



max planck institut  
informatik



UNIVERSITÄT  
DES  
SAARLANDES

# High Level Computer Vision

## Deep Learning for Computer Vision Part 4

Bernt Schiele - [schiele@mpi-inf.mpg.de](mailto:schiele@mpi-inf.mpg.de)

Mario Fritz - [mfritz@mpi-inf.mpg.de](mailto:mfritz@mpi-inf.mpg.de)

<https://www.mpi-inf.mpg.de/hlcv>

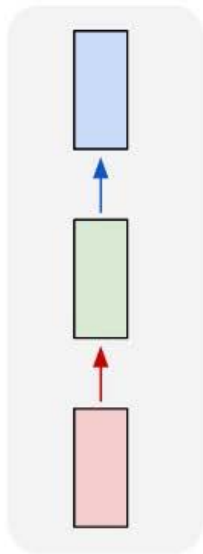
# Overview

---

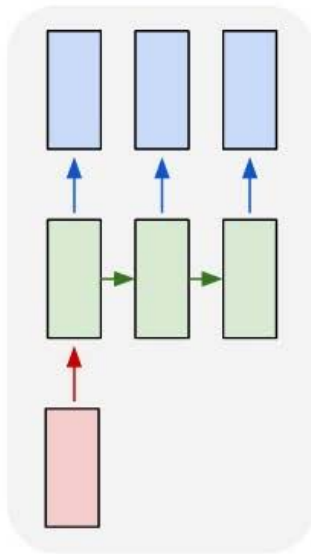
- Recurrent Neural Networks
  - ▶ motivation for recurrent neural networks
  - ▶ a particularly successful RNN: Long Short Term Memory (LSTM)
  - ▶ slide credit to Andrej Karpathy, Jeff Donahue and Marcus Rohrbach
  
- Yann LeCun...
  - ▶ What's Wrong With Deep Learning (keynote June 2015)
  - ▶ slide credit to Yann LeCun (and Xiaogang Wang)

# Recurrent Networks offer a lot of flexibility:

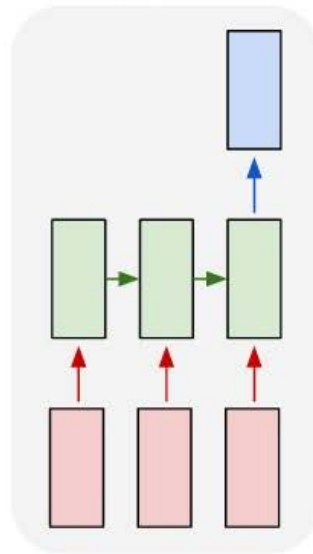
one to one



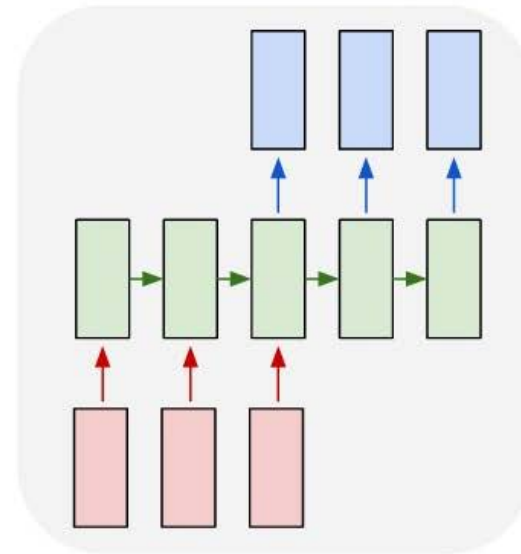
one to many



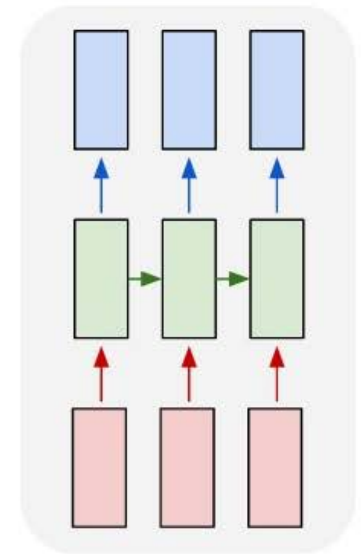
many to one



many to many

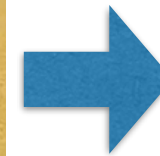


many to many



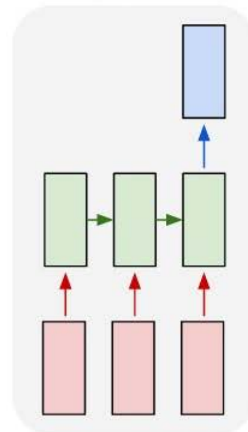
# Sequences in Vision

Sequences in the input...  
(many-to-one)



- Running
- Jumping
- Dancing
- Fighting
- Eating

many to one

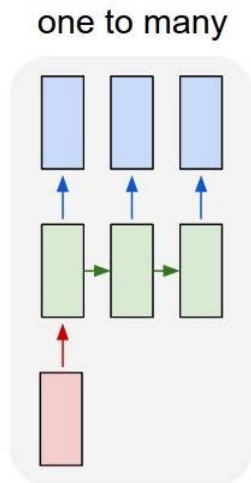


# Sequences in Vision

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

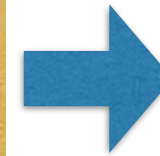


*A happy brown dog.*

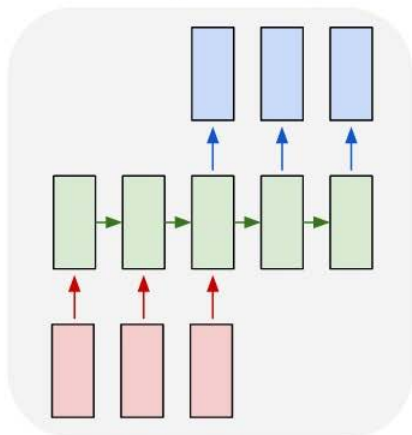


# Sequences in Vision

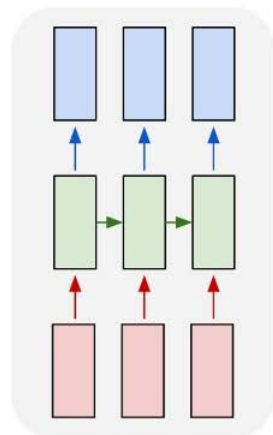
Sequences everywhere!  
(many-to-many)



many to many

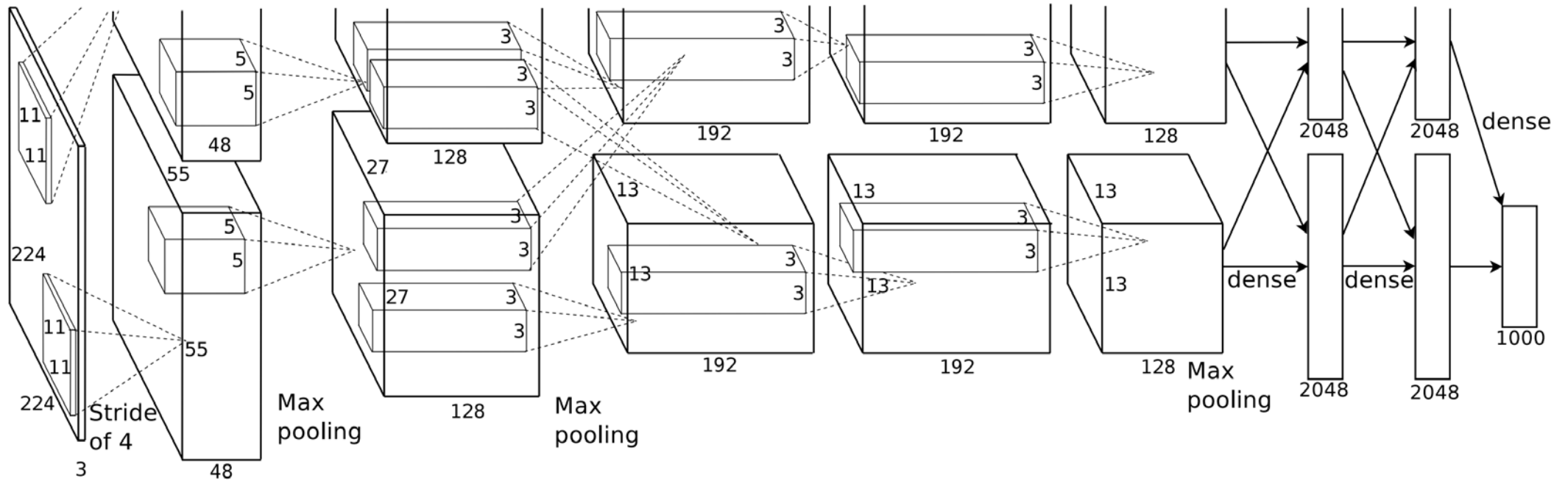


many to many



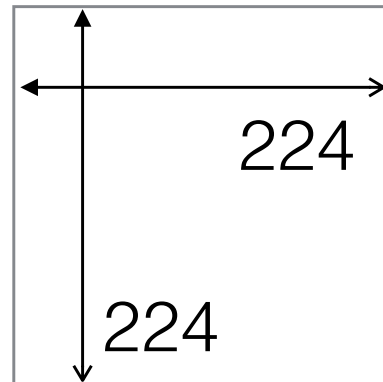
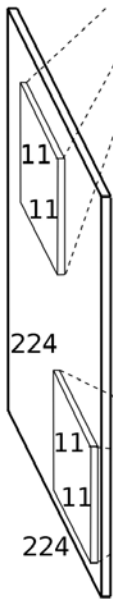
*A dog jumps over a hurdle.*

# ConvNets



# Problem #1

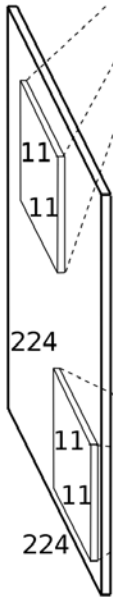
fixed-size, static input



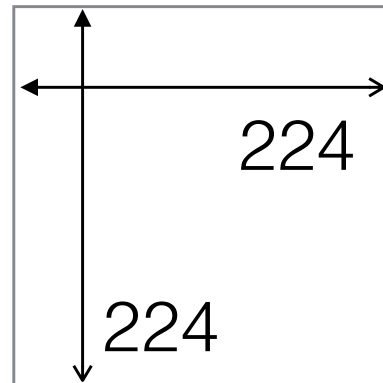


# Problem #1

fixed-size, static input



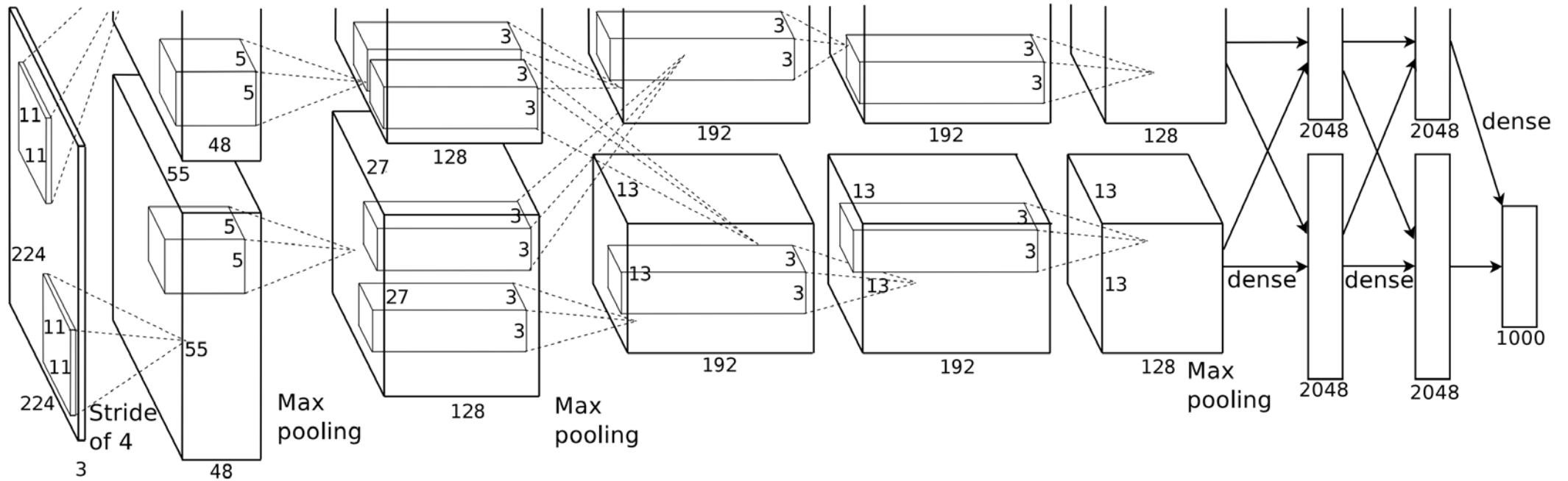
3



???



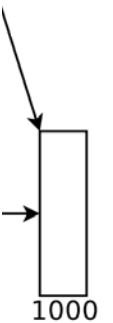
# Problem #2



# Problem #2

output is a single choice from a fixed list of options

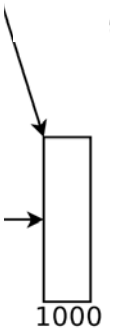
- cat
- dog
- horse
- fish
- snake



# Problem #2

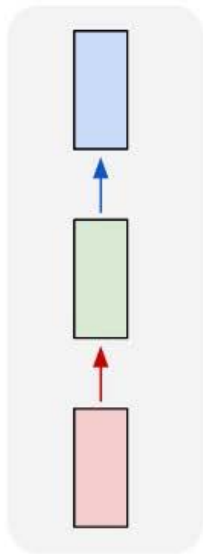
output is a single choice from a fixed list of options

- a happy brown dog
- a big brown dog
- a happy red dog
- a big red dog
- ...

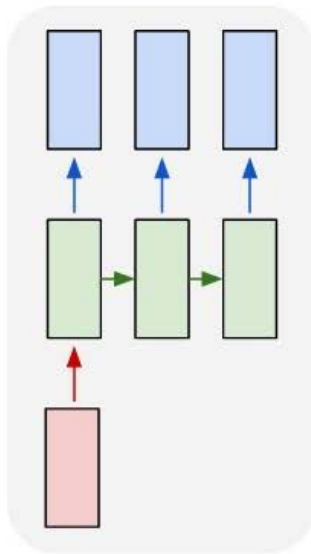


# Recurrent Networks offer a lot of flexibility:

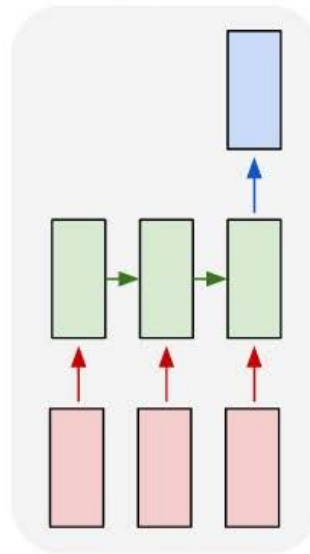
one to one



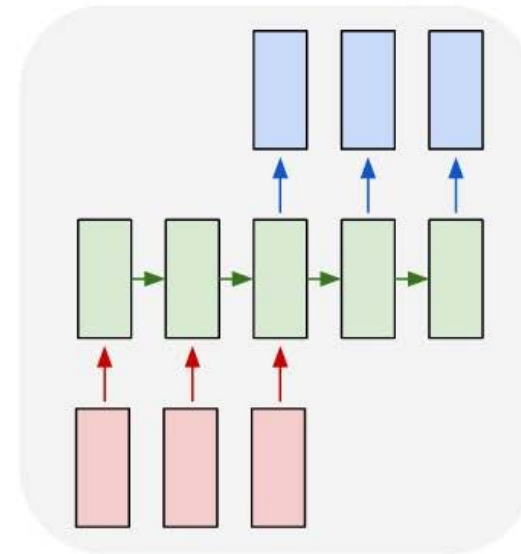
one to many



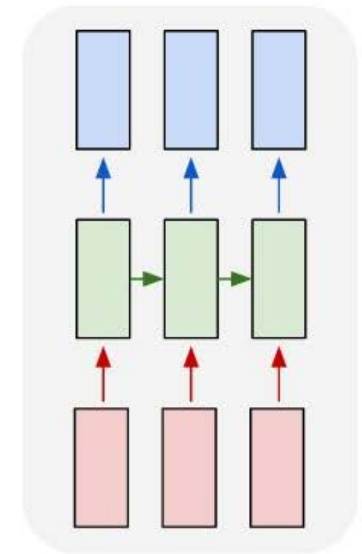
many to one



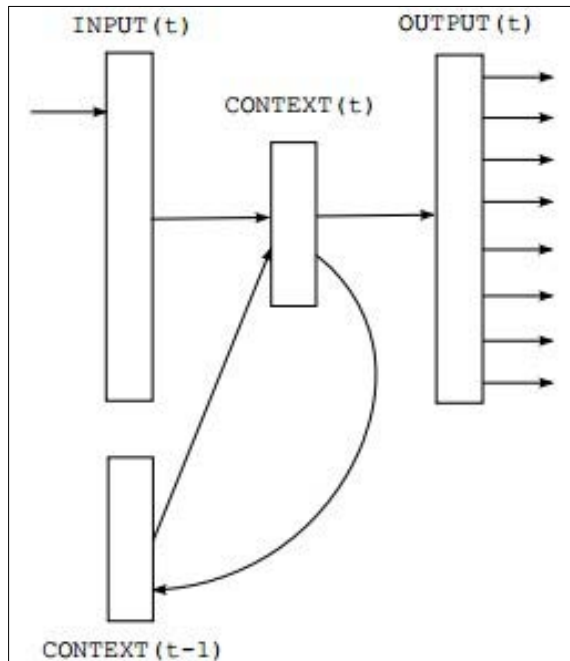
many to many



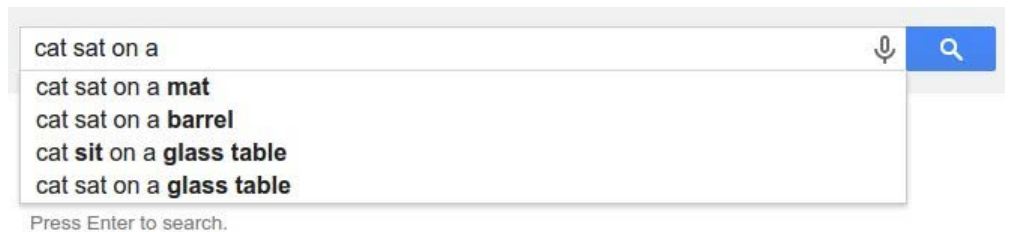
many to many



# Language Models



Word-level language model. Similar to:



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

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

We want to train a **language model**:

$P(\text{next word} \mid \text{previous words})$

i.e. want these to be high:

$P(\text{cat} \mid [ \langle S \rangle ])$

$P(\text{sat} \mid [ \langle S \rangle, \text{cat} ])$

$P(\text{on} \mid [ \langle S \rangle, \text{cat}, \text{sat} ])$

$P(\text{mat} \mid [ \langle S \rangle, \text{cat}, \text{sat}, \text{on} ])$

$P(\langle E \rangle \mid [ \langle S \rangle, \text{cat}, \text{sat}, \text{on}, \text{mat} ])$

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

We want to train a **language model**:

$P(\text{next word} \mid \text{previous words})$

First, suppose we had only a finite, 1-word history:  
i.e. want these to be high:

$P(\text{cat} \mid \langle S \rangle)$

$P(\text{sat} \mid \text{cat})$

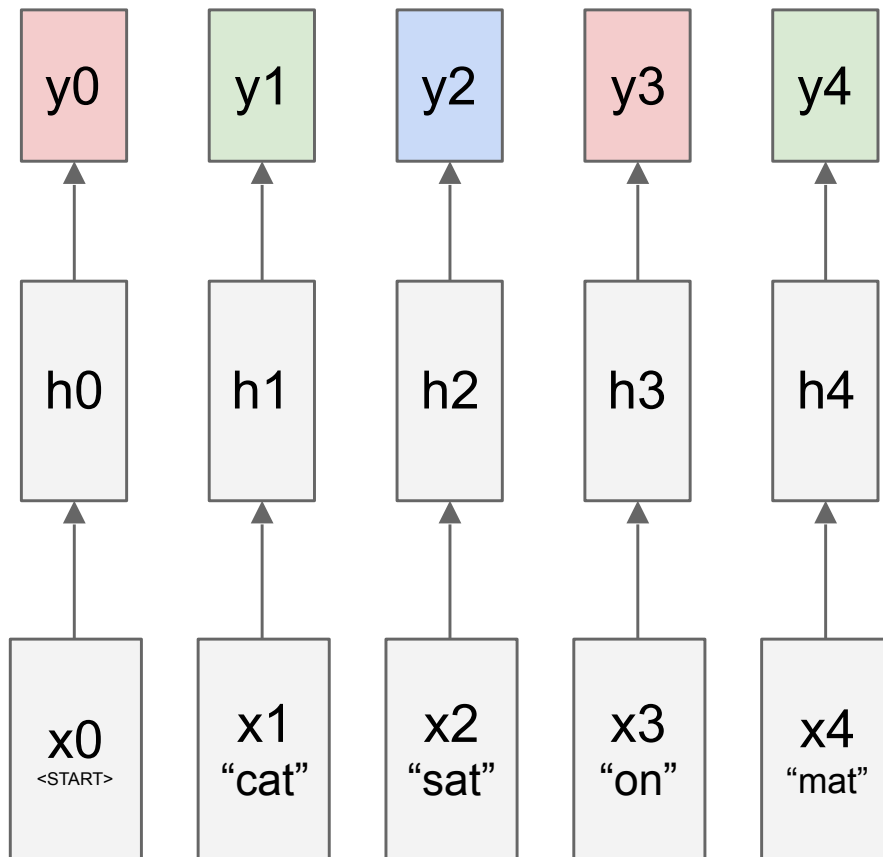
$P(\text{on} \mid \text{sat})$

$P(\text{mat} \mid \text{on})$

$P(\langle E \rangle \mid \text{mat})$

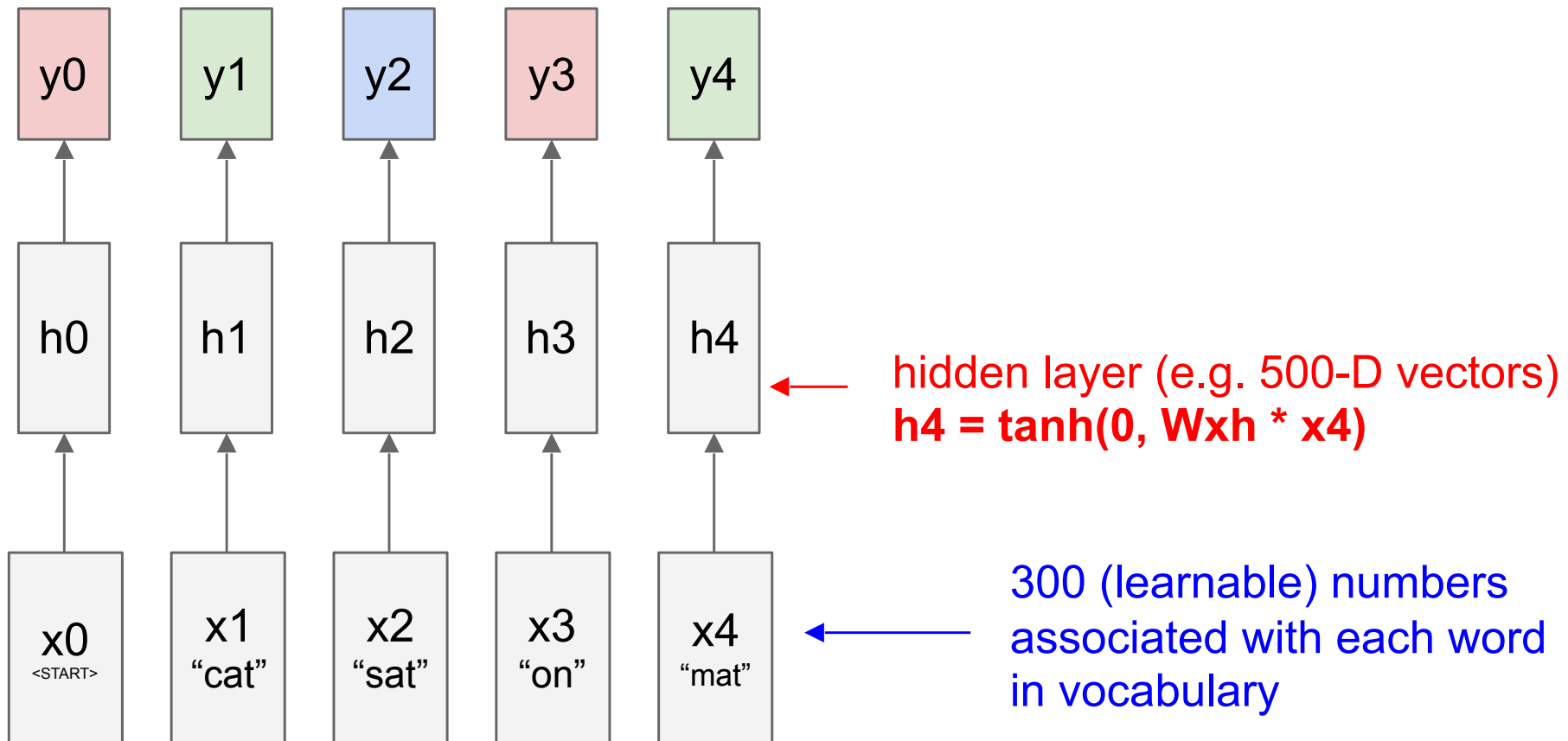


“cat sat on mat”

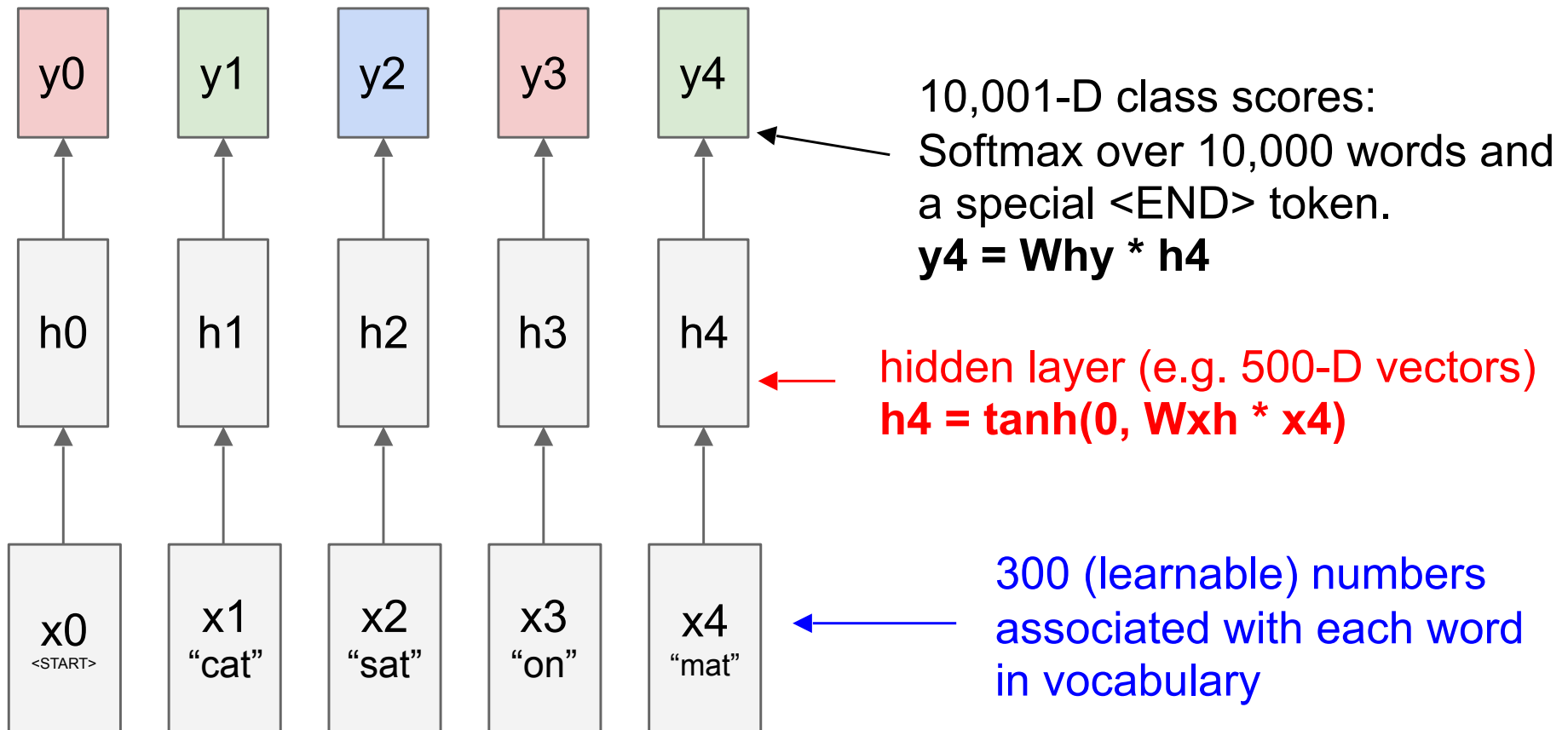


← 300 (learnable) numbers associated with each word in vocabulary

“cat sat on mat”

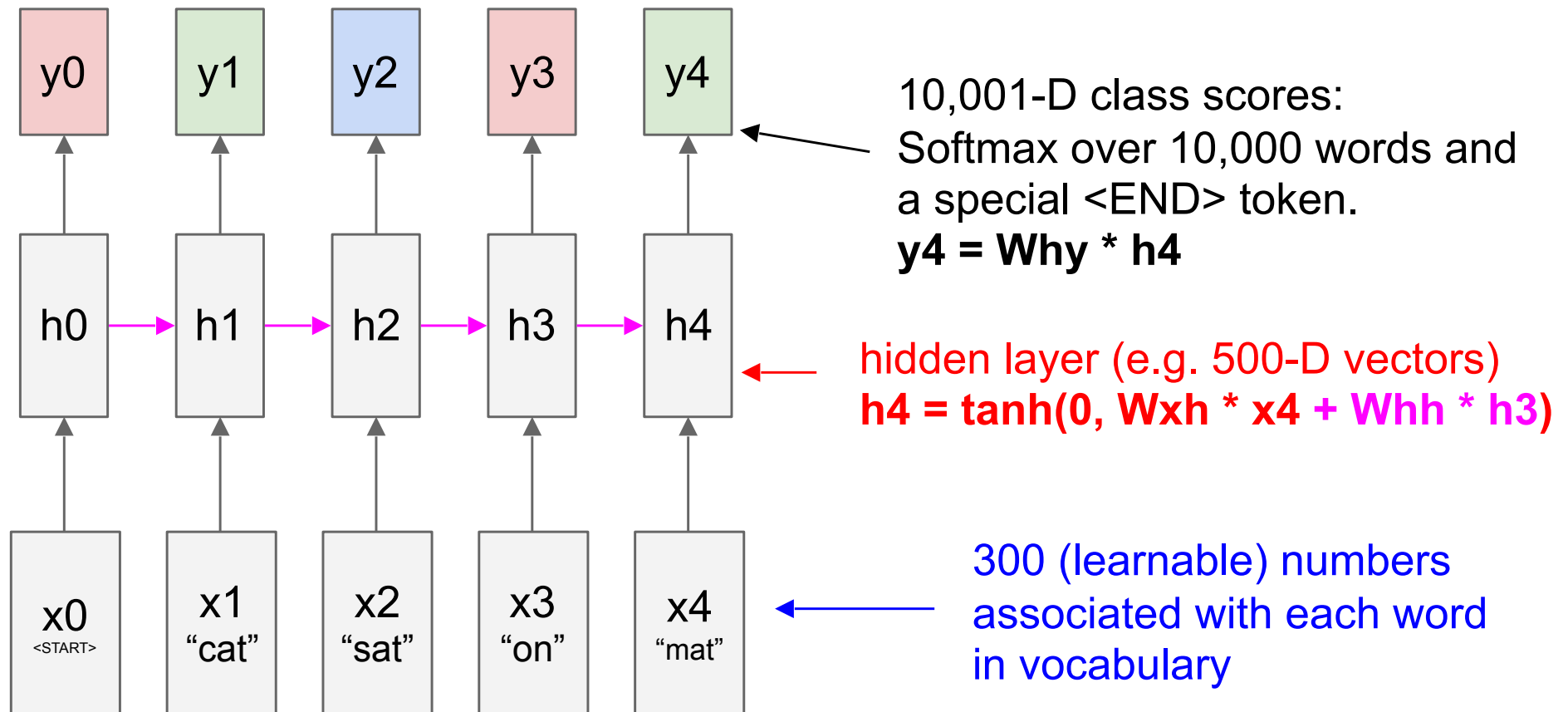


“cat sat on mat”



## Recurrent Neural Network:

“cat sat on mat”



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

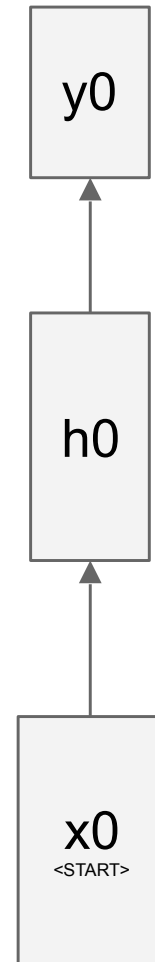
$P(\text{next word} \mid \text{previous words})$



x0  
<START>

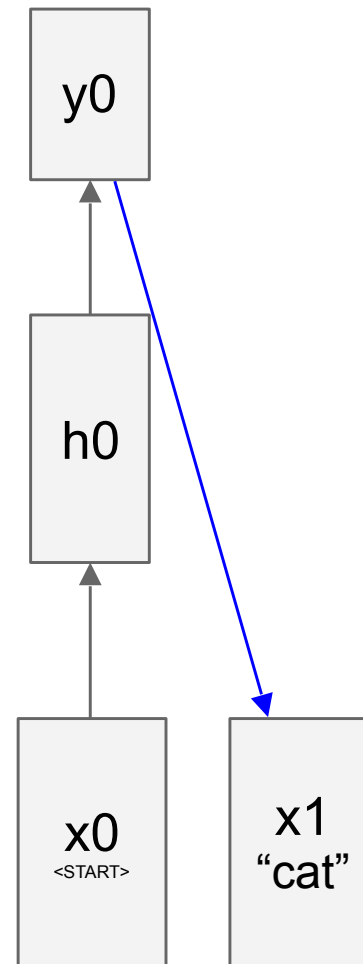
Training this on a lot of sentences would give us a language model. A way to predict

$P(\text{next word} \mid \text{previous words})$



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

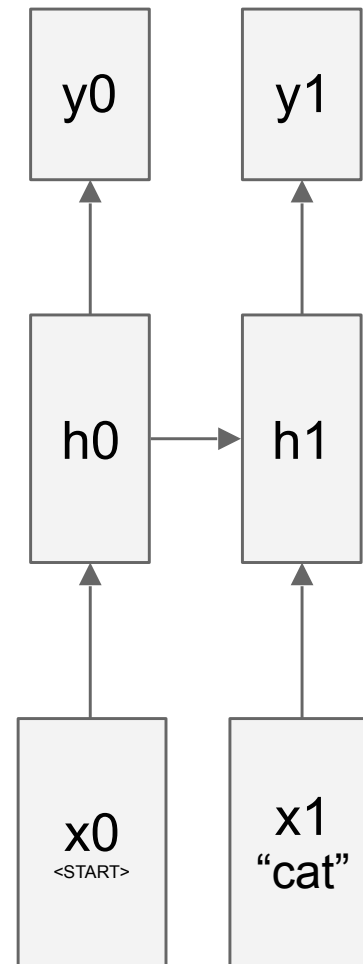
$P(\text{next word} \mid \text{previous words})$



sample!

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

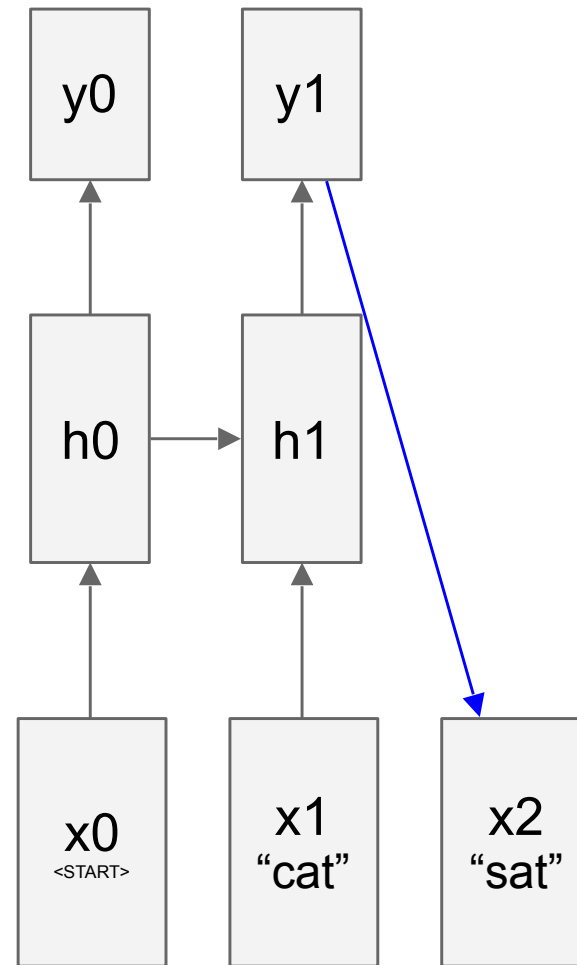
$P(\text{next word} \mid \text{previous words})$





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

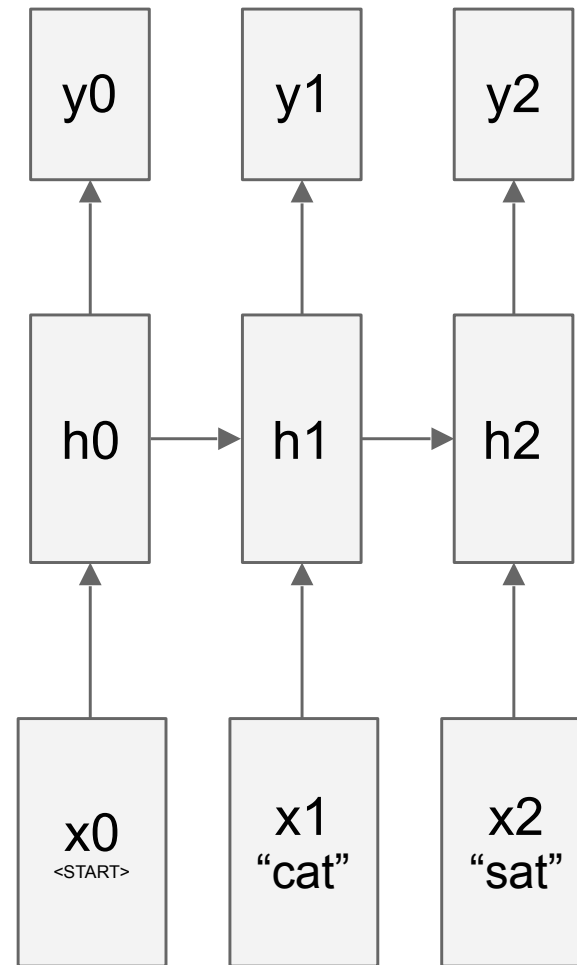
$P(\text{next word} \mid \text{previous words})$



sample!

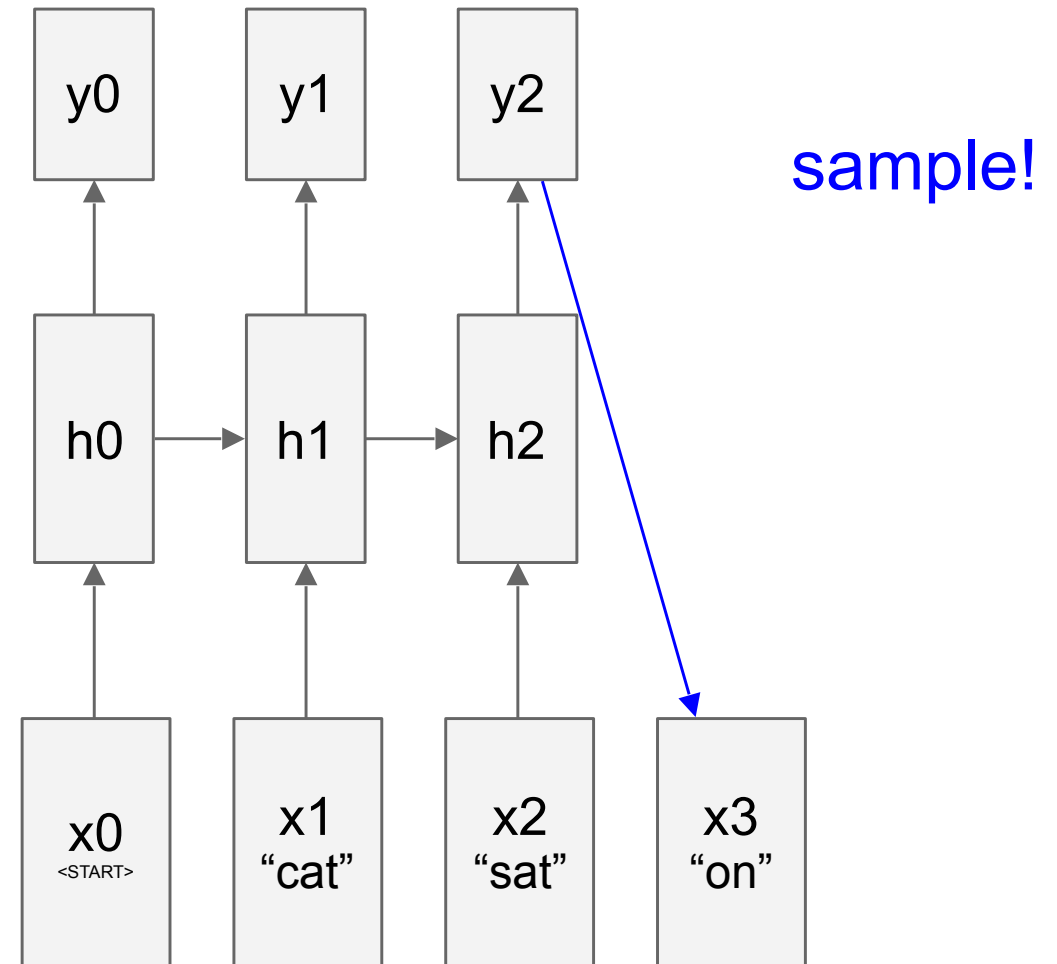
Training this on a lot of sentences would give us a language model. A way to predict

$P(\text{next word} \mid \text{previous words})$



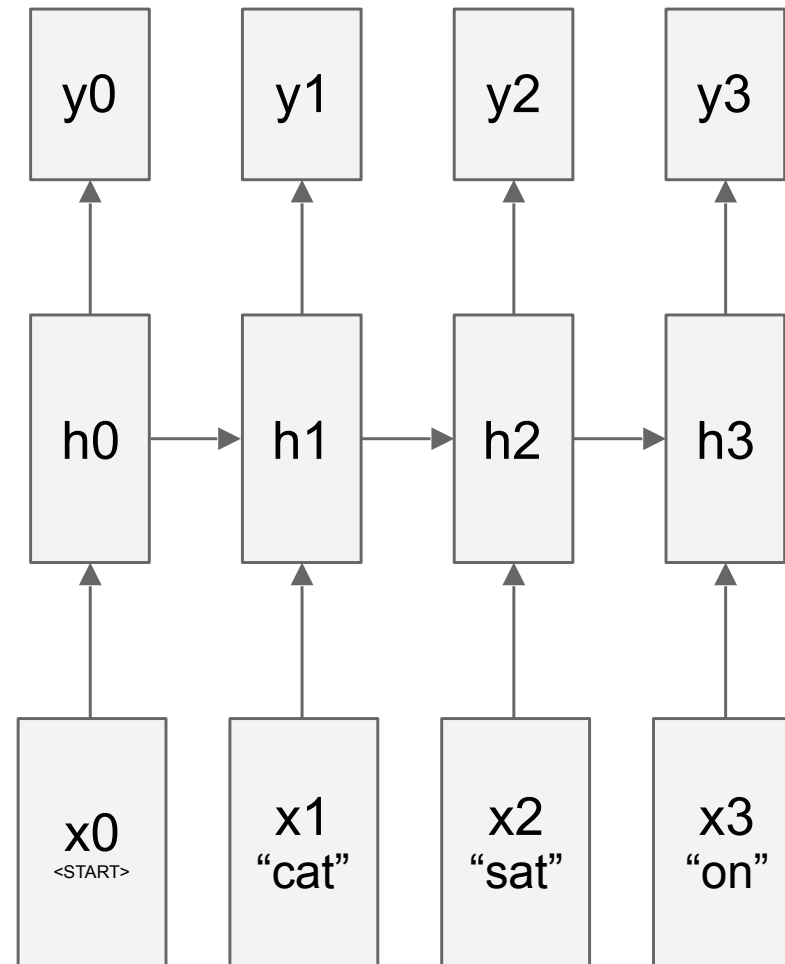
Training this on a lot of sentences would give us a language model. A way to predict

$P(\text{next word} \mid \text{previous words})$



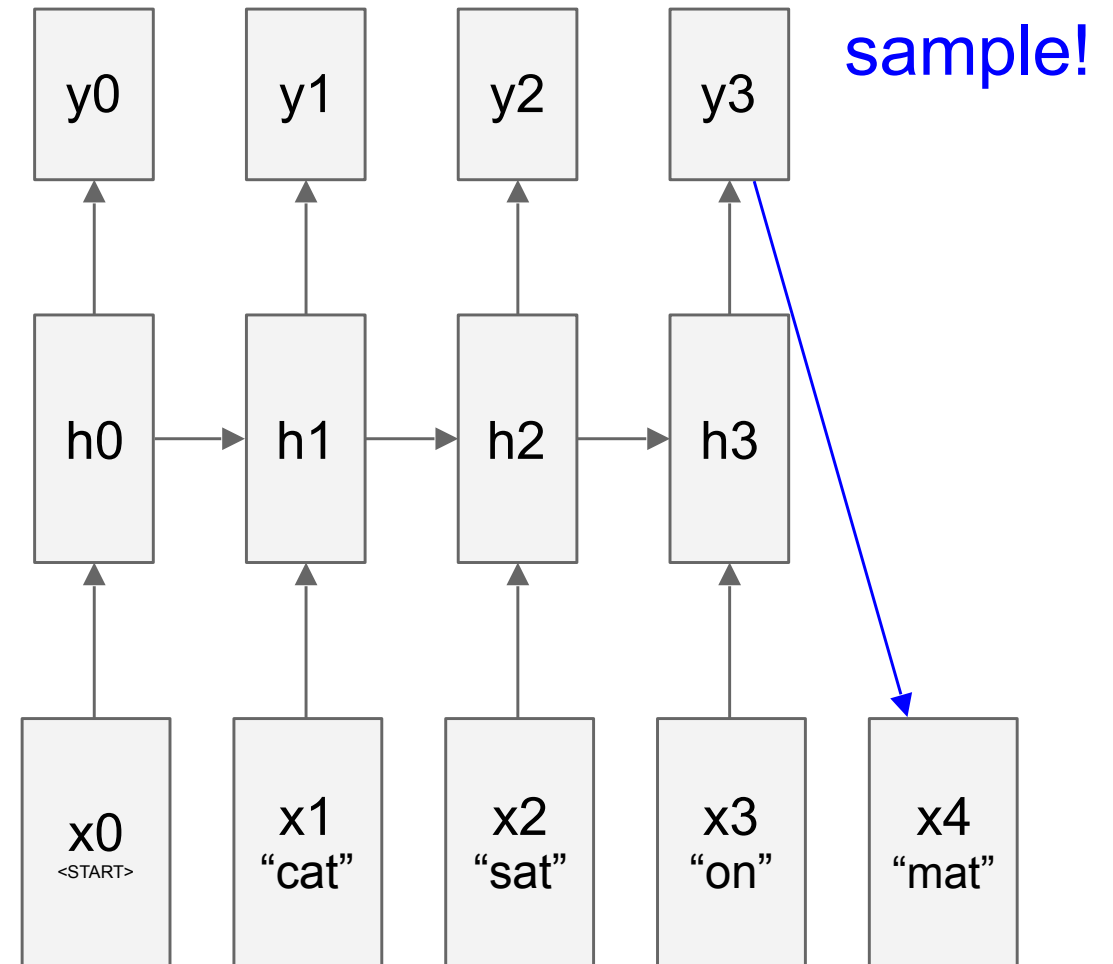
Training this on a lot of sentences would give us a language model. A way to predict

$P(\text{next word} \mid \text{previous words})$



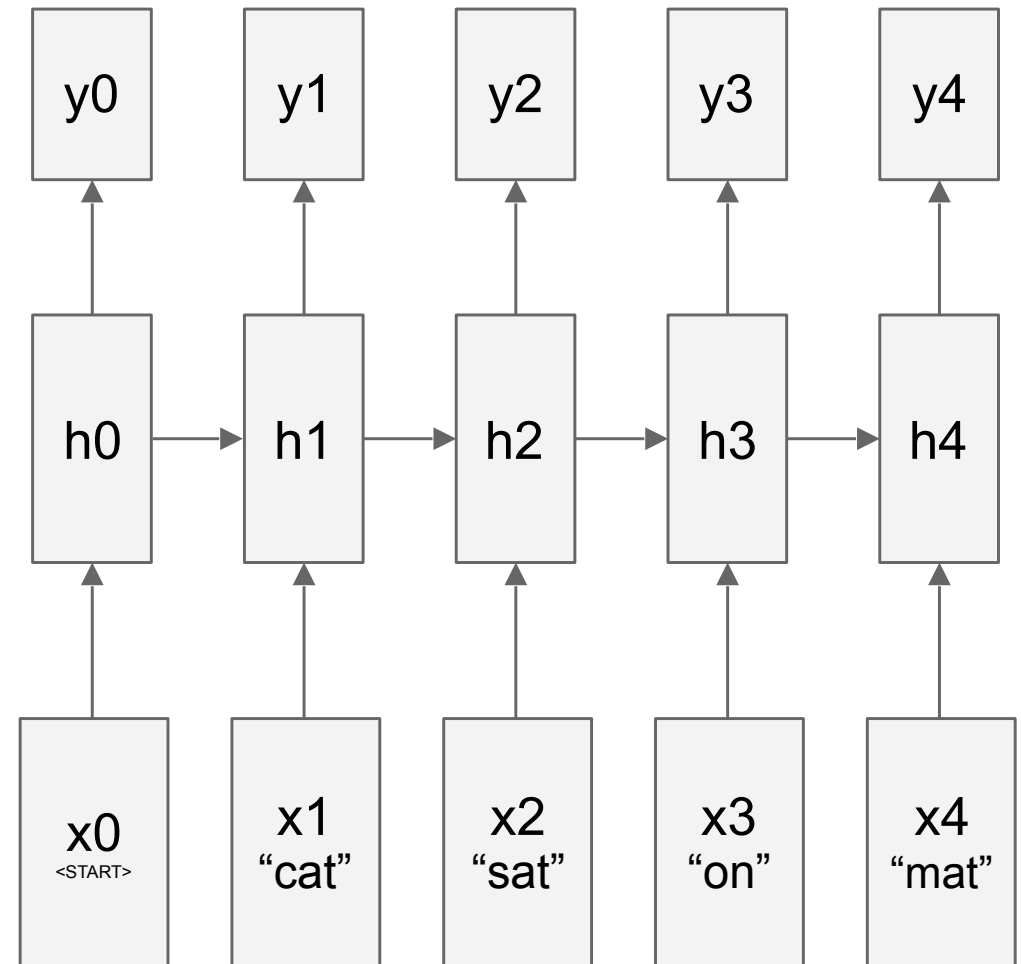
Training this on a lot of sentences would give us a language model. A way to predict

$P(\text{next word} \mid \text{previous words})$



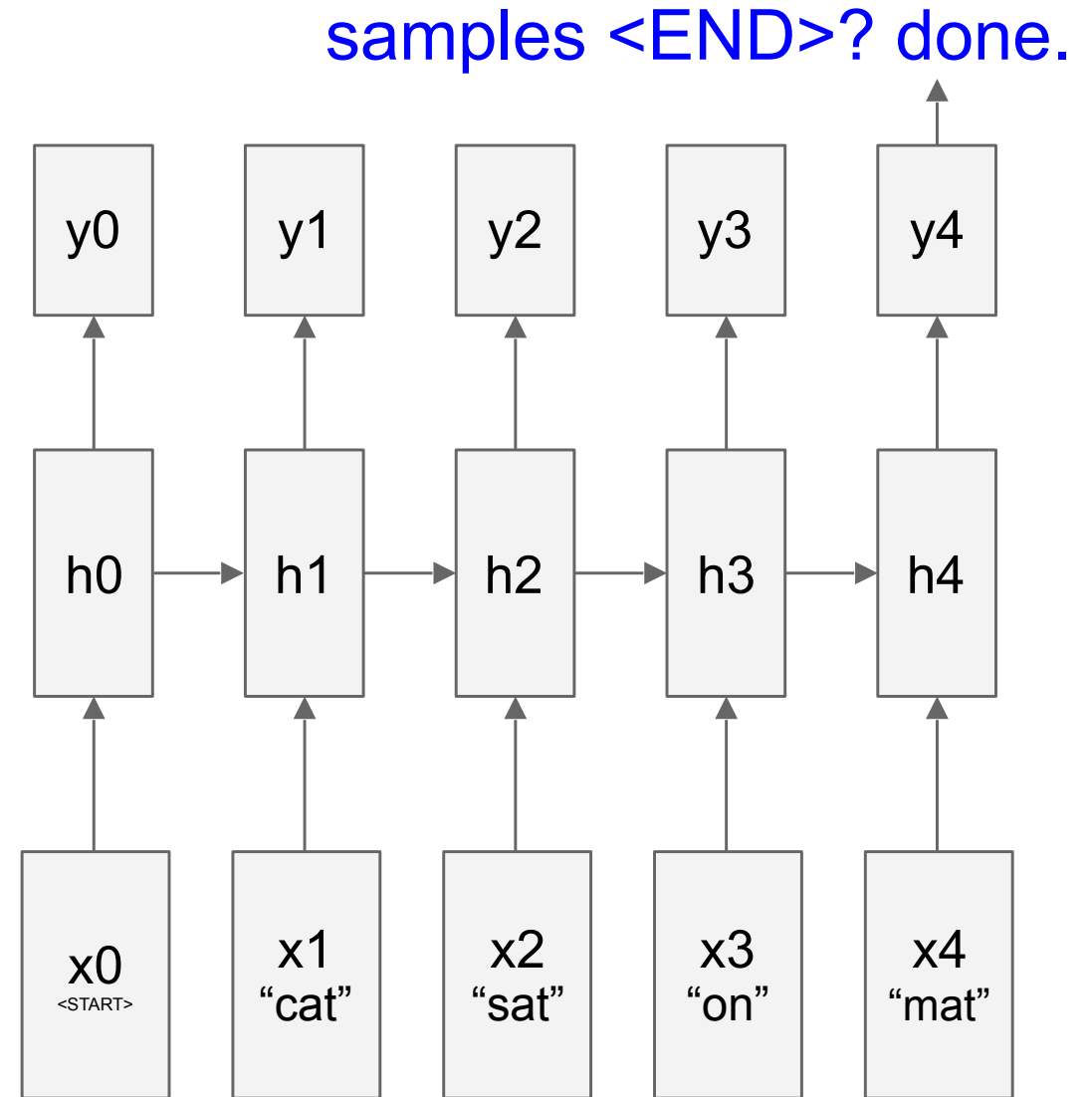
Training this on a lot of sentences would give us a language model. A way to predict

$P(\text{next word} \mid \text{previous words})$



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

$P(\text{next word} \mid \text{previous words})$

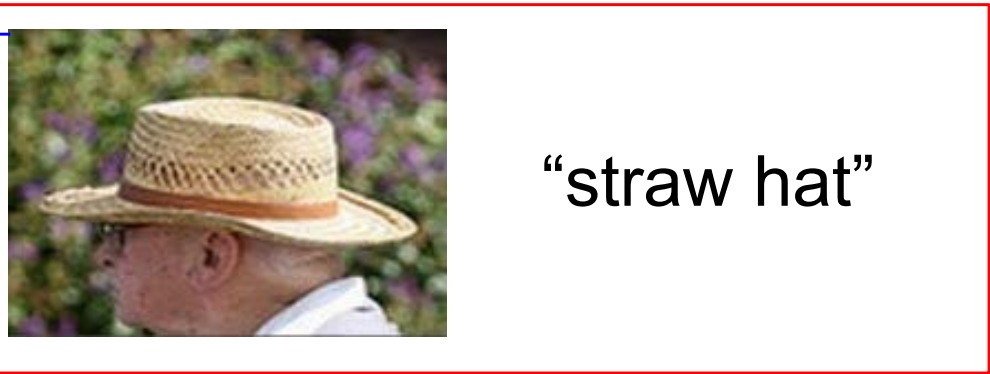




“straw hat”

training example





“straw hat”

training example



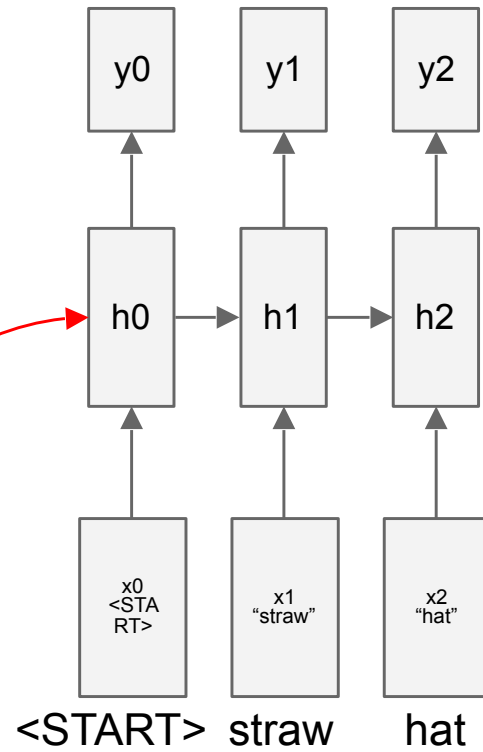
“straw hat”

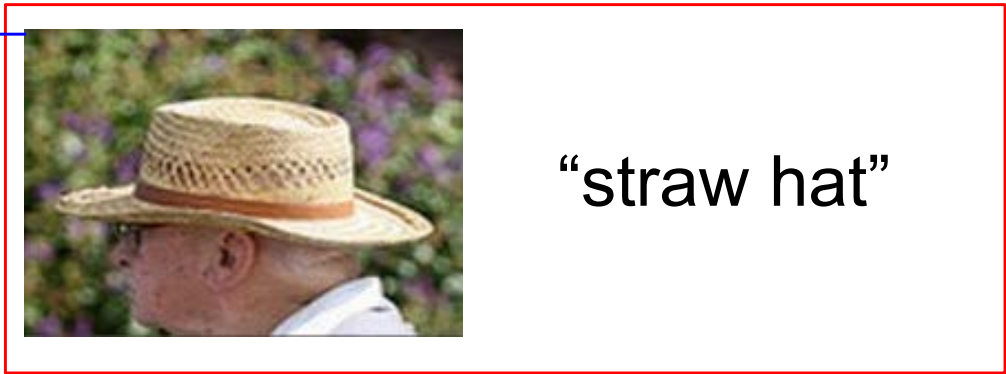
training example



“straw hat”

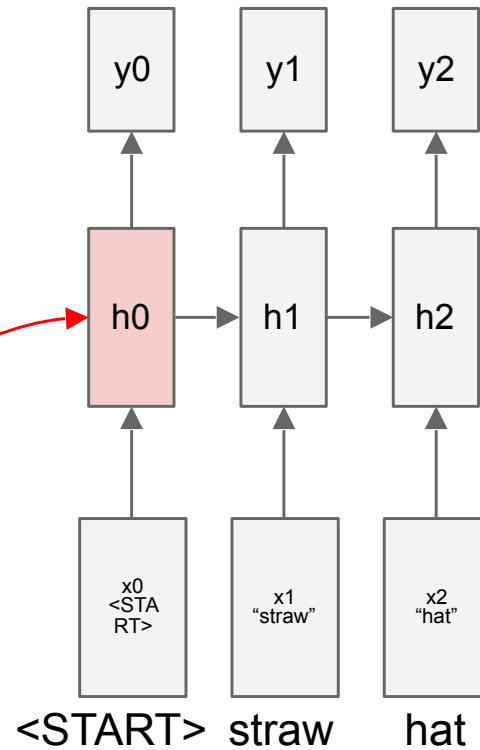
training example





“straw hat”

training example



before:

$$h_0 = \tanh(0, W_{xh} * x_0)$$

now:

$$h_0 = \tanh(0, W_{xh} * x_0 + W_{ih} * v)$$

slide credit: Andrej Karpathy



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

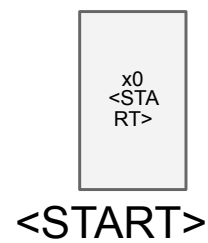
maxpool

FC-4096

FC-4096



test image



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

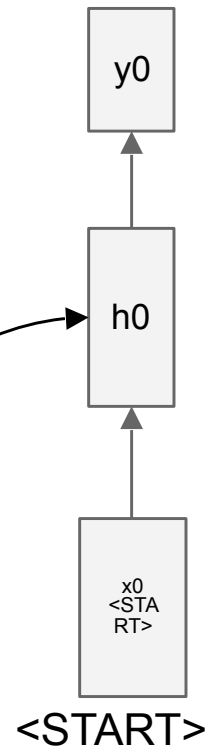
maxpool

FC-4096

FC-4096

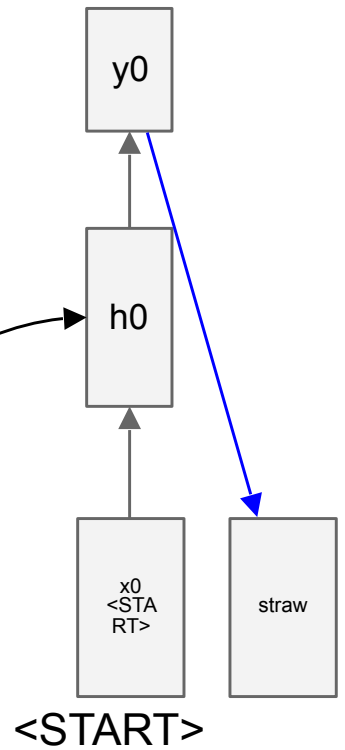


test image





test image

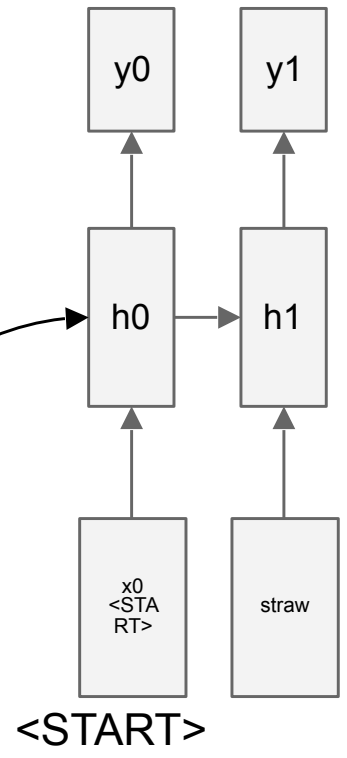


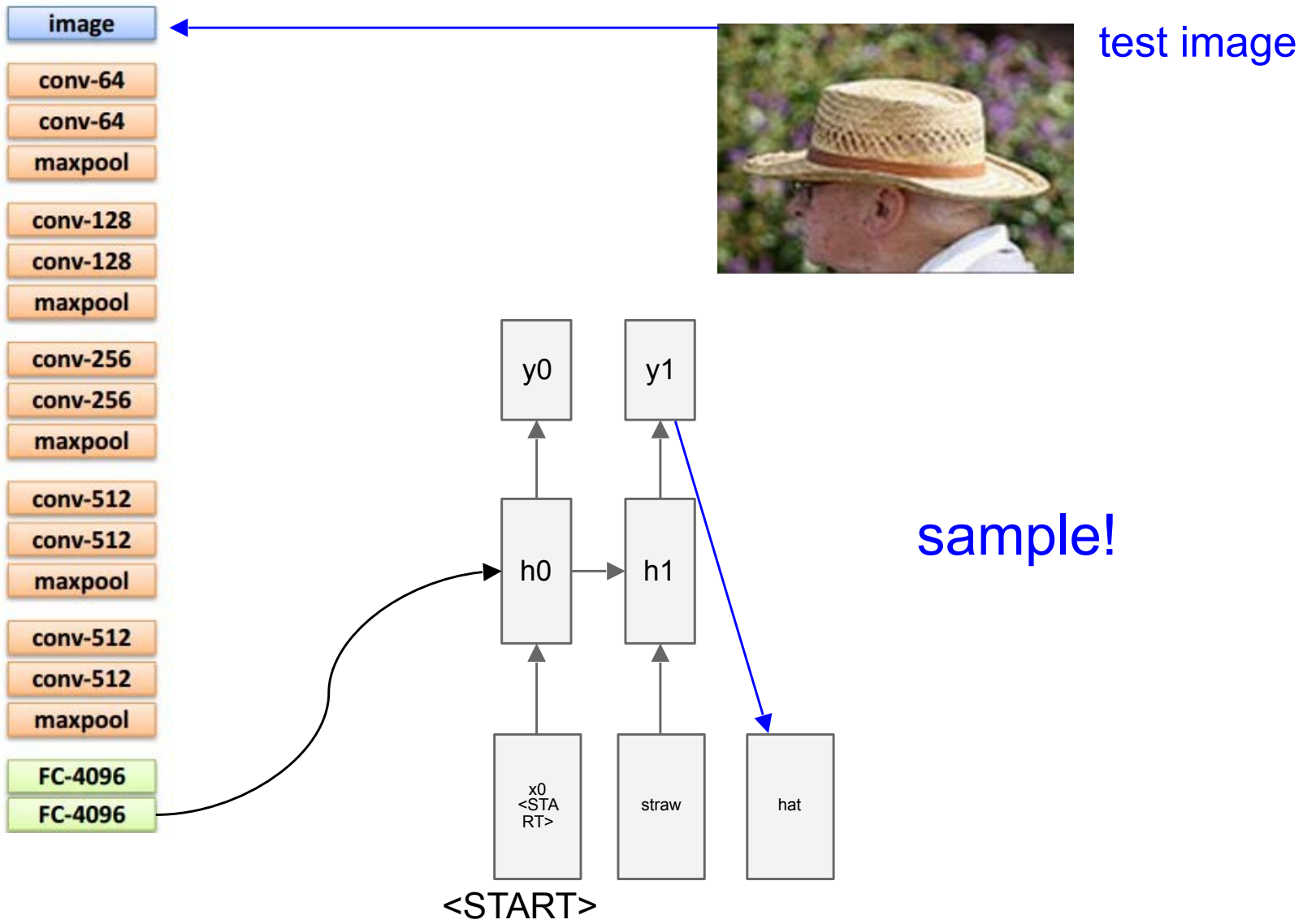
sample!

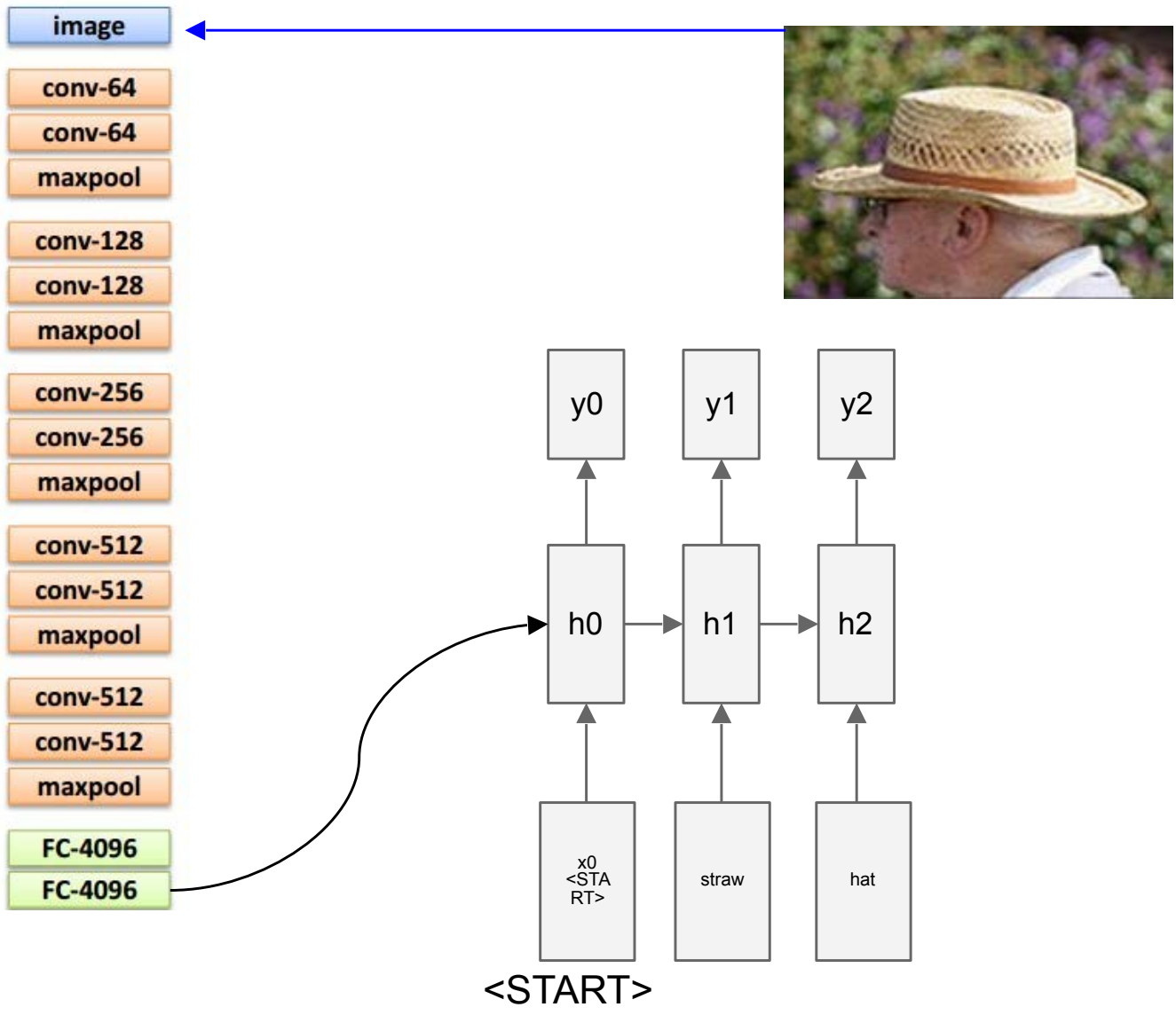


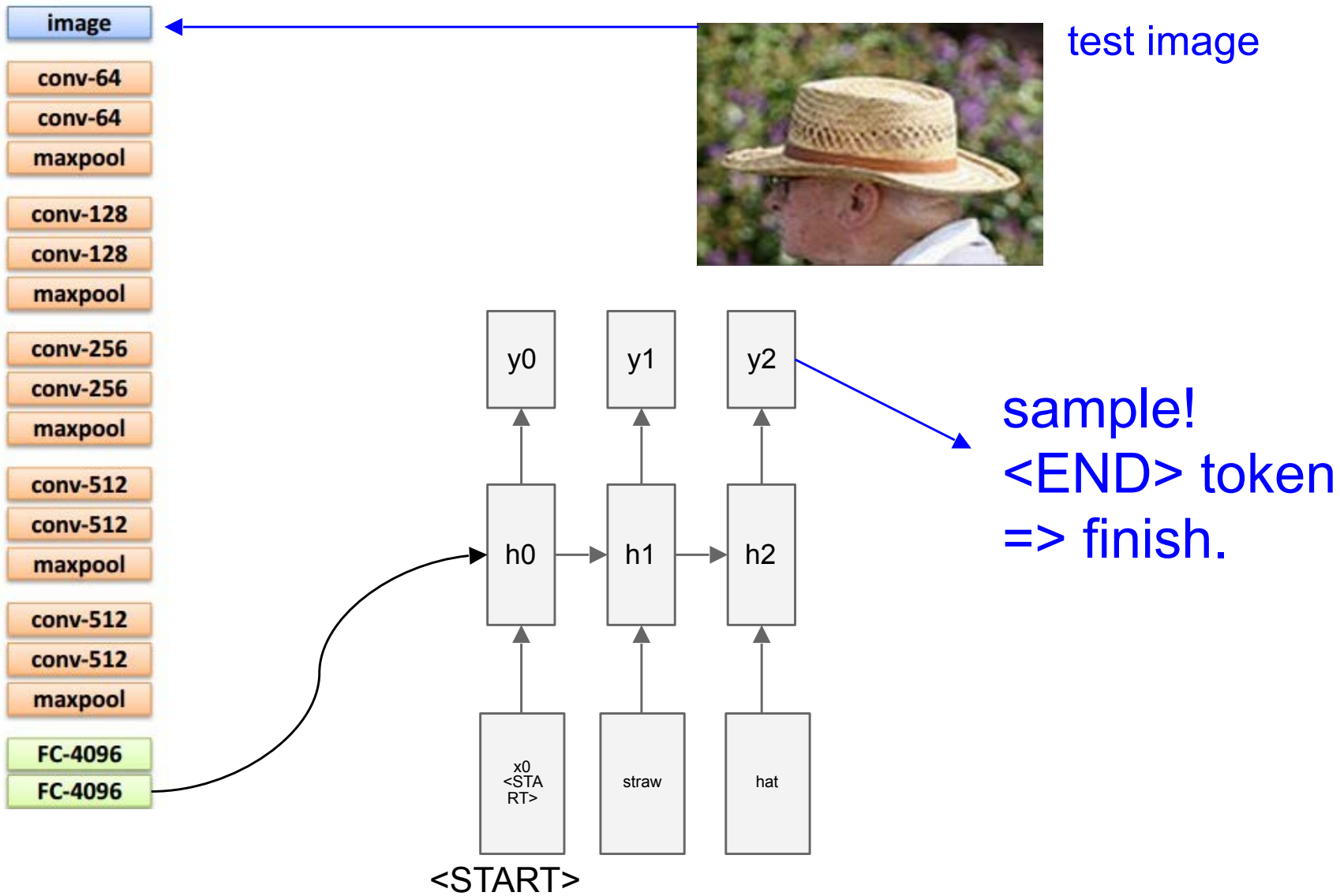


test image









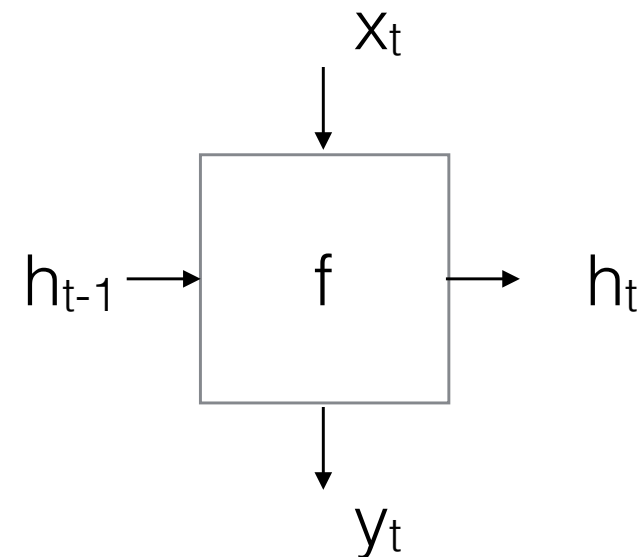
# Sequence Learning

- Instances of the form  $\mathbf{x} = \langle x_1, x_2, x_3, \dots, x_T \rangle$
- Variable sequence length  $T$
- Learn a transition function  $f$  with parameters  $W$ :
- $f$  should update hidden state  $h_t$  and output  $y_t$

$$h_0 := 0$$

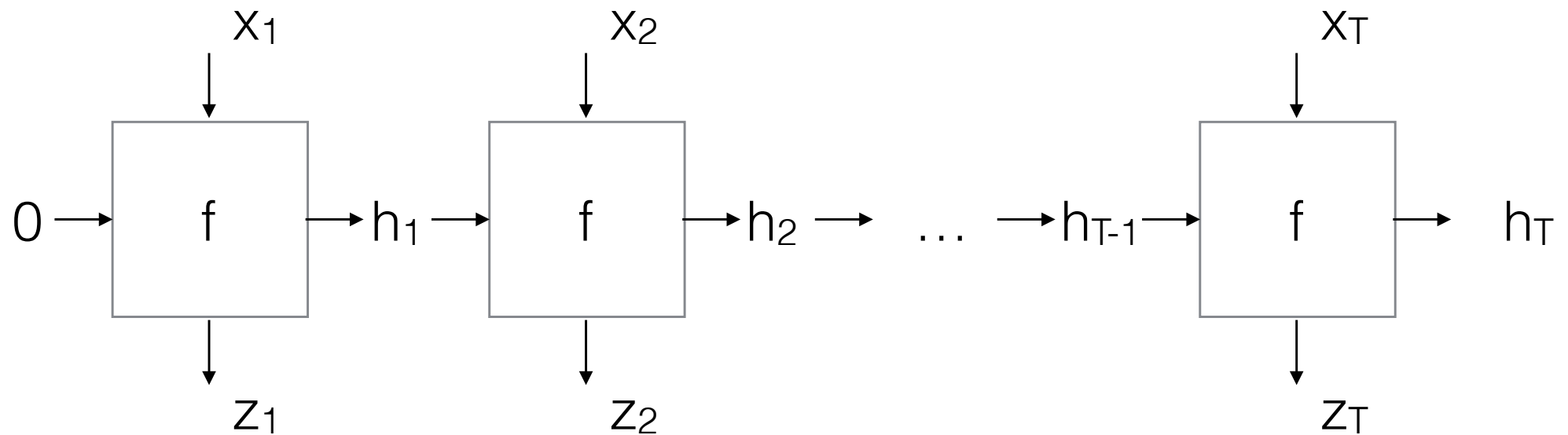
for  $t = 1, 2, 3, \dots, T$ :

$$\langle y_t, h_t \rangle = f_W(x_t, h_{t-1})$$



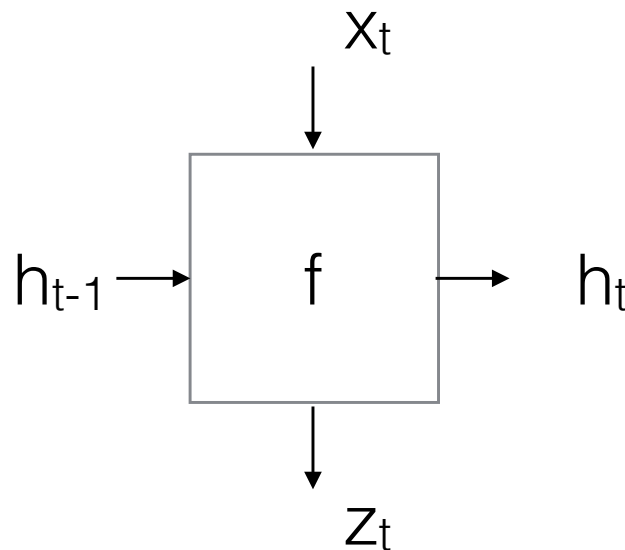
# Sequence Learning

Equivalent to a T-layer deep network, unrolled in time



# Sequence Learning

- What should the transition function  $f$  be?



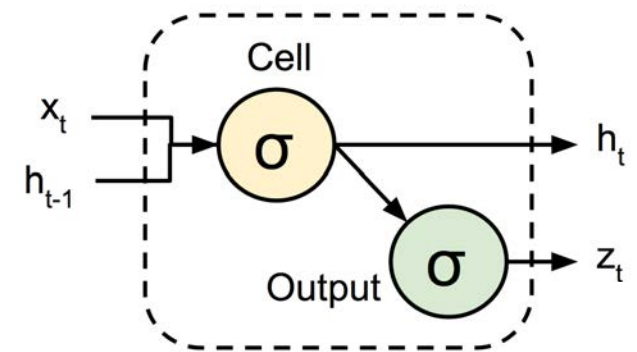
- At a minimum, we want something **non-linear** and **differentiable**

# Sequence Learning

- First attempt — a “vanilla” RNN:

$$h_t = \sigma(W_{hx}x_t + W_{hh}h_{t-1} + b_h)$$

$$z_t = \sigma(W_{hz}h_t + b_z)$$

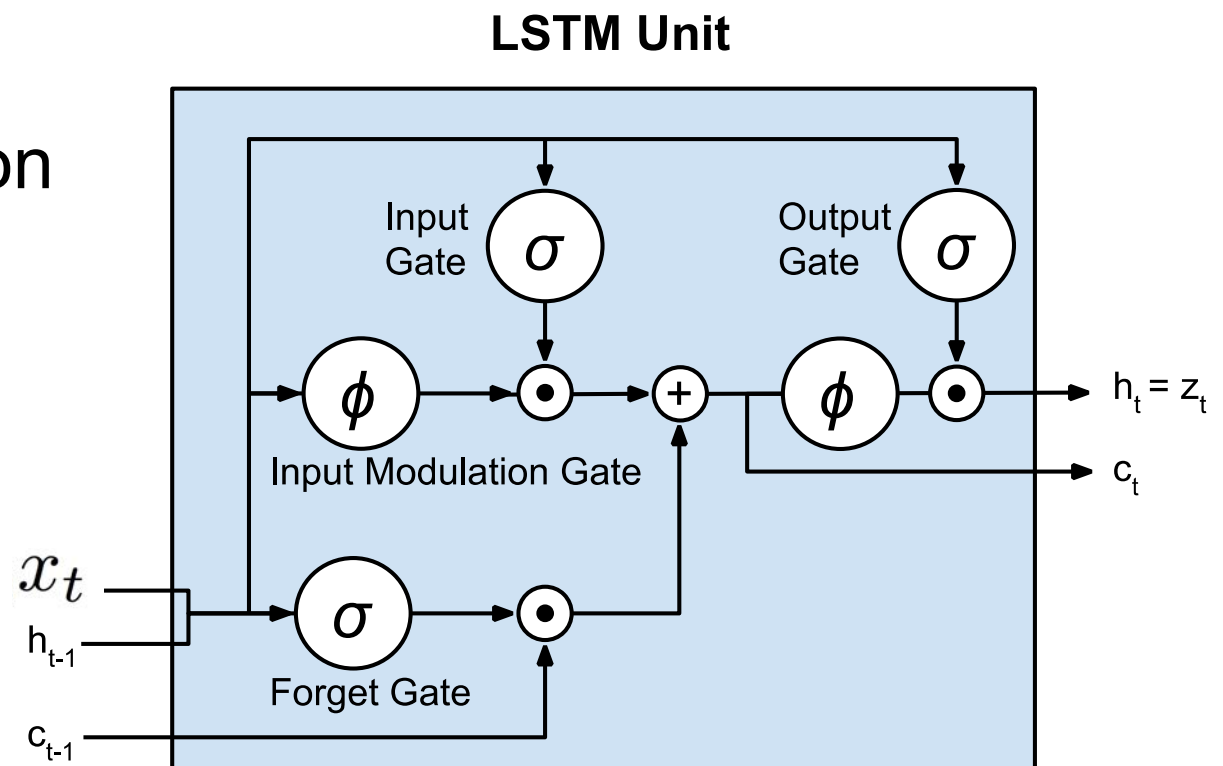


- Problems
  - Difficult to train — vanishing/exploding gradients
  - Unable to “select” inputs, hidden state, outputs



# Sequence Learning

- LSTM - Long Short Term Memory [Hochreiter & Schmidhuber, 1997]
- Selectively propagate or forget hidden state
- Allows long-term dependencies to be learned
- Effective for
  - speech recognition
  - handwriting recognition
  - translation
  - parsing



# LSTM for sequence modeling

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$

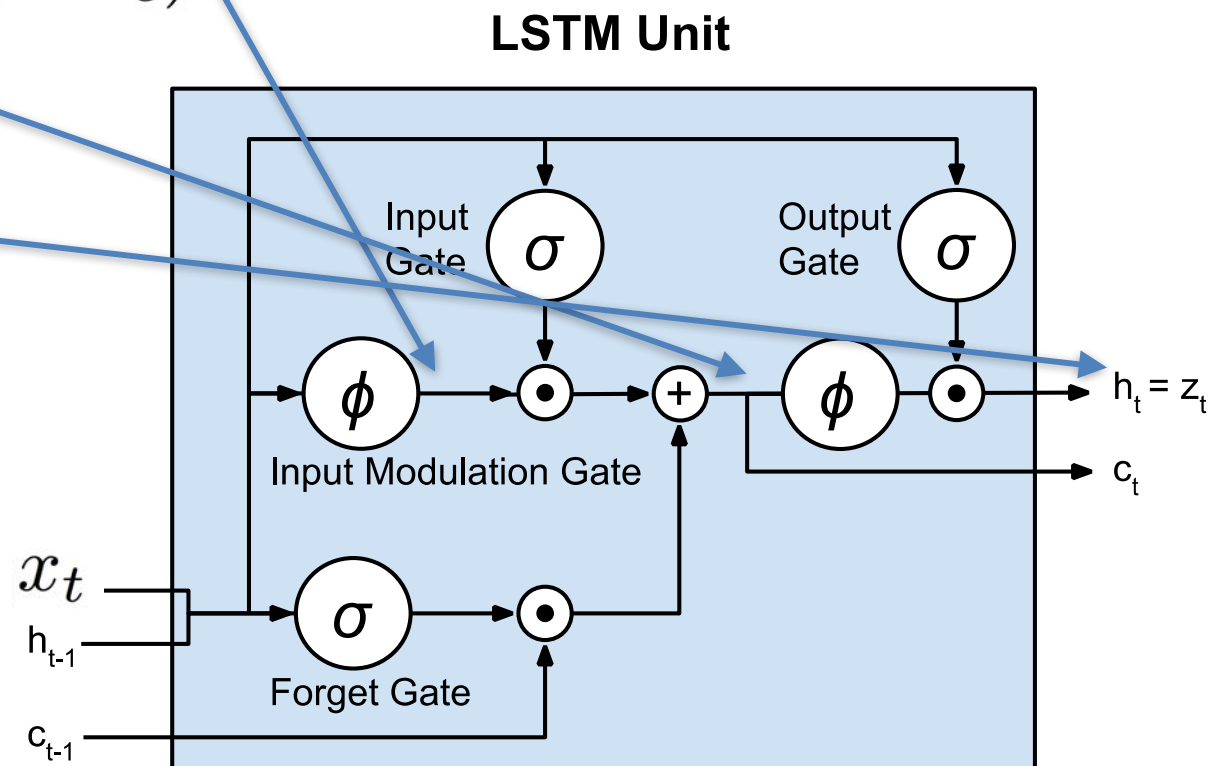
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$

$$g_t = \phi(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$

$$h_t = o_t \odot \phi(c_t)$$



# LSTM for sequence modeling

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$

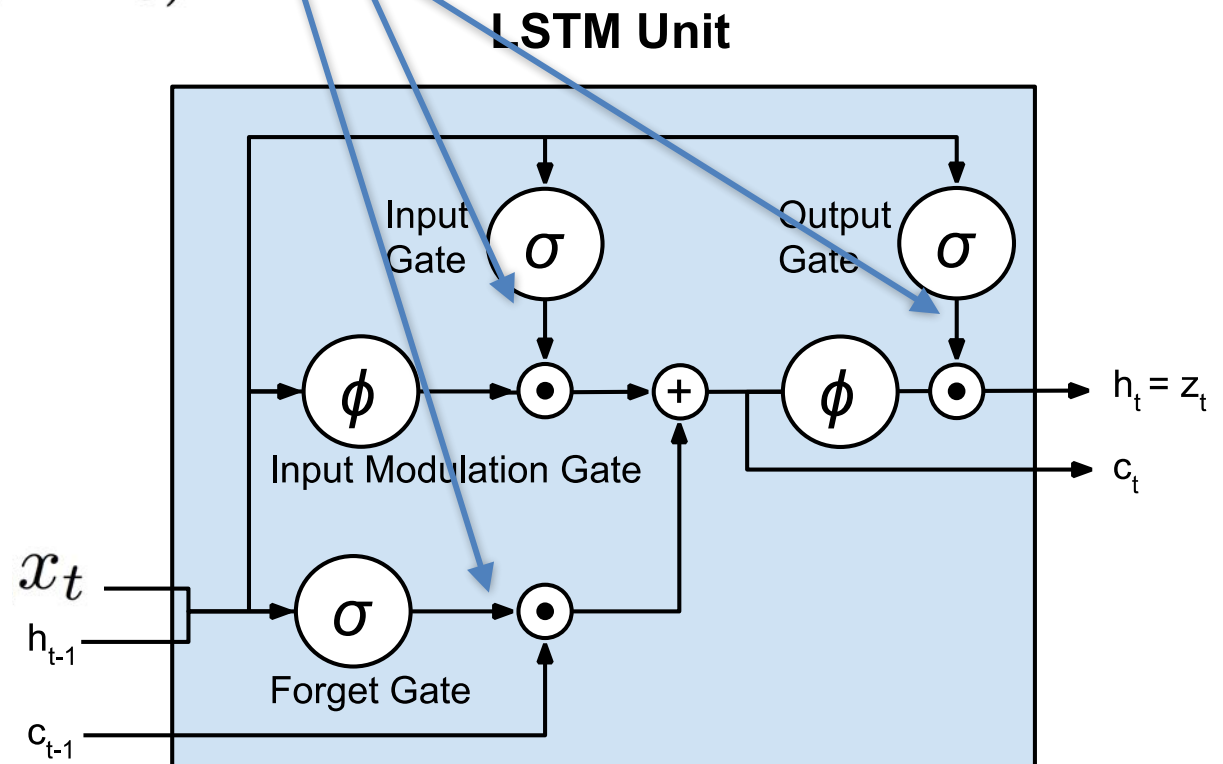
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$

$$g_t = \phi(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

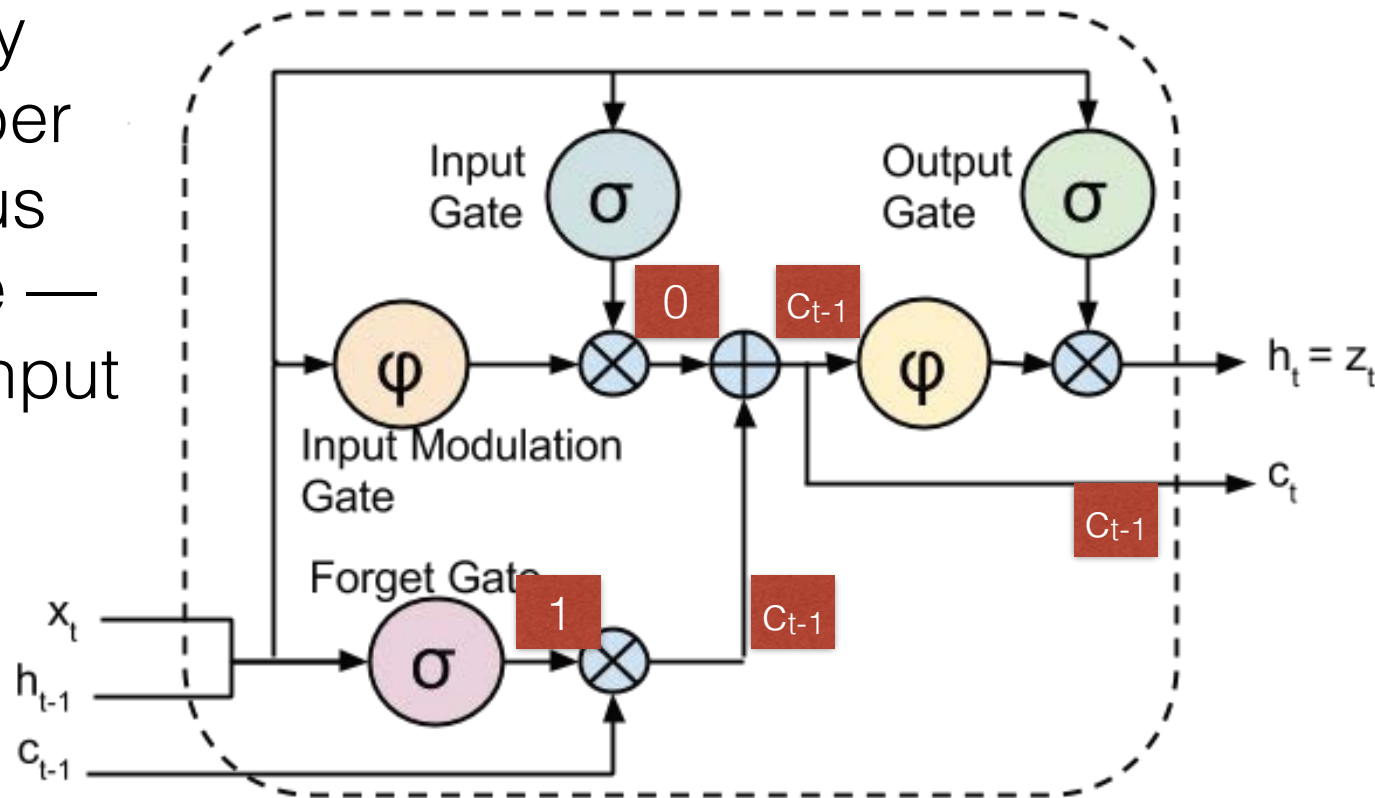
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$

$$h_t = o_t \odot \phi(c_t)$$



# Sequence Learning

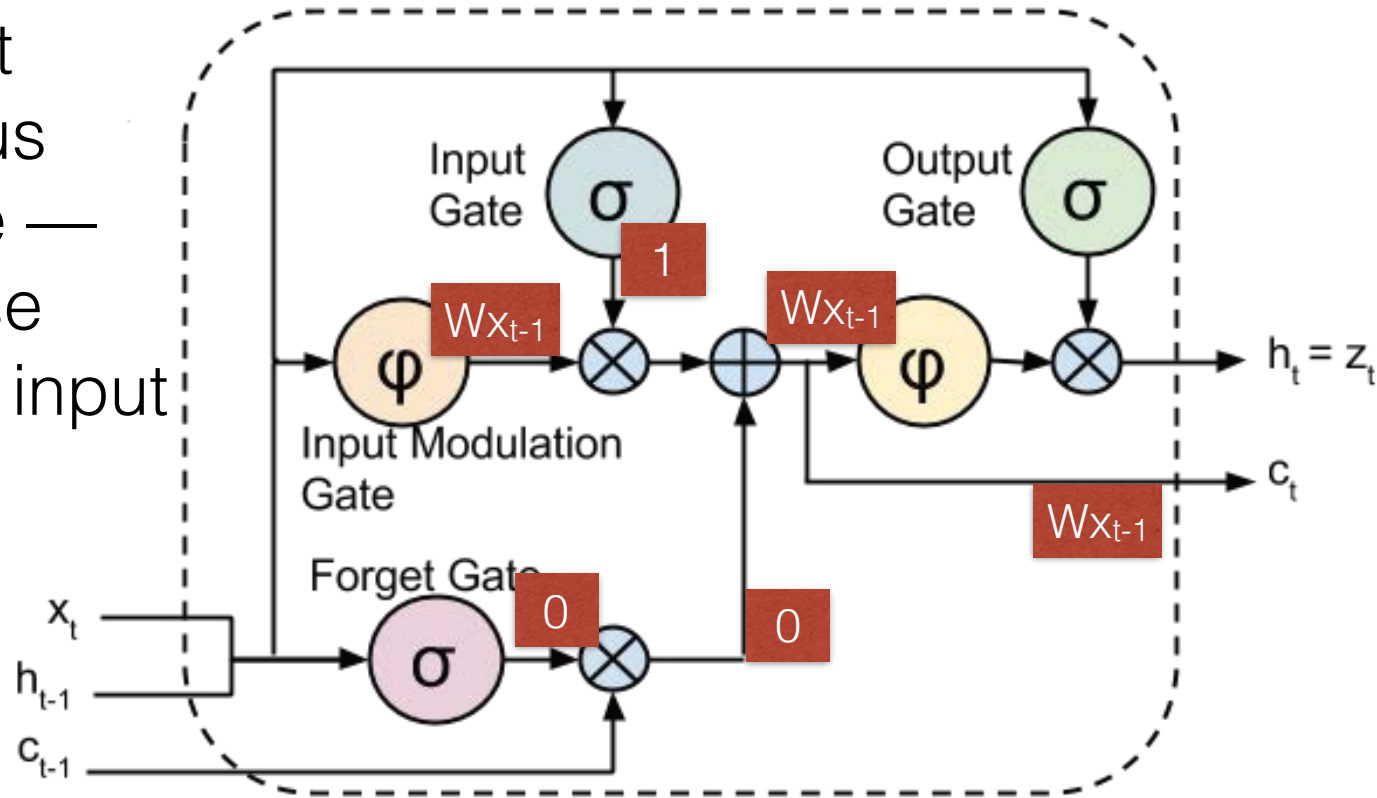
Exactly  
remember  
previous  
cell state —  
discard input



LSTM  
(Hochreiter &  
Schmidhuber, 1997)

# Sequence Learning

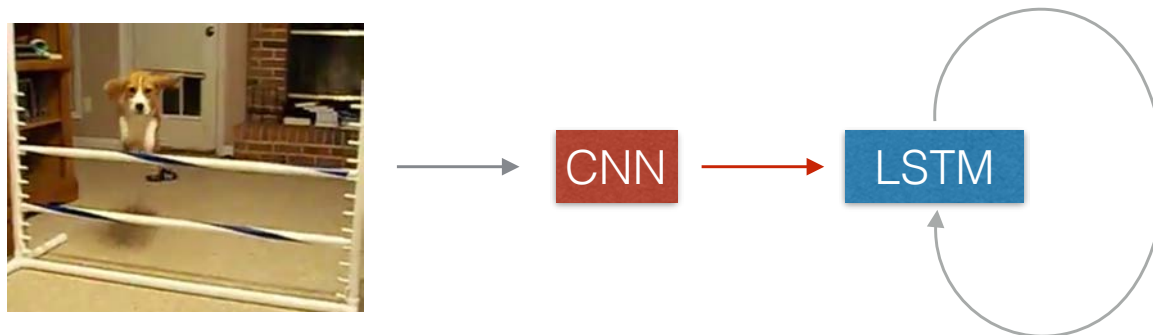
Forget  
previous  
cell state —  
only use  
modulated input



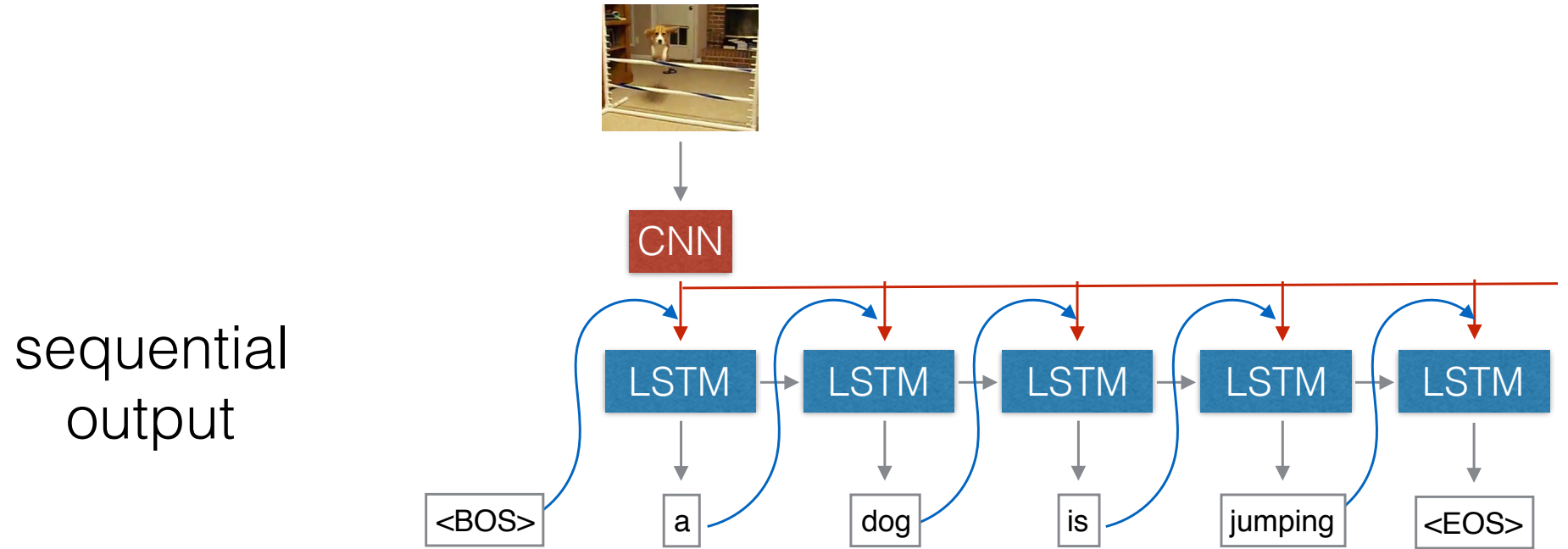
LSTM  
(Hochreiter &  
Schmidhuber, 1997)

# LRCN

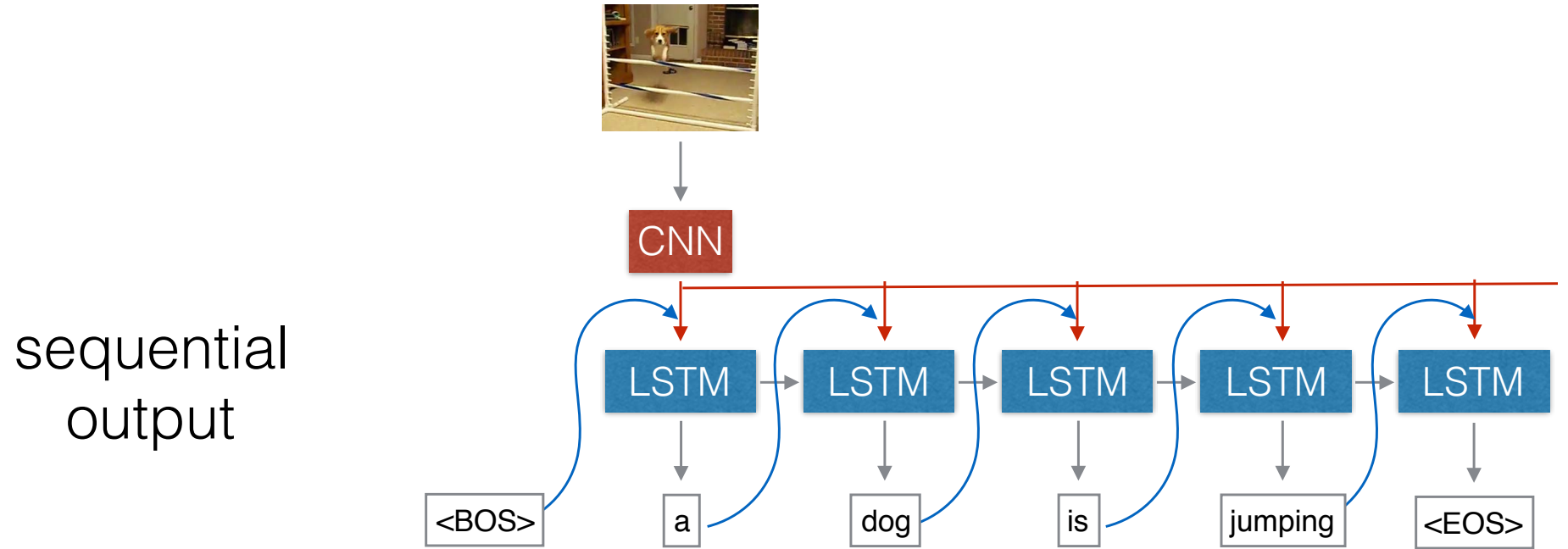
- Long-term Recurrent Convolutional Networks
- End-to-end trainable framework for sequence problems in vision

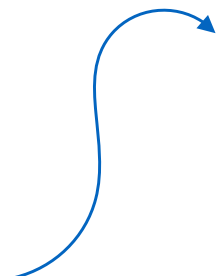


# Image Description



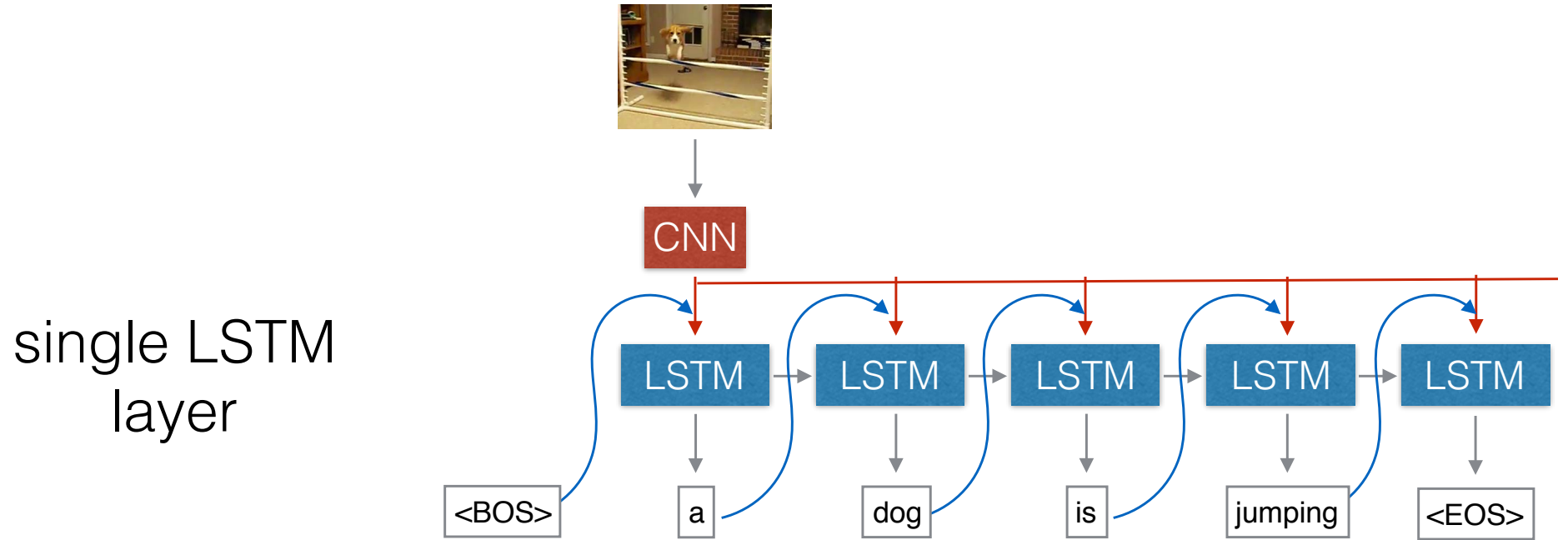
# Image Description



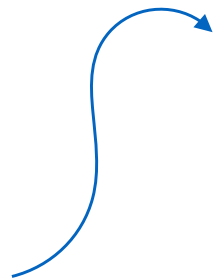
 = embed a one-hot vector



# Image Description

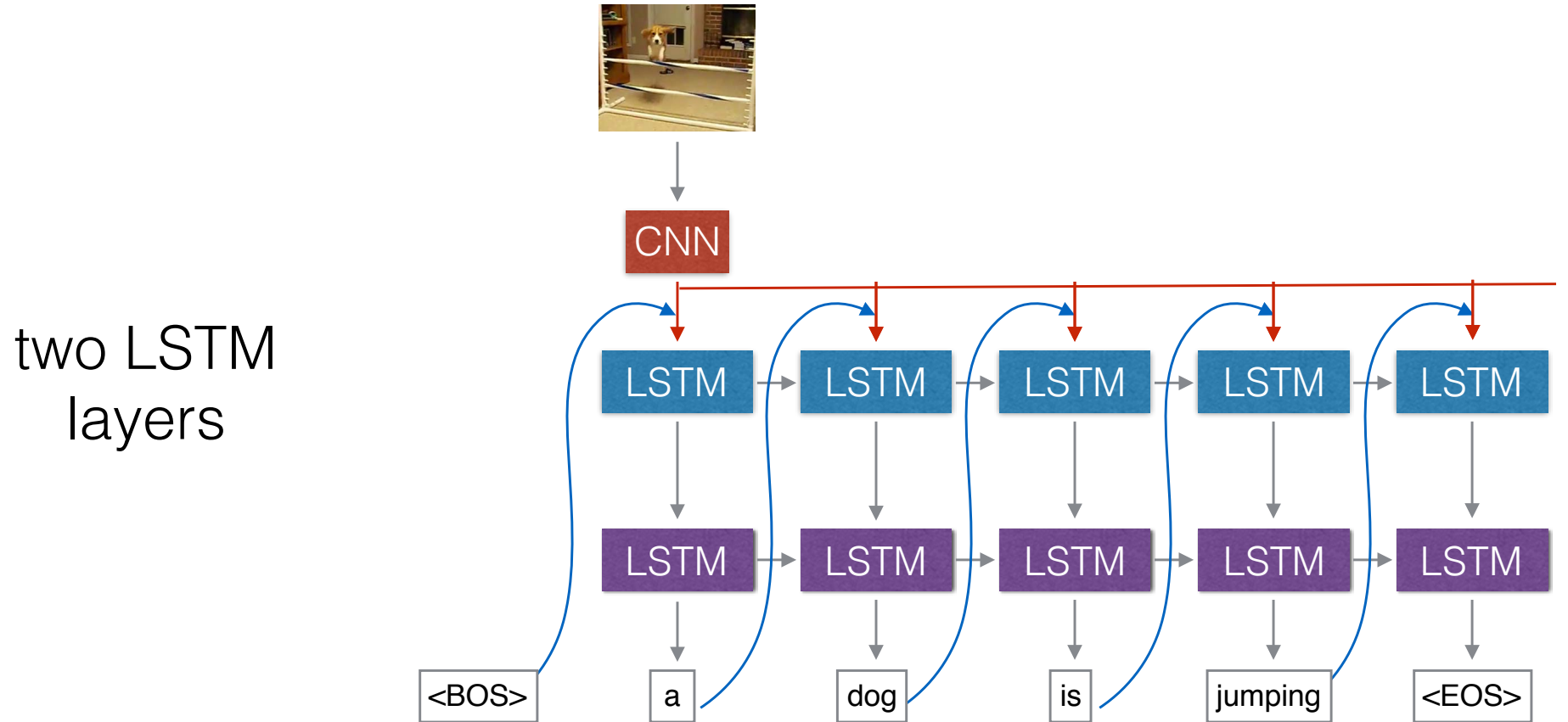


single LSTM layer



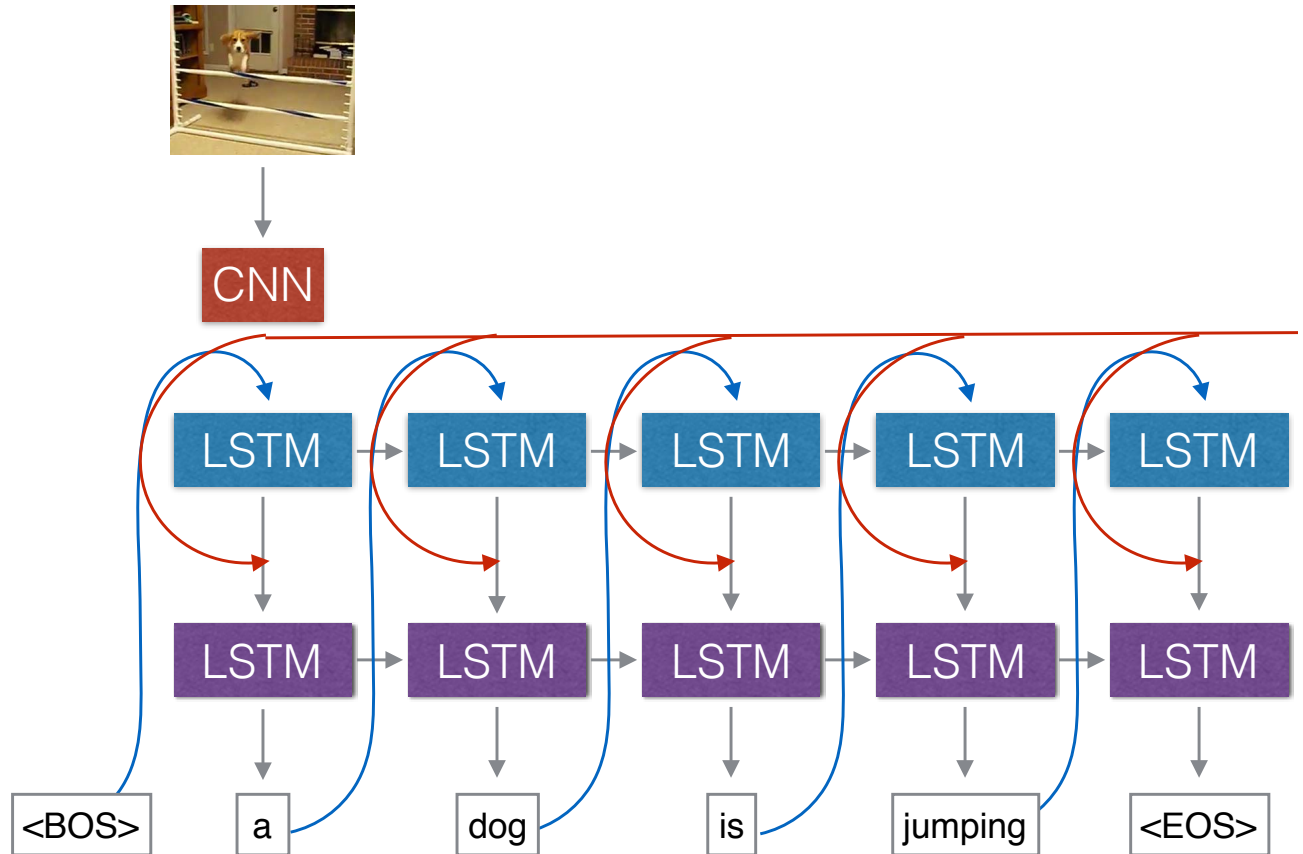
= embed a one-hot vector

# Image Description



# Image Description

two LSTM  
layers, factored



# Image Description

Architecture	Flickr30k [1] Caption-to-Image Recall@1
Single Layer	14.1%
Two Layer	3.8%
Two Layer, Factored	<b>17.5%</b>
Four Layer, Factored	15.8%

[1] P. Hodosh, A. Young, M. Lai, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions.

# Image Description

COCO [1] CIDEr-D c5 Scores	CaffeNet	VGGNet [2]
Raw	68.8%	77.3%
Finetuned	75.8%	<b>83.9%</b>

[1] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft coco: Common objects in context. *arXiv preprint arXiv:1405.0312*, 2014.

[2] K. Simonyan & A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. ICLR 2015.

# Image Description



A female tennis player in action on the court.



A group of young men playing a game of soccer.



A man riding a wave on top of a surfboard.

# Image Description



A black and white cat is sitting on a chair.

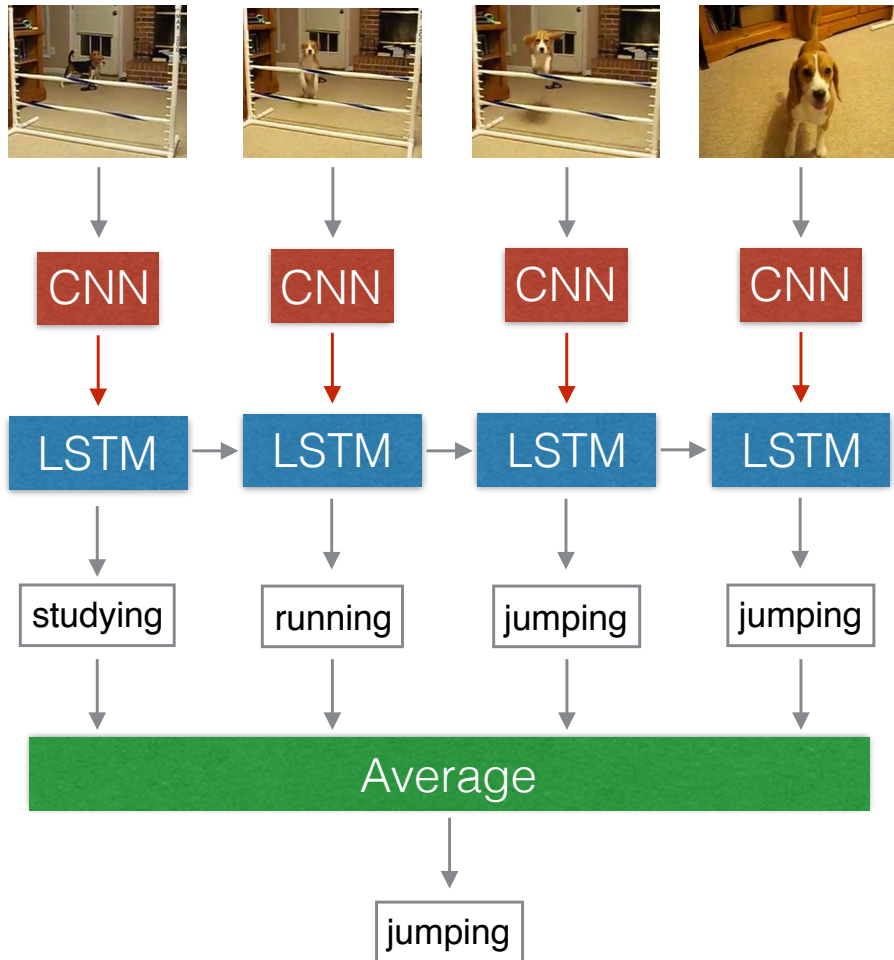


A large clock mounted to the side of a building.



A bunch of fruit that are sitting on a table.

# Activity Recognition



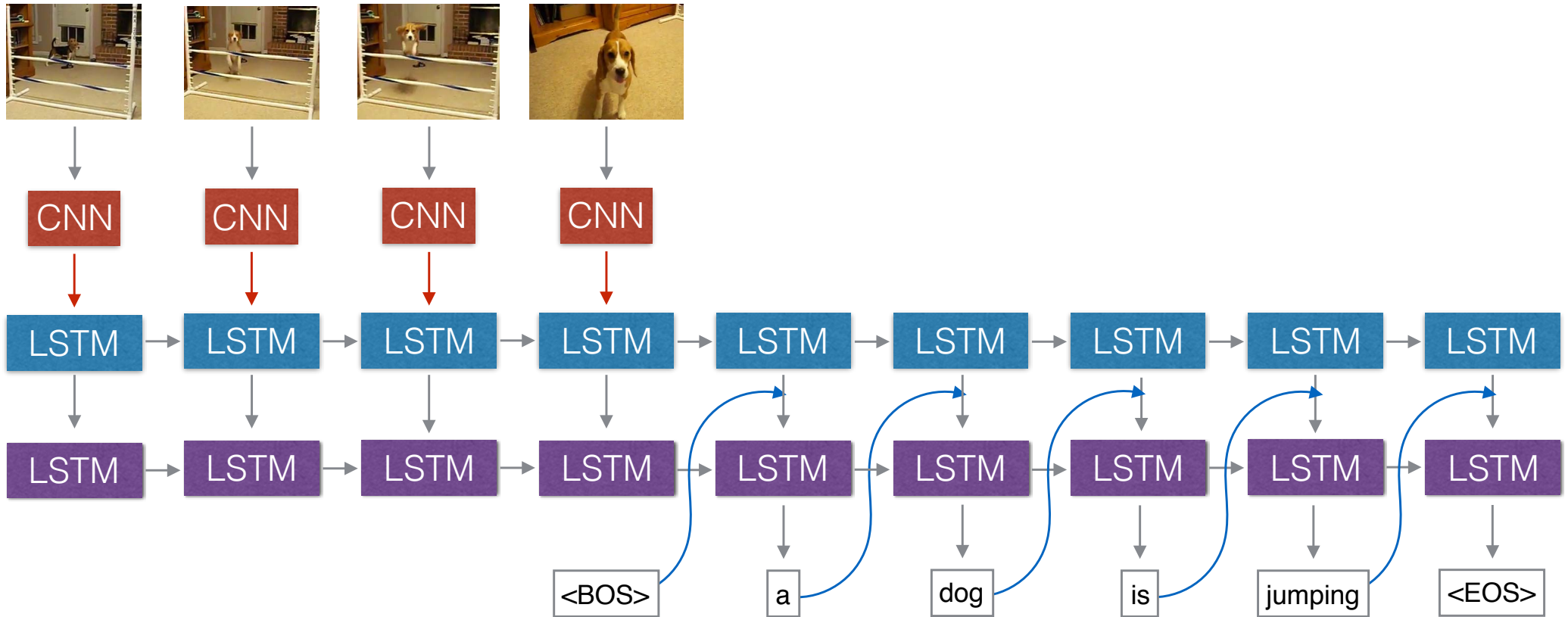
sequential  
input



# Activity Recognition

UCF101 Class. Acc.	RGB	Optical Flow	RGB+ Flow
Single- Frame CNN	67.7%	72.2%	78.8%
LRCN	<b>68.2%</b>	<b>77.5%</b>	<b>82.7%</b>

# Video Description



sequential input & output

# Video Description

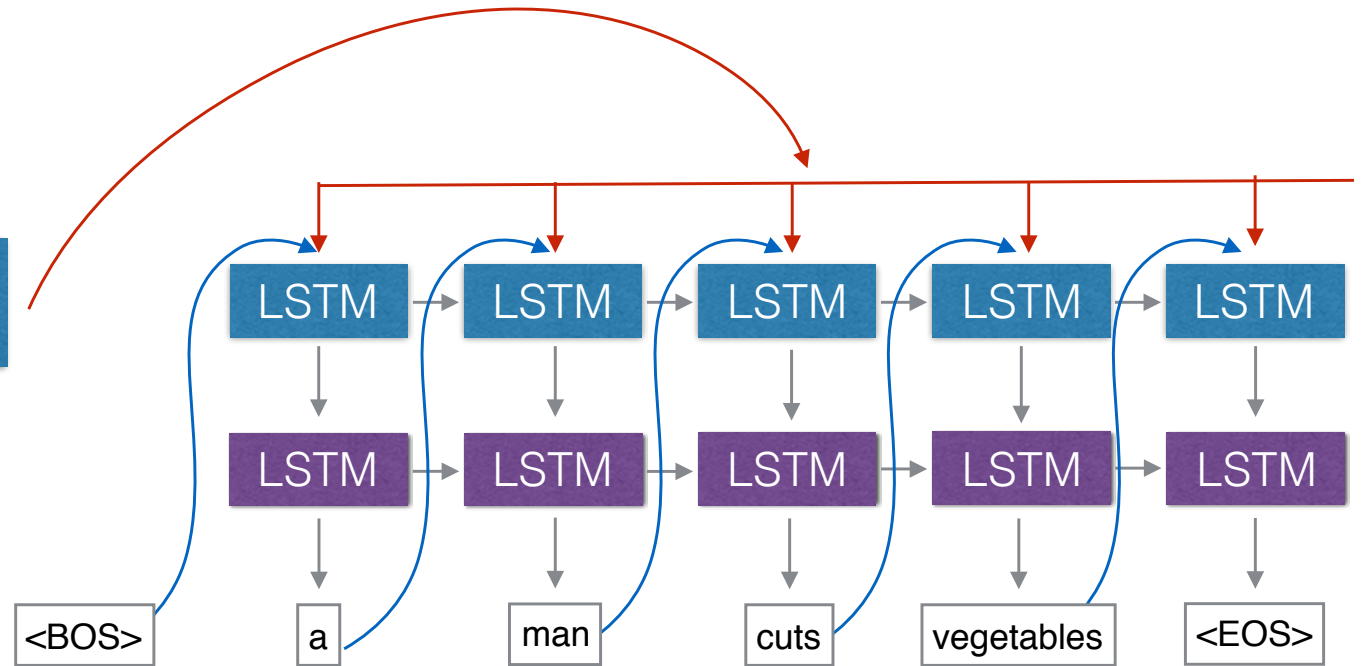
MPII TACoS Multi-Level Dataset



# Video Description



Pre-trained Detector Predictions

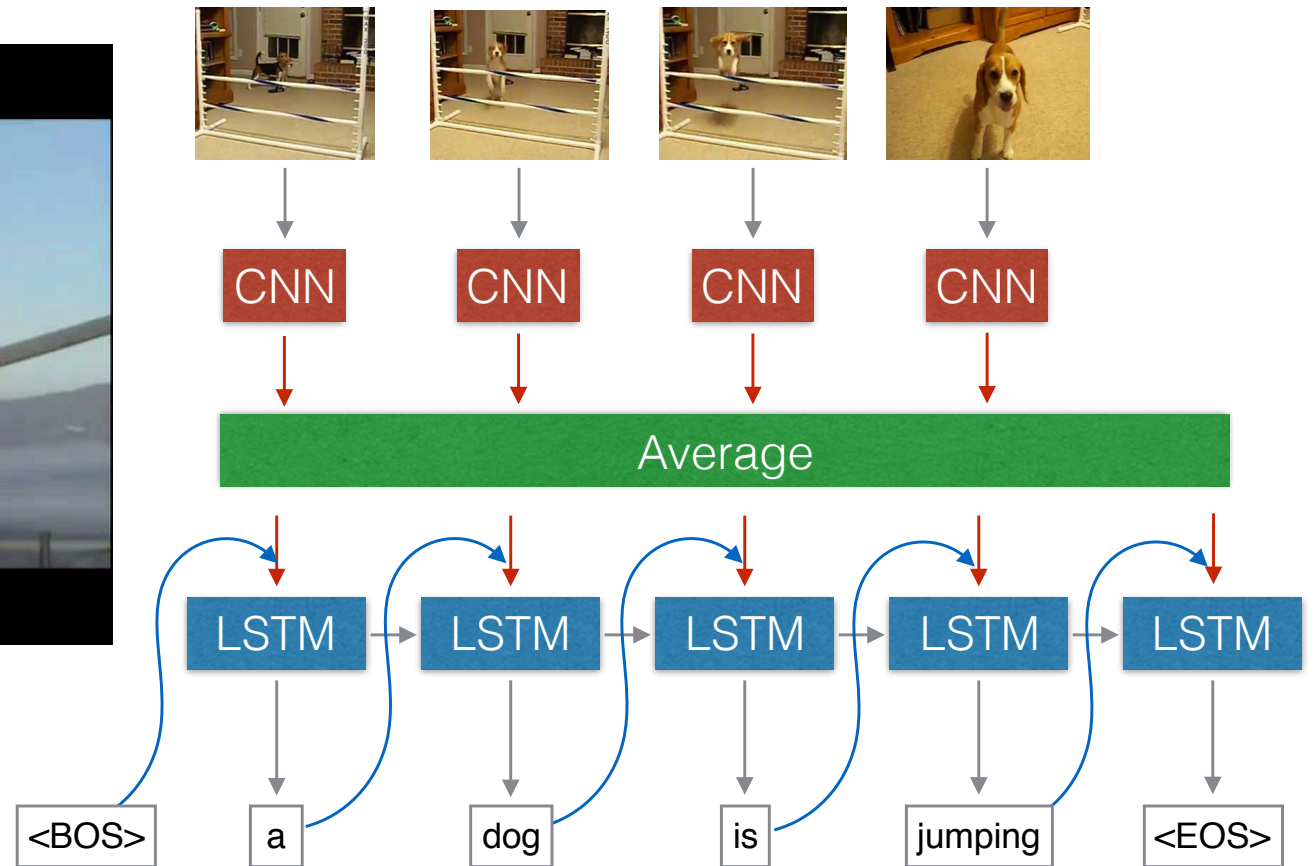


# Video Description



Approach	Generation Accuracy (BLEU)
SMT	26.9%
LRCN	<b>28.8%</b>

# Video Description





# Wow I can't believe that worked



a group of people standing  
around a room with  
remotes  
logprob: -9.17



a young boy is holding a  
baseball bat  
logprob: -7.61



a cow is standing in the middle of a street  
logprob: -8.84

# Wow I can't believe that worked



a cat is sitting on a toilet seat  
logprob: -7.79



a display case filled with lots of different types of donuts  
logprob: -7.78



a group of people sitting at a table with wine glasses  
logprob: -6.71



Well, I can kind of see it



a man standing next to a clock on a wall  
logprob: -10.08



a young boy is holding a  
baseball bat  
logprob: -7.65



a cat is sitting on a couch with a remote control  
logprob: -12.45

Well, I can kind of see it



a baby laying on a bed with a stuffed bear  
logprob: -8.66



a table with a plate of food and a cup of coffee  
logprob: -9.93



a young boy is playing frisbee in the park  
logprob: -9.52



# Not sure what happened there...



a toilet with a seat up in a bathroom  
logprob: -13.44



a woman holding a teddy bear in front of a mirror  
logprob: -9.65



a horse is standing in the middle of a road  
logprob: -10.34

# “The Unreasonable Effectiveness of Recurrent Neural Networks”

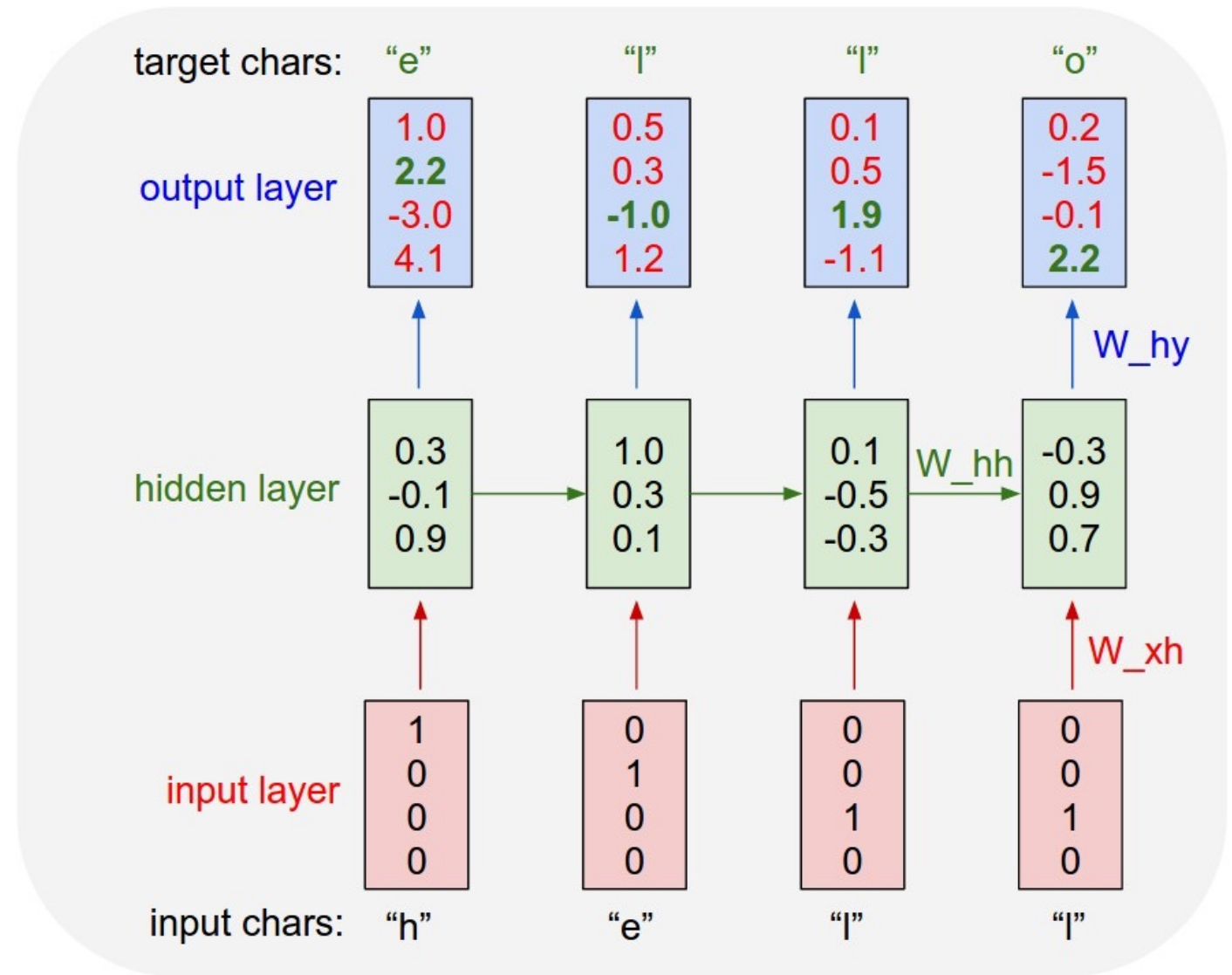
[karpathy.github.io](http://karpathy.github.io)

# Character-level language model example

Vocabulary:  
[h,e,l,o]

Example training sequence:  
"hello"

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



## Sonnet 116 – Let me not ...

*by William Shakespeare*





















Let me not to the marriage of true minds  
Admit impediments. Love is not love  
Which alters when it alteration finds,  
Or bends with the remover to remove:  
O no! it is an ever-fixed mark  
That looks on tempests and is never shaken;  
It is the star to every wandering bark,  
Whose worth's unknown, although his height be taken.  
Love's not Time's fool, though rosy lips and cheeks  
Within his bending sickle's compass come:  
Love alters not with his brief hours and weeks,  
But bears it out even to the edge of doom.  
If this be error and upon me proved,  
I never writ, nor no man ever loved.



# The Stacks Project

[home](#)
[about](#)
[tags explained](#)
[tag lookup](#)
[browse](#)
[search](#)
[bibliography](#)
[recent comments](#)
[blog](#)
[add slogans](#)

## Browse chapters

Part	Chapter	online	TeX source	view pdf
Preliminaries				
	1. Introduction	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	2. Conventions	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	3. Set Theory	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	4. Categories	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	5. Topology	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	6. Sheaves on Spaces	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	7. Sites and Sheaves	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	8. Stacks	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	9. Fields	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 
	10. Commutative Algebra	<a href="#">online</a>	<a href="#">tex</a> 	<a href="#">pdf</a> 

## Parts

- [Preliminaries](#)
- [Schemes](#)
- [Topics in Scheme Theory](#)
- [Algebraic Spaces](#)
- [Topics in Geometry](#)
- [Deformation Theory](#)
- [Algebraic Stacks](#)
- [Miscellany](#)

## Statistics

The Stacks project now consists of

- 455910 lines of code
- 14221 tags (56 inactive tags)
- 2366 sections



For  $\bigoplus_{n=1, \dots, m}$  where  $\mathcal{L}_{m_\bullet} = 0$ , hence we can find a closed subset  $\mathcal{H}$  in  $\mathcal{H}$  and any sets  $\mathcal{F}$  on  $X$ ,  $U$  is a closed immersion of  $S$ , then  $U \rightarrow T$  is a separated algebraic space.

*Proof.* Proof of (1). It also start we get

$$S = \text{Spec}(R) = U \times_X U \times_X U$$

and the comparicoly in the fibre product covering we have to prove the lemma generated by  $\coprod Z \times_U U \rightarrow V$ . Consider the maps  $M$  along the set of points  $Sch_{fppf}$  and  $U \rightarrow U$  is the fibre category of  $S$  in  $U$  in Section, ?? and the fact that any  $U$  affine, see Morphisms, Lemma ???. Hence we obtain a scheme  $S$  and any open subset  $W \subset U$  in  $Sh(G)$  such that  $\text{Spec}(R') \rightarrow S$  is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that  $f_i$  is of finite presentation over  $S$ . We claim that  $\mathcal{O}_{X,x}$  is a scheme where  $x, x', s'' \in S'$  such that  $\mathcal{O}_{X,x'} \rightarrow \mathcal{O}_{X',x''}$  is separated. By Algebra, Lemma ?? we can define a map of complexes  $GL_{S'}(x'/S'')$  and we win.  $\square$

To prove study we see that  $\mathcal{F}|_U$  is a covering of  $\mathcal{X}'$ , and  $\mathcal{T}_i$  is an object of  $\mathcal{F}_{X/S}$  for  $i > 0$  and  $\mathcal{F}_p$  exists and let  $\mathcal{F}_i$  be a presheaf of  $\mathcal{O}_X$ -modules on  $\mathcal{C}$  as a  $\mathcal{F}$ -module. In particular  $\mathcal{F} = U/\mathcal{F}$  we have to show that

$$\widetilde{M}^\bullet = \mathcal{I}^\bullet \otimes_{\text{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1} \mathcal{F}$$

is a unique morphism of algebraic stacks. Note that

$$\text{Arrows} = (Sch/S)_{fppf}^{opp}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \mapsto (U, \text{Spec}(A))$$

is an open subset of  $X$ . Thus  $U$  is affine. This is a continuous map of  $X$  is the inverse, the groupoid scheme  $S$ .

*Proof.* See discussion of sheaves of sets.  $\square$

The result for prove any open covering follows from the less of Example ???. It may replace  $S$  by  $X_{spaces, \acute{e}tale}$  which gives an open subspace of  $X$  and  $T$  equal to  $S_{Zar}$ , see Descent, Lemma ???. Namely, by Lemma ?? we see that  $R$  is geometrically regular over  $S$ .

**Lemma 0.1.** Assume (3) and (3) by the construction in the description.

Suppose  $X = \lim |X|$  (by the formal open covering  $X$  and a single map  $\underline{Proj}_X(\mathcal{A}) = \text{Spec}(B)$  over  $U$  compatible with the complex

$$\text{Set}(\mathcal{A}) = \Gamma(X, \mathcal{O}_{X, \mathcal{O}_X}).$$

When in this case of to show that  $\mathcal{Q} \rightarrow \mathcal{C}_{Z/X}$  is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition ?? (without element is when the closed subschemes are catenary. If  $T$  is surjective we may assume that  $T$  is connected with residue fields of  $S$ . Moreover there exists a closed subspace  $Z \subset X$  of  $X$  where  $U$  in  $X'$  is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem

(1)  $f$  is locally of finite type. Since  $S = \text{Spec}(R)$  and  $Y = \text{Spec}(R)$ .

*Proof.* This is form all sheaves of sheaves on  $X$ . But given a scheme  $U$  and a surjective étale morphism  $U \rightarrow X$ . Let  $U \cap U = \coprod_{i=1, \dots, n} U_i$  be the scheme  $X$  over  $S$  at the schemes  $X_i \rightarrow X$  and  $U = \lim_i X_i$ .  $\square$

The following lemma surjective restrocomposes of this implies that  $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{\mathcal{X}, \dots, 0}$ .

**Lemma 0.2.** Let  $X$  be a locally Noetherian scheme over  $S$ ,  $E = \mathcal{F}_{X/S}$ . Set  $\mathcal{I} = \mathcal{J}_1 \subset \mathcal{I}'_n$ . Since  $\mathcal{I}^n \subset \mathcal{I}^n$  are nonzero over  $i_0 \leq \mathfrak{p}$  is a subset of  $\mathcal{J}_{n,0} \circ \overline{A}_2$  works.

**Lemma 0.3.** In Situation ???. Hence we may assume  $\mathfrak{q}' = 0$ .

*Proof.* We will use the property we see that  $\mathfrak{p}$  is the next functor (??). On the other hand, by Lemma ?? we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_X(D)$$

where  $K$  is an  $F$ -algebra where  $\delta_{n+1}$  is a scheme over  $S$ .  $\square$



*Proof.* Omitted. □

**Lemma 0.1.** *Let  $\mathcal{C}$  be a set of the construction.*

*Let  $\mathcal{C}$  be a gerber covering. Let  $\mathcal{F}$  be a quasi-coherent sheaves of  $\mathcal{O}$ -modules. We have to show that*

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

*Proof.* This is an algebraic space with the composition of sheaves  $\mathcal{F}$  on  $X_{\acute{e}tale}$  we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where  $\mathcal{G}$  defines an isomorphism  $\mathcal{F} \rightarrow \mathcal{F}$  of  $\mathcal{O}$ -modules. □

**Lemma 0.2.** *This is an integer  $\mathcal{Z}$  is injective.*

*Proof.* See Spaces, Lemma ?? □

**Lemma 0.3.** *Let  $S$  be a scheme. Let  $X$  be a scheme and  $X$  is an affine open covering. Let  $\mathcal{U} \subset \mathcal{X}$  be a canonical and locally of finite type. Let  $X$  be a scheme. Let  $X$  be a scheme which is equal to the formal complex.*

*The following to the construction of the lemma follows.*

*Let  $X$  be a scheme. Let  $X$  be a scheme covering. Let*

$$b : X \rightarrow Y' \rightarrow Y \rightarrow Y \rightarrow Y' \times_X Y \rightarrow X.$$

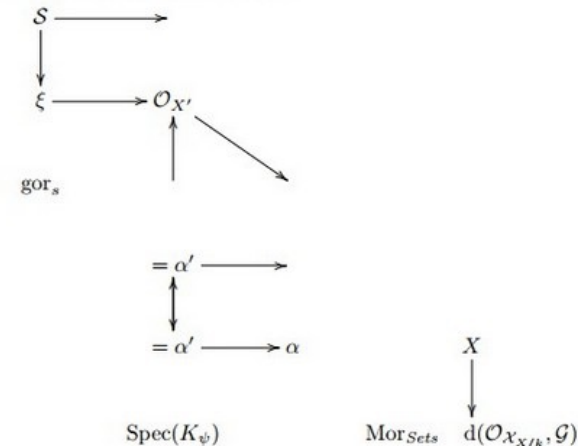
*be a morphism of algebraic spaces over  $S$  and  $Y$ .*

*Proof.* Let  $X$  be a nonzero scheme of  $X$ . Let  $X$  be an algebraic space. Let  $\mathcal{F}$  be a quasi-coherent sheaf of  $\mathcal{O}_X$ -modules. The following are equivalent

- (1)  $\mathcal{F}$  is an algebraic space over  $S$ .
- (2) If  $X$  is an affine open covering.

Consider a common structure on  $X$  and  $X$  the functor  $\mathcal{O}_X(U)$  which is locally of finite type. □

This since  $\mathcal{F} \in \mathcal{F}$  and  $x \in \mathcal{G}$  the diagram



is a limit. Then  $\mathcal{G}$  is a finite type and assume  $S$  is a flat and  $\mathcal{F}$  and  $\mathcal{G}$  is a finite type  $f_*$ . This is of finite type diagrams, and

- the composition of  $\mathcal{G}$  is a regular sequence,
- $\mathcal{O}_{X'}$  is a sheaf of rings.

□

*Proof.* We have see that  $X = \text{Spec}(R)$  and  $\mathcal{F}$  is a finite type representable by algebraic space. The property  $\mathcal{F}$  is a finite morphism of algebraic stacks. Then the cohomology of  $X$  is an open neighbourhood of  $U$ . □

*Proof.* This is clear that  $\mathcal{G}$  is a finite presentation, see Lemmas ??.

A reduced above we conclude that  $U$  is an open covering of  $\mathcal{C}$ . The functor  $\mathcal{F}$  is a “field

$$\mathcal{O}_{X,x} \rightarrow \mathcal{F}_x^{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \rightarrow \mathcal{O}_{X'}^{-1} \mathcal{O}_{X,\lambda}(\mathcal{O}_{X_n}^{\bar{v}})$$

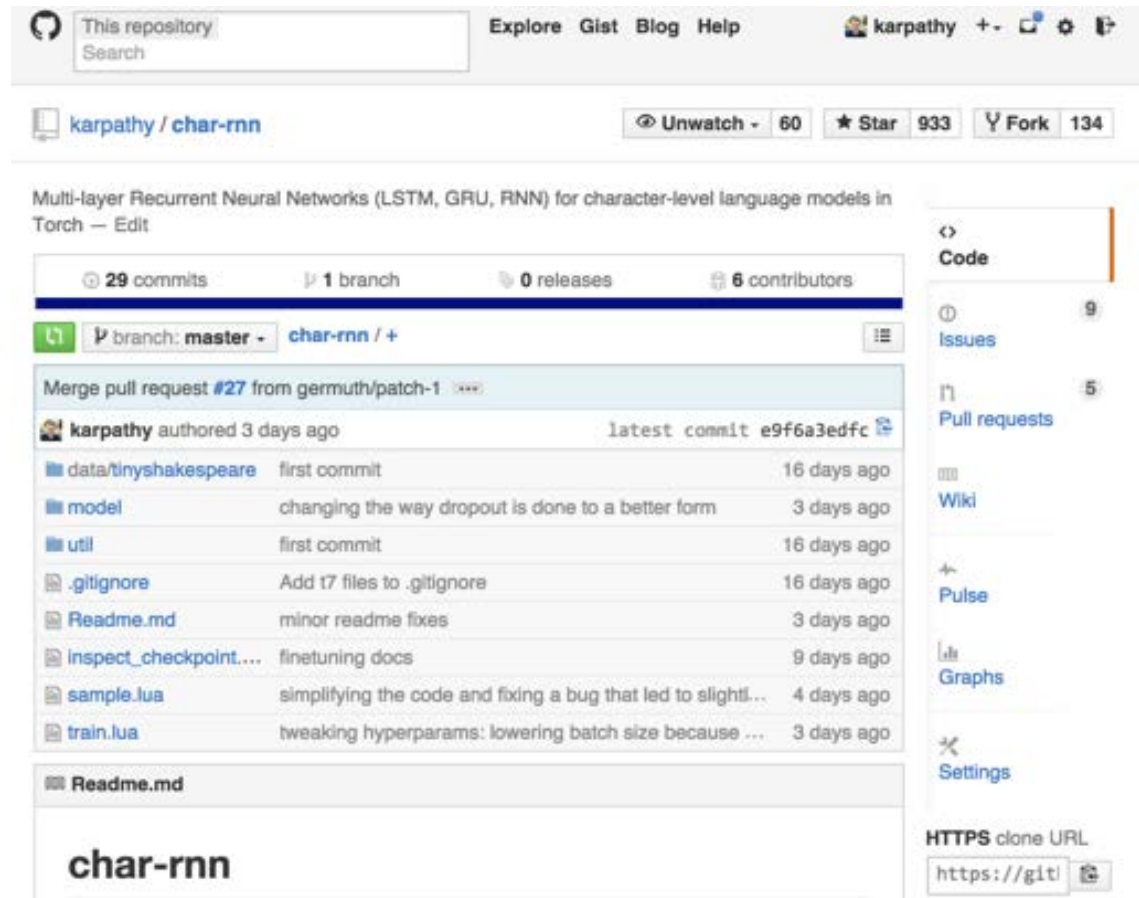
is an isomorphism of covering of  $\mathcal{O}_{X_i}$ . If  $\mathcal{F}$  is the unique element of  $\mathcal{F}$  such that  $X$  is an isomorphism.

The property  $\mathcal{F}$  is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme  $\mathcal{O}_X$ -algebra with  $\mathcal{F}$  are opens of finite type over  $S$ .

If  $\mathcal{F}$  is a scheme theoretic image points. □

If  $\mathcal{F}$  is a finite direct sum  $\mathcal{O}_{X_\lambda}$  is a closed immersion, see Lemma ?? . This is a sequence of  $\mathcal{F}$  is a similar morphism.

# Try it yourself: **char-rnn** on Github (uses Torch7)



This screenshot shows the GitHub repository page for 'karpathy / char-rnn'. The repository is described as 'Multi-layer Recurrent Neural Networks (LSTM, GRU, RNN) for character-level language models in Torch'. It has 29 commits, 1 branch, 0 releases, and 6 contributors. The latest commit is 'e9f6a3edfc' by 'karpathy' 3 days ago. The repository includes files such as 'data/tinyshakespeare', 'model', 'util', '.gitignore', 'Readme.md', 'inspect\_checkpoint...', 'sample.lua', and 'train.lua'. The README file is titled 'char-rnn'. The right sidebar shows navigation options like Code, Issues (9), Pull requests (5), Wiki, Pulse, Graphs, and Settings. The HTTPS clone URL is 'https://github.com/karpathy/char-rnn.git'.

This repository  
Search

Explore Gist Blog Help karpathy +- ⚙️ 📄

karpathy / char-rnn Unwatch - 60 ★ Star 933 🍴 Fork 134

Multi-layer Recurrent Neural Networks (LSTM, GRU, RNN) for character-level language models in Torch — Edit

🔗 29 commits 🌿 1 branch 📦 0 releases 👤 6 contributors

🌿 branch: master - char-rnn / +

Merge pull request #27 from germuth/patch-1

karpathy authored 3 days ago latest commit e9f6a3edfc

data/tinyshakespeare	first commit	16 days ago
model	changing the way dropout is done to a better form	3 days ago
util	first commit	16 days ago
.gitignore	Add 17 files to .gitignore	16 days ago
Readme.md	minor readme fixes	3 days ago
inspect_checkpoint...	finetuning docs	9 days ago
sample.lua	simplifying the code and fixing a bug that led to slightl...	4 days ago
train.lua	tweaking hyperparams: lowering batch size because ...	3 days ago

📖 Readme.md

## char-rnn

Code

Issues 9

Pull requests 5

Wiki

Pulse

Graphs

Settings

HTTPS clone URL  
https://github.com/karpathy/char-rnn.git

# Cooking Recipes

Title: BASIC CHEESE WINGS:  
Categories: Desserts  
Yield: 6 Servings

3 Eggs  
2 tb Chopped fresh curry  
-or cooking spray  
1 c Water; cooked  
2 Lemons minced mushrooms  
3 oz Sweet cooked rice  
1/2 Onion; chopped  
3 c Butter, melted  
2 ts Soy sauce  
1 ts Cinnamon  
2 md Sugar or food coloring;  
-stems cored bowl  
2 tb Salt and freshly grated  
1/4 ts Ground ginger  
1/2 c Flour  
1 tb Water; fresh parsley  
1 c Water (or or)  
1 Clove garlic, minced

Preheat oven to 350F. Combine sugar, salt, baking soda, celery and sugar. Add the chicken broth well. Add the cornstarch to the pan; cool. Add the olive oil, oil, and basil or cooking spray. Pour the onions until melted.

# Obama Speeches

*Good afternoon. God bless you.*

*The United States will step up to the cost of a new challenges of the American people that will share the fact that we created the problem. They were attacked and so that they have to say that all the task of the final days of war that I will not be able to get this done. The promise of the men and women who were still going to take out the fact that the American people have fought to make sure that they have to be able to protect our part. It was a chance to stand together to completely look for the commitment to borrow from the American people. And the fact is the men and women in uniform and the millions of our country with the law system that we should be a strong stretchs of the forces that we can afford to increase our spirit of the American people and the leadership of our country who are on the Internet of American lives.*

*Thank you very much. God bless you, and God bless the United States of America.*



**RNN Bible**  
@RNN\_Bible  
Random bible verses generated using Recurrent Neural Networks (char-rnn).

[Tweet to RNN Bible](#)

3 Followers you know



TWEETS 80 FOLLOWING 1 FOLLOWERS 51

**Tweets** [Tweets & replies](#)

**RNN Bible** @RNN\_Bible · 3h  
32:22 And they shall be the children of Israel, and they that shall come upon us, that they may be their God.

**RNN Bible** @RNN\_Bible · 7h  
2:11 Therefore shall they see thy chastisement for them, they shall live: I will sing praise to thee in the night thy servant.

**RNN Bible** @RNN\_Bible · 11h  
8:26 And they set the book of the law which Michal the Baptist came near to Man.



This repository Search Explore Gist Blog Help karpathy +- [notifications] [settings] [logout]

torvalds / linux Watch - 3,711 Star 23,054 Fork 9,141

Linux kernel source tree

520,037 commits 1 branch 420 releases 5,039 contributors

branch: master - linux / +

Merge branch 'drm-fixes' of git://people.freedesktop.org/~airlied/linux

torvalds authored 9 hours ago latest commit 4b1786927d

Documentation	Merge git://git.kernel.org/pub/scm/linux/kernel/git/hab/target-pending	6 days ago
arch	Merge branch 'x86-urgent-for-linus' of git://git.kernel.org/pub/scm/l...	a day ago
block	block: discard bdi_unregister() in favour of bdi_destroy()	9 days ago
crypto	Merge git://git.kernel.org/pub/scm/linux/kernel/git/herbert/crypto-2.6	10 days ago
drivers	Merge branch 'drm-fixes' of git://people.freedesktop.org/~airlied/linux	9 hours ago
firmware	firmware/ihex2fw.c: restore missing default in switch statement	2 months ago
fs	vfs: read file_handle only once in handle_to_path	4 days ago
include	Merge branch 'perf-urgent-for-linus' of git://git.kernel.org/pub/scm/...	a day ago
init	init: fix regression by supporting devices with major:minor:offset fo...	a month ago
ipc	Merge branch 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel...	a month ago

Code

Pull requests 74

Pulse

Graphs

HTTPS clone URL

https://github.c [copy]

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

Clone in Desktop

Download ZIP

```
static void do_command(struct seq_file *m, void *v)
{
    int column = 32 << (cmd[2] & 0x80);
    if (state)
        cmd = (int)(int_state ^ (in_8(&ch->ch_flags) & Cmd) ? 2 : 1);
    else
        seq = 1;
    for (i = 0; i < 16; i++) {
        if (k & (1 << i))
            pipe = (in_use & UMXTHREAD_UNCCA) +
                ((count & 0x00000000ffffffff) & 0x0000000f) << 8;
        if (count == 0)
            sub(pid, ppc_md.kexec_handle, 0x20000000);
        pipe_set_bytes(i, 0);
    }
    /* Free our user pages pointer to place camera if all dash */
    subsystem_info = &of_changes[PAGE_SIZE];
    rek_controls(offset, idx, &soffset);
    /* Now we want to deliberately put it to device */
    control_check_polarity(&context, val, 0);
    for (i = 0; i < COUNTER; i++)
        seq_puts(s, "policy ");
}
```

# Learning from Linux Source Code

```

static void do_command(struct seq_file *m, void *v)
{
    int column = 32 << (cmd[2] & 0x80);
    if (state)
        cmd = (int)(int_state ^ (in_8(&ch->ch_flags) & