigm}(W_{\text{xz}}x_t + b_{\text{z}}) \\ r &= \text{sigm}(W_{\text{xr}}x_t + W_{\text{hr}}h_t + b_{\text{r}}) \\ h_{t+1} &= \tanh(W_{\text{hh}}(r \odot h_t) + \tanh(x_t) + b_{\text{h}}) \odot z \\ &+ h_t \odot (1 - z) \end{aligned}$$

MUT2:

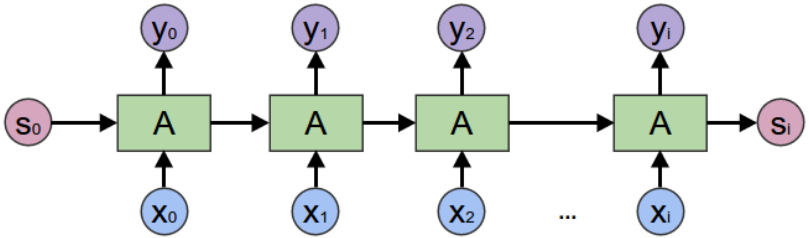
$$\begin{aligned} z &= \text{sigm}(W_{\text{xz}}x_t + W_{\text{hz}}h_t + b_{\text{z}}) \\ r &= \text{sigm}(x_t + W_{\text{hr}}h_t + b_{\text{r}}) \\ h_{t+1} &= \tanh(W_{\text{hh}}(r \odot h_t) + W_{xh}x_t + b_{\text{h}}) \odot z \\ &+ h_t \odot (1 - z) \end{aligned}$$

MUT3:

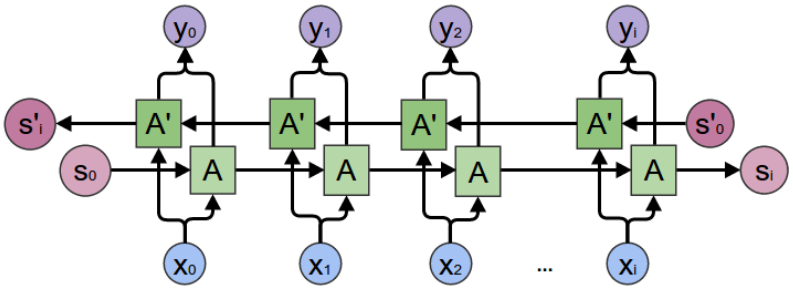
$$\begin{aligned} z &= \text{sigm}(W_{\text{xz}}x_t + W_{\text{hz}}\tanh(h_t) + b_{\text{z}}) \\ r &= \text{sigm}(W_{\text{xr}}x_t + W_{\text{hr}}h_t + b_{\text{r}}) \\ h_{t+1} &= \tanh(W_{\text{hh}}(r \odot h_t) + W_{xh}x_t + b_{\text{h}}) \odot z \\ &+ h_t \odot (1 - z) \end{aligned}$$

Other variants

RNN



Bidirectional RNN



Make predictions over a sequence with both **past** and **future** context.

Colah's blog: http://colah.github.io/posts/2015-09-NN-Types-FP/

RNN applications

- Image Captioning
- Image Captioning with Attention
- Question Answering
- Visual Question Answering
- Speech Recognition
- Action Recognition in Videos
- Text Parsing
- Machine Translation

Image Captioning

I look better than Tom Cruise.

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

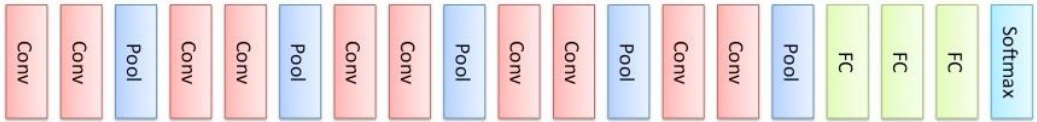


Image Captioning

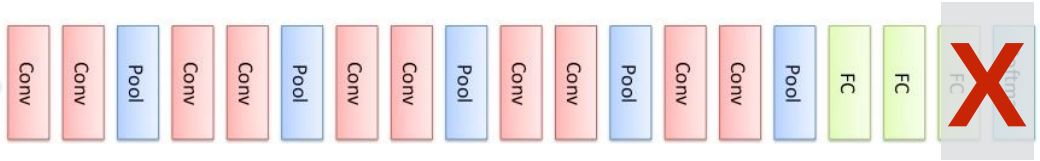
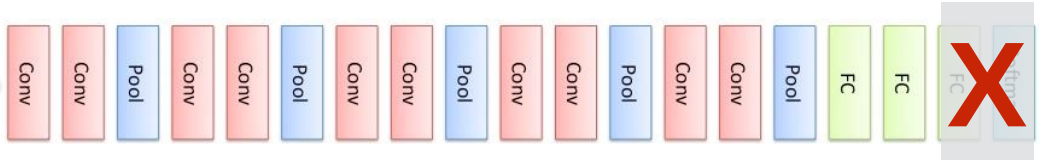


Image Captioning



0

Image Captioning

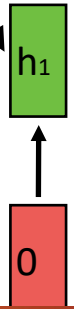
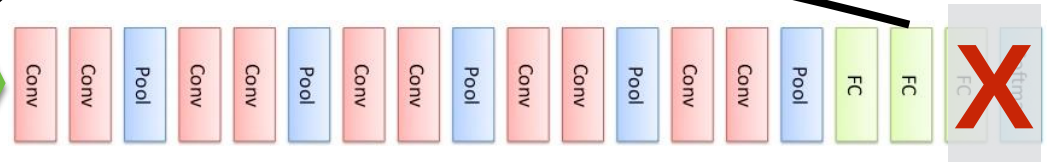


Image Captioning



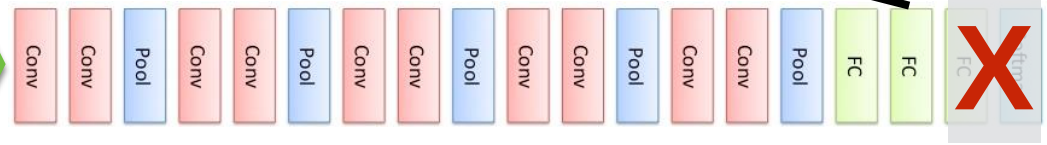
I

y_1

h_1

0

Image Captioning



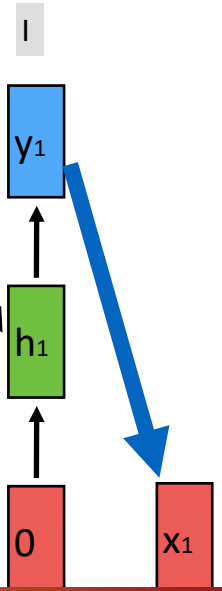
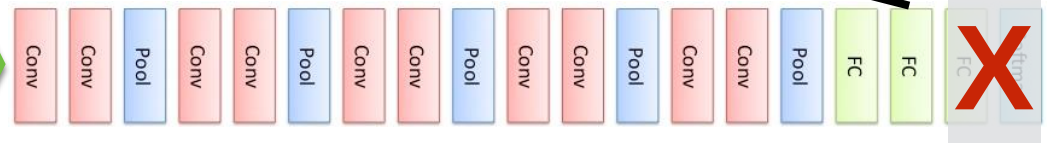


Image Captioning



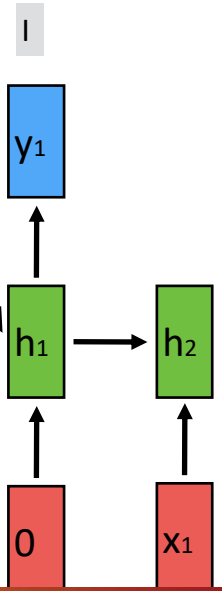
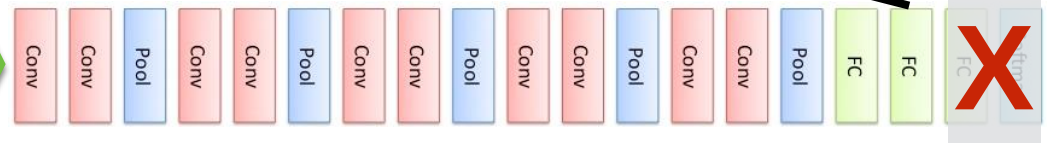


Image Captioning



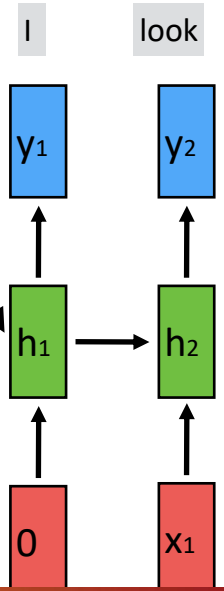
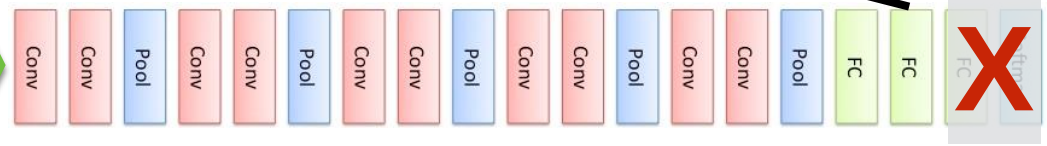


Image Captioning



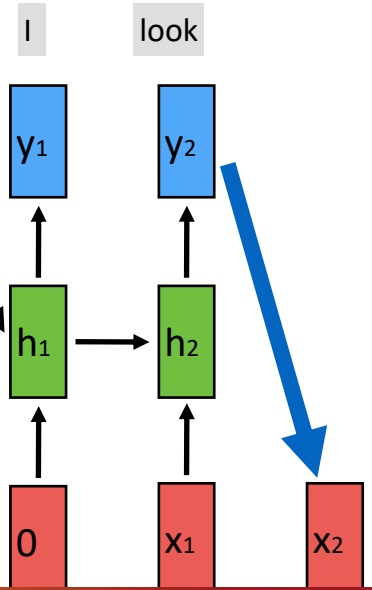
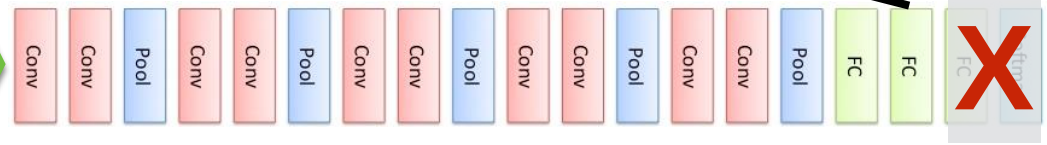


Image Captioning



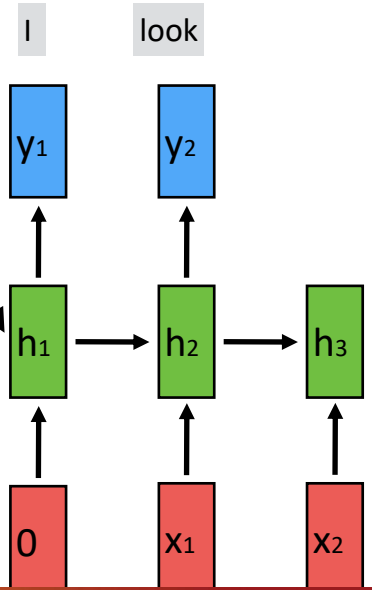
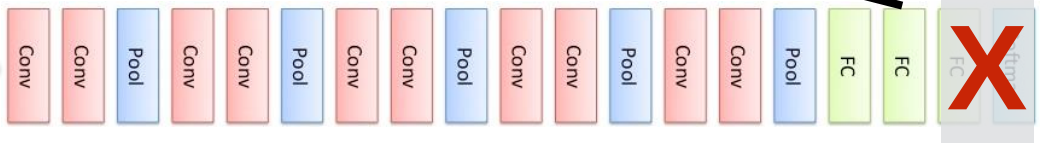


Image Captioning



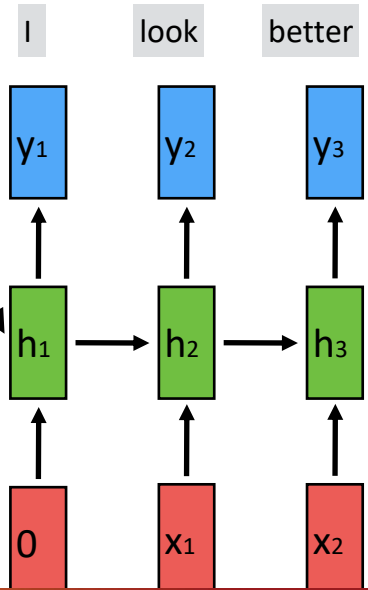
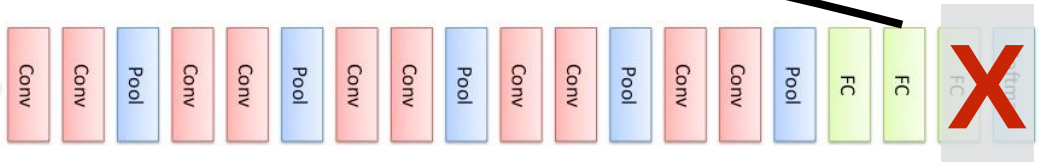


Image Captioning



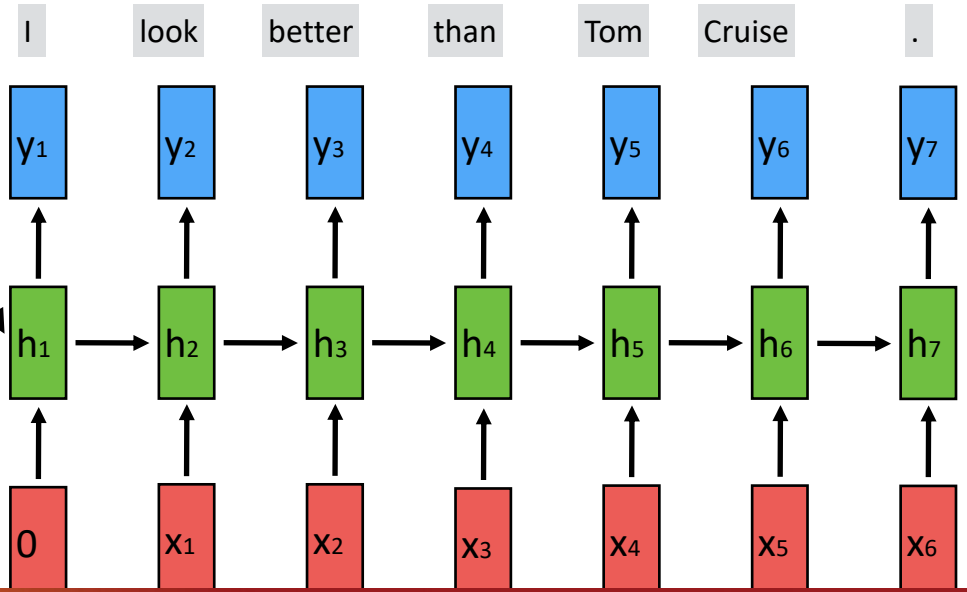


Image Captioning: Examples

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

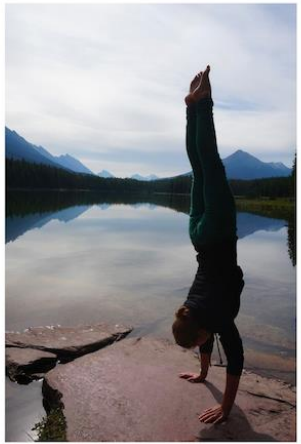
Two giraffes standing in a grassy field

A man riding a dirt bike on a dirt track

Image Captioning: Failures

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

Question Answering

Mary moved to the bathroom.

John went to the hallway.

Where is Mary? bathroom

https://allenai.github.io/bi-att-flow/demo/
Dialogue Learning With Human-in-the-loop, Jiwei et al.

Question Answering

Text

Mary moved to the bathroom.

John went to the hallway.

Question

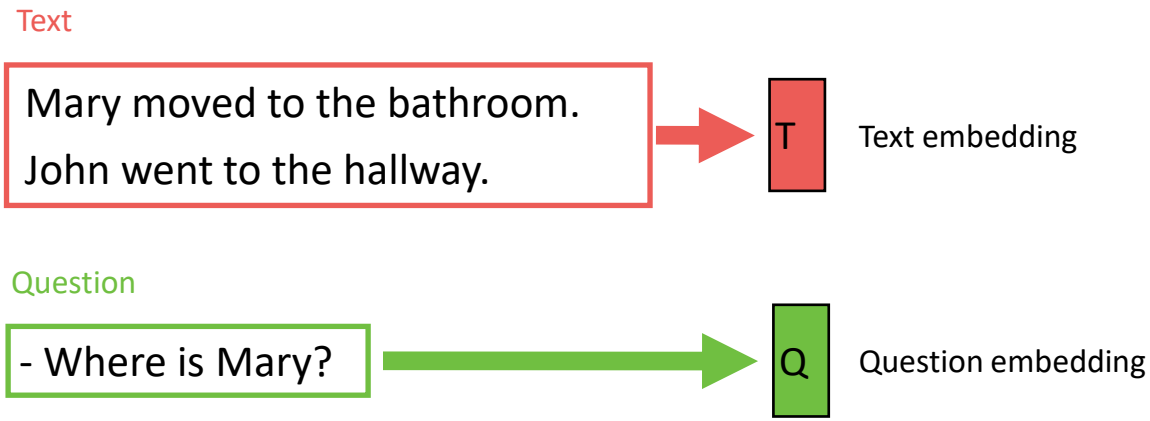
Where is Mary?

Answer

bathroom

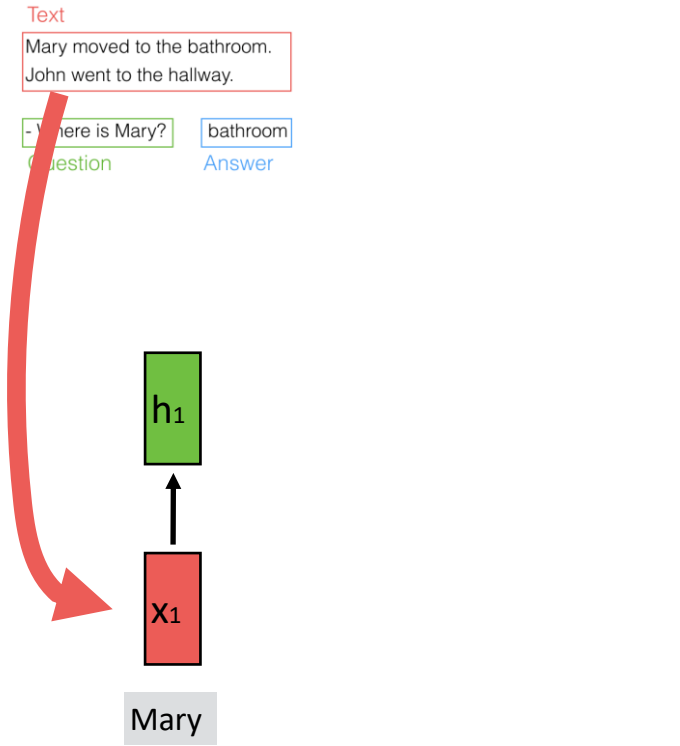
https://allenai.github.io/bi-att-flow/demo/
Dialogue Learning With Human-in-the-loop, Jiwei et al.

Question Answering

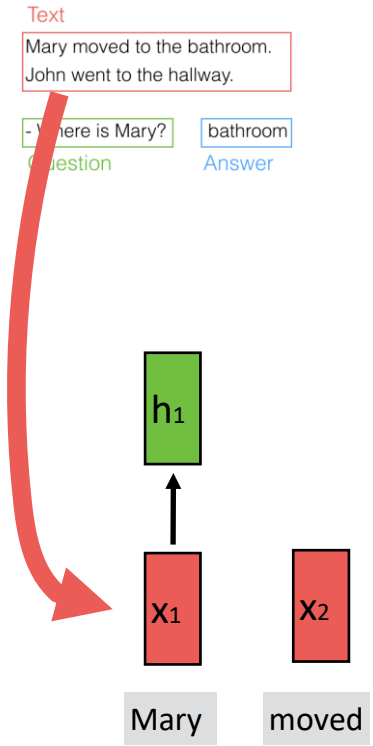


Question Answering

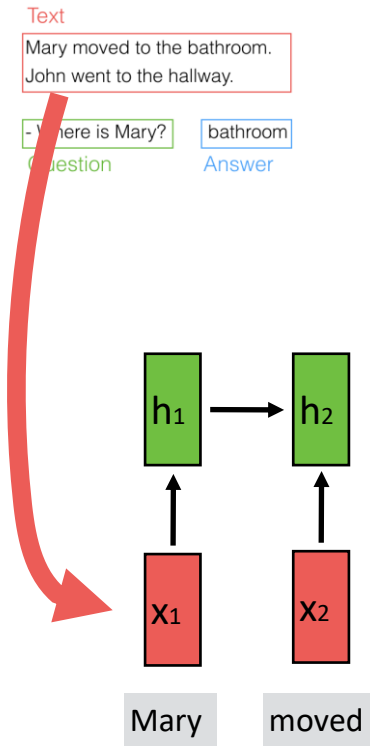
Question Answering



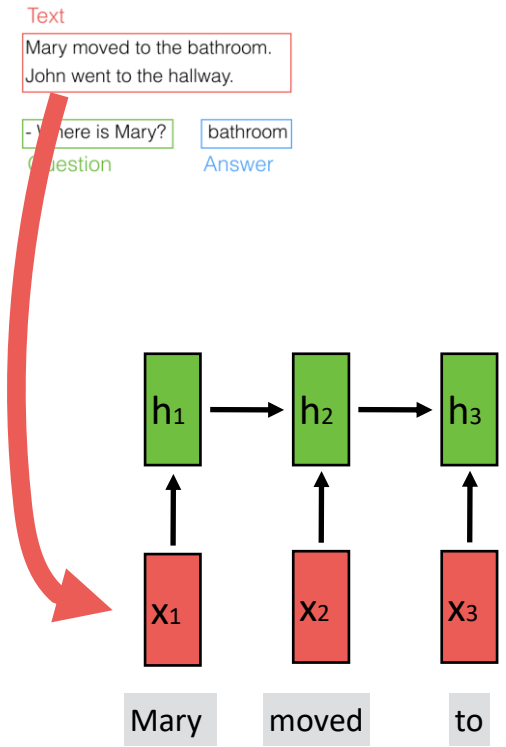
Question Answering



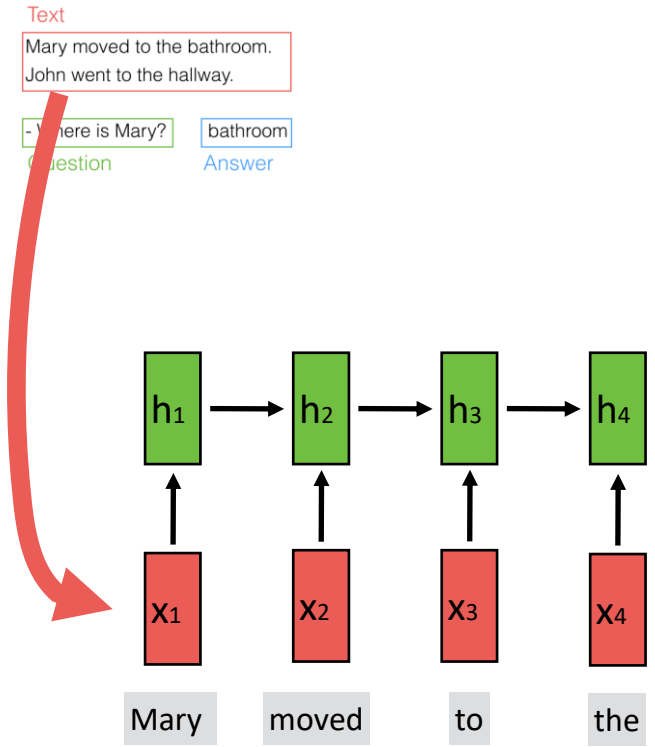
Question Answering



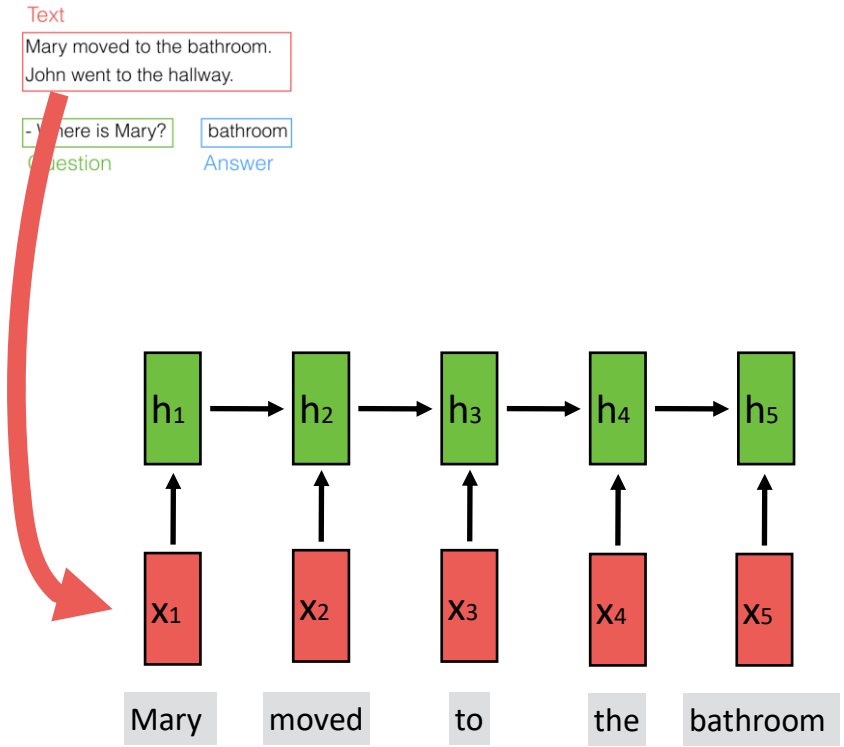
Question Answering



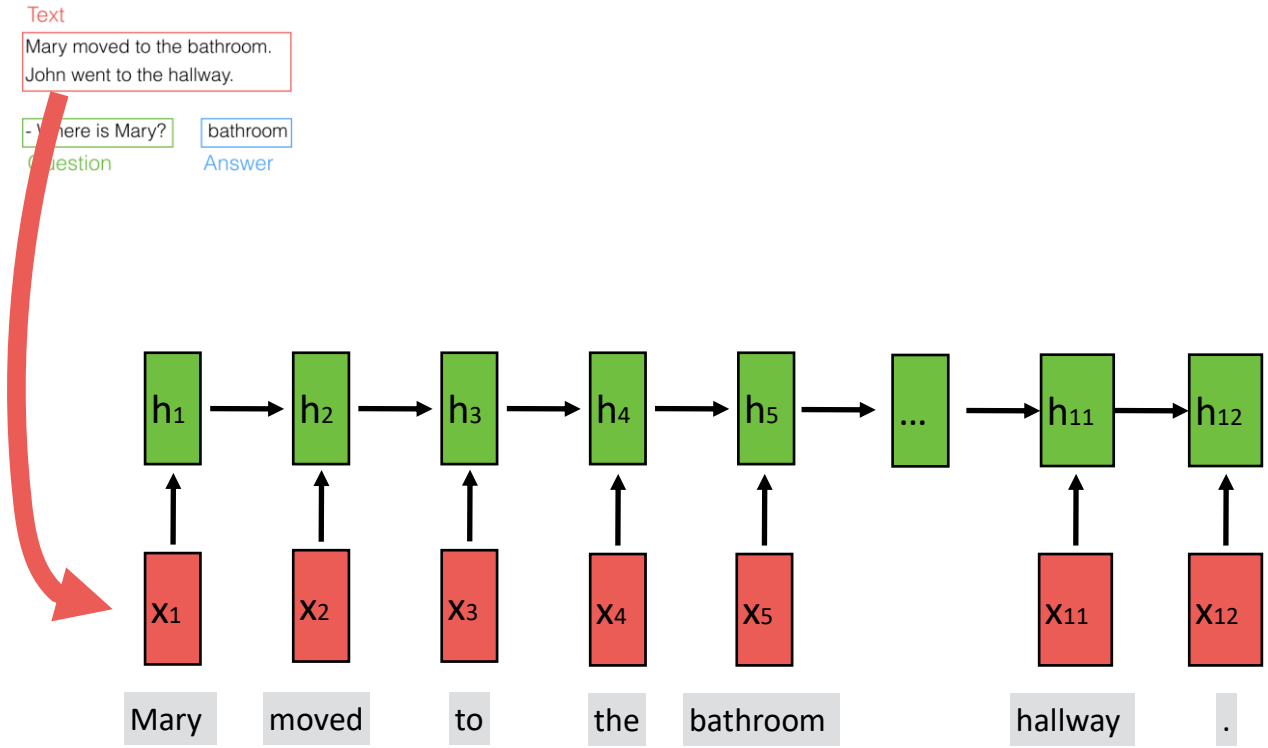
Question Answering



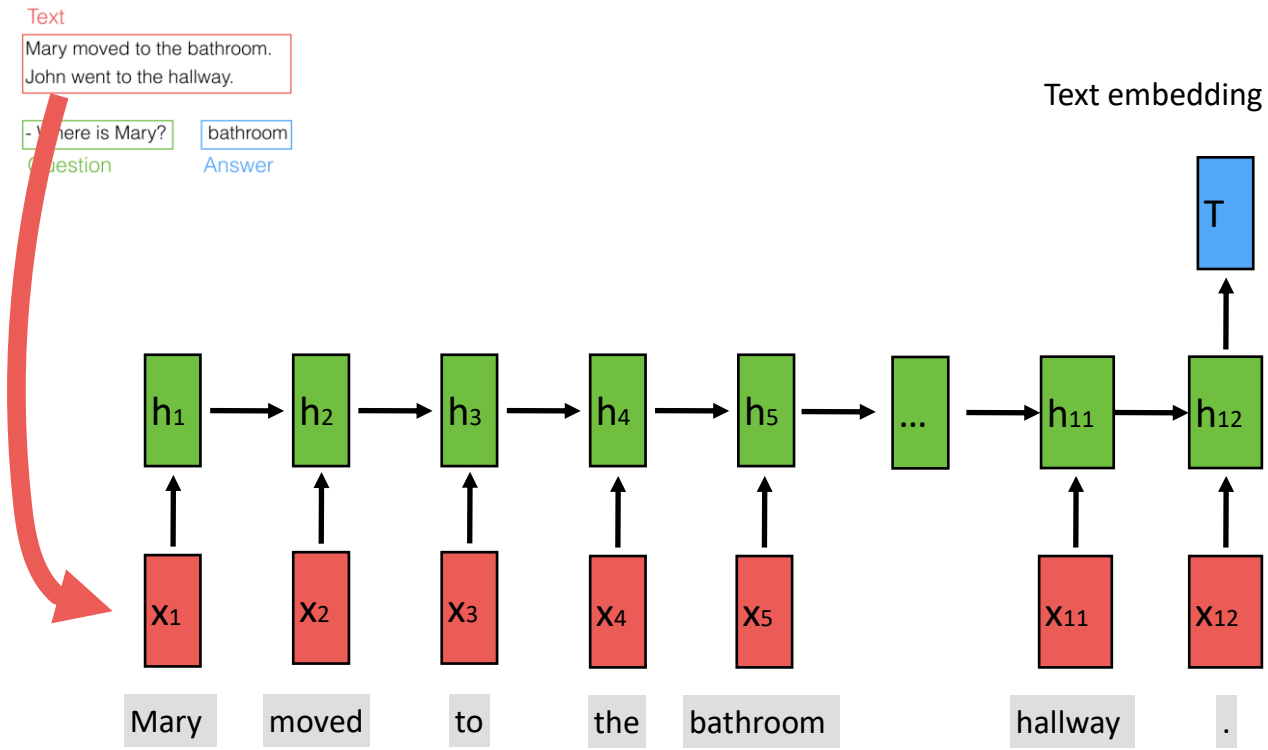
Question Answering



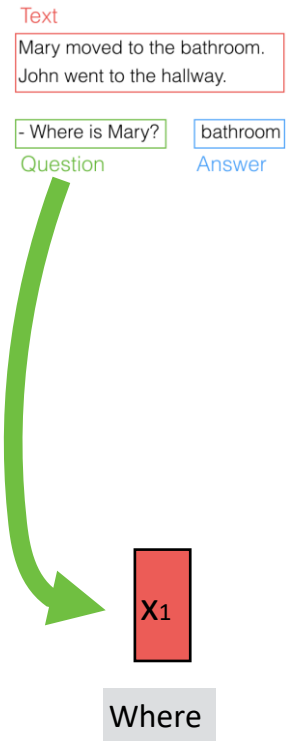
Question Answering



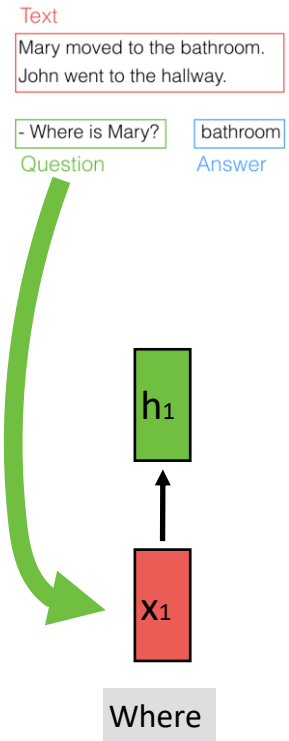
Question Answering



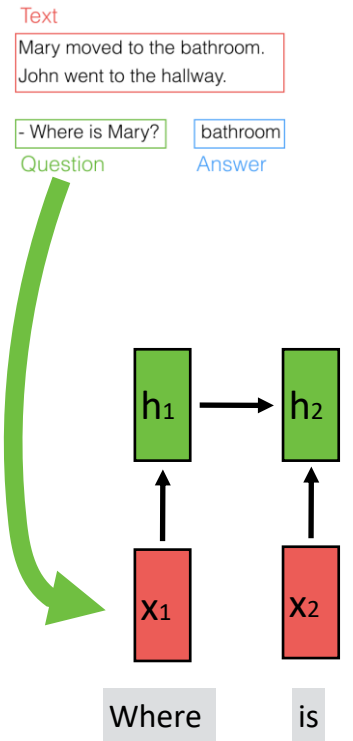
Question Answering



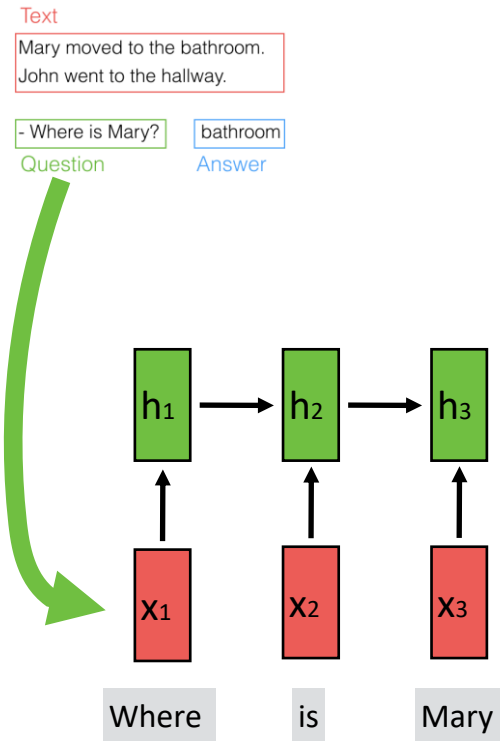
Question Answering



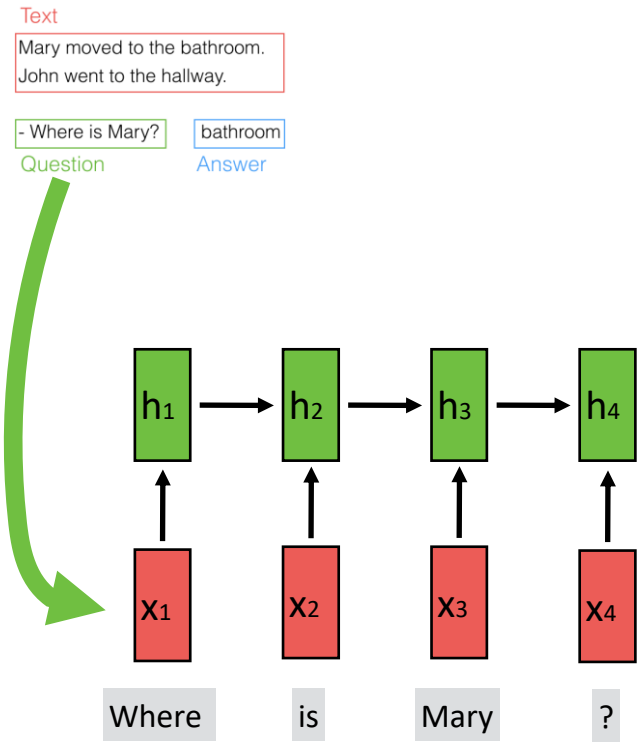
Question Answering



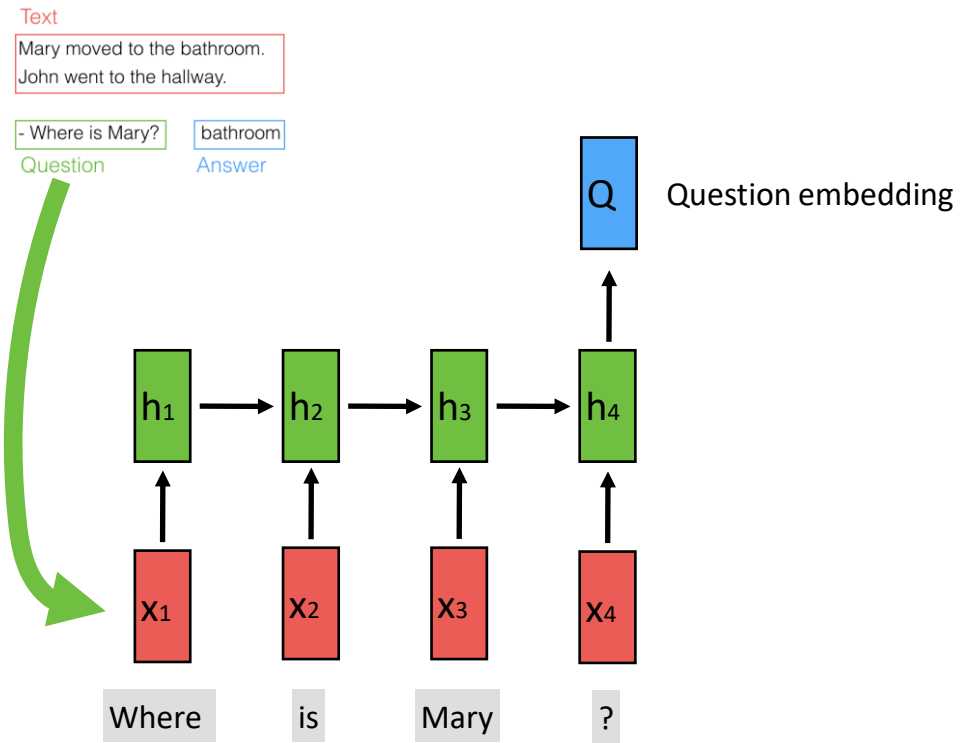
Question Answering



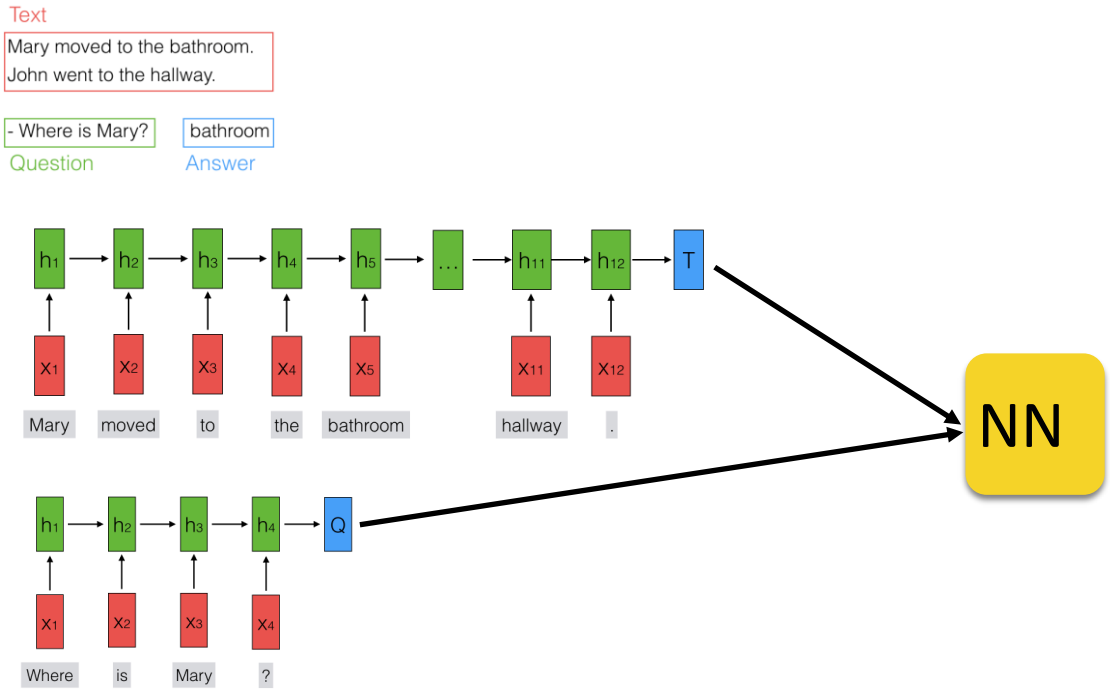
Question Answering



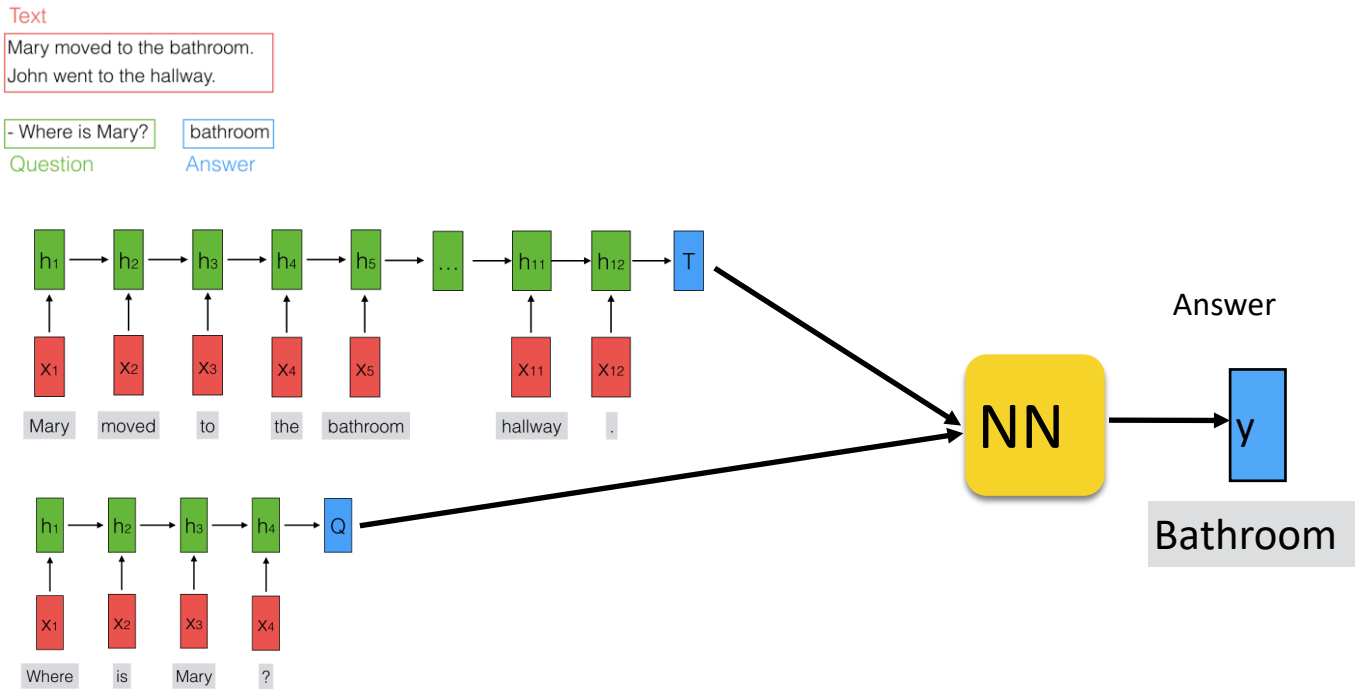
Question Answering



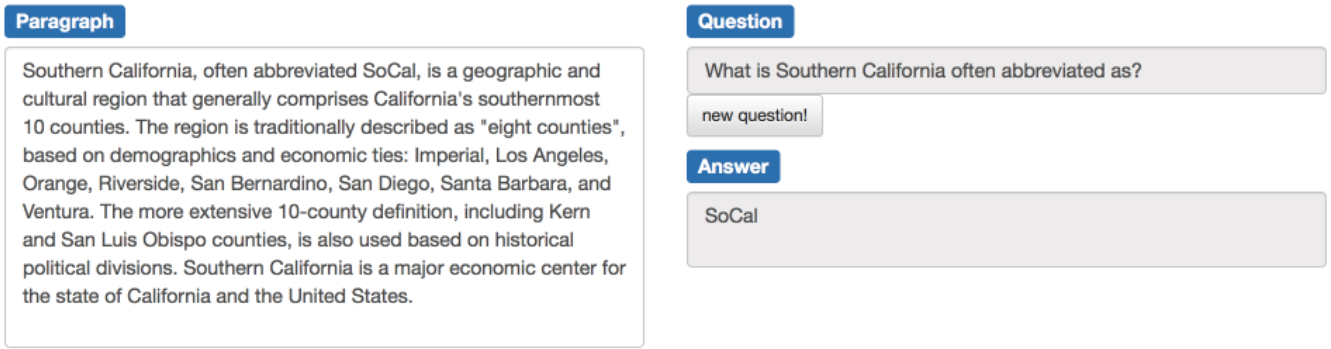
Question Answering



Question Answering



DEMO



- https://allenai.github.io/bi-att-flow/demo/

Visual Question Answering

What color are her eyes?
What is the mustache made of?

How many slices of pizza are there?
Is this a vegetarian pizza?

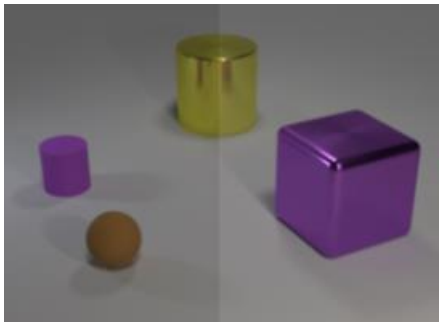
Is this person expecting company?
What is just under the tree?

Does it appear to be rainy?
Does this person have 20/20 vision?

VQA: Visual Question Answering, Agrawal et al
A simple neural network module for relational reasoning, Santoro et al

Visual Question Answering

Image



Question

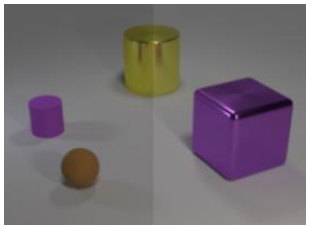
What is the size of the brown sphere?

Answer

Small

Visual Question Answering

Image

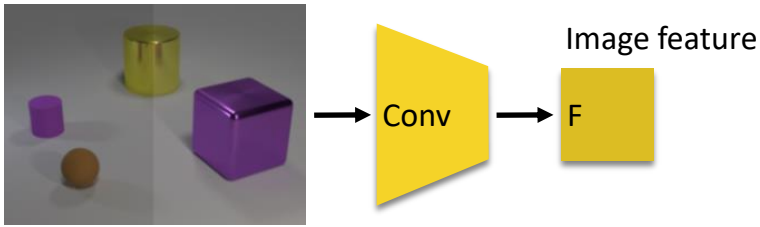


Question

What is the size of the brown sphere?

Visual Question Answering

Image

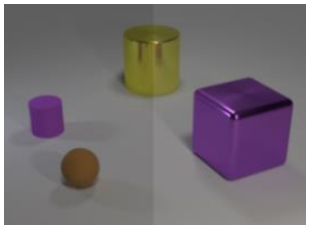


Question

What is the size of the brown sphere?

Visual Question Answering

Image



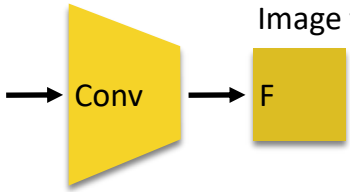
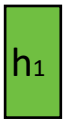


Image feature

Question

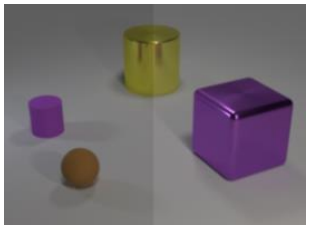
What is the size of the brown sphere?



What

Visual Question Answering

Image



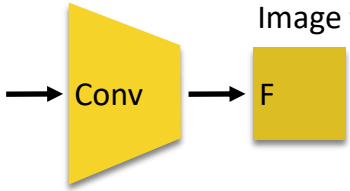
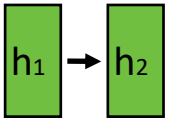


Image feature

Question

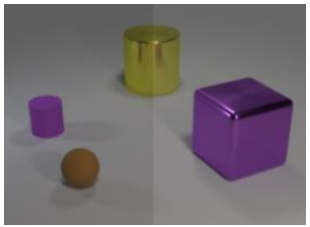
What is the size of the brown sphere?



What is

Visual Question Answering

Image



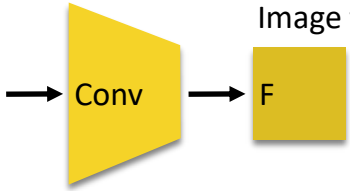
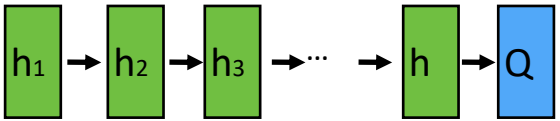


Image feature

Question

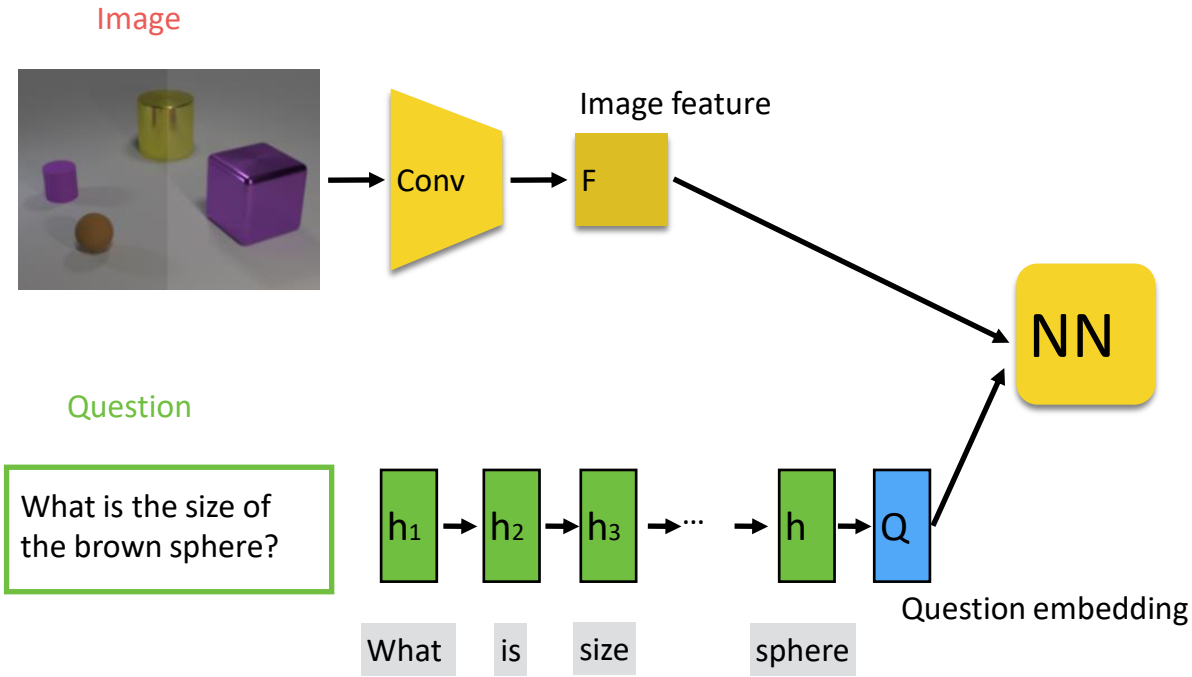
What is the size of the brown sphere?



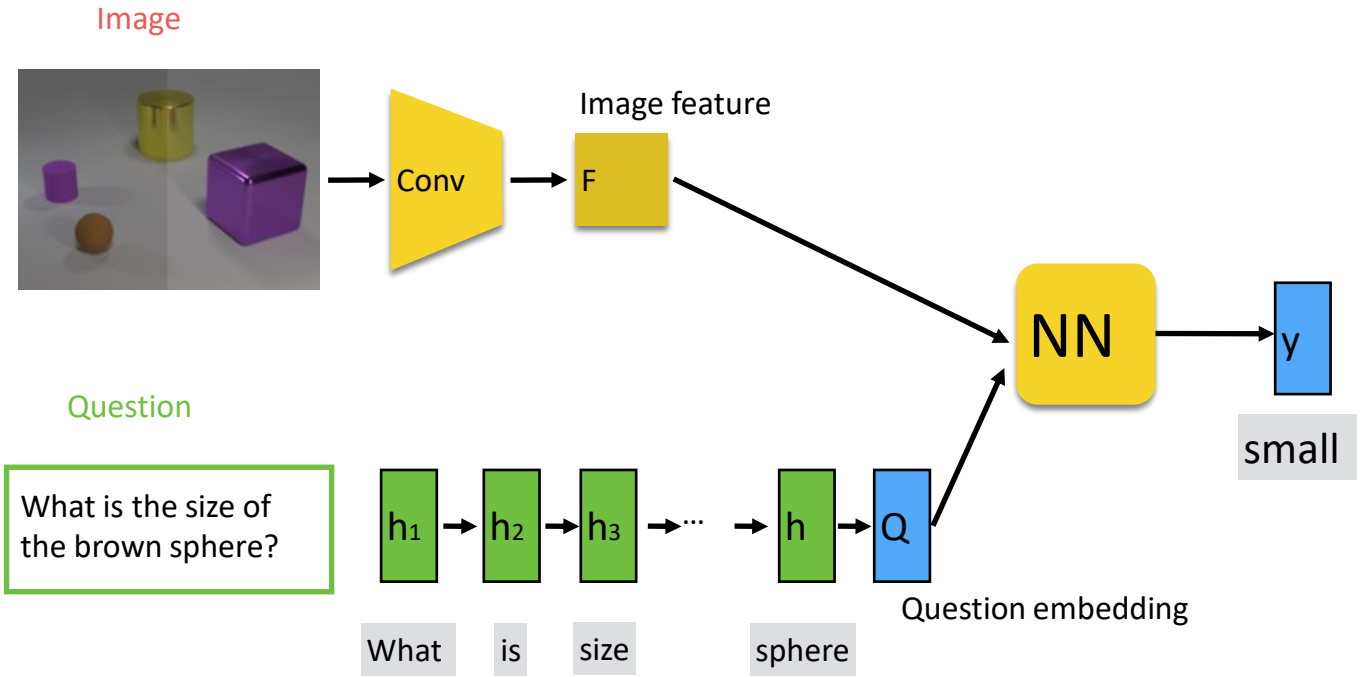
Question embedding

What is size sphere

Visual Question Answering



Visual Question Answering



RNN & LSTM Summary

- RNNs deal with various sequential data or dependencies.
- LSTM is go-to models (rather than vanilla RNNs)
- RNNs are still under development (relative to CNNs)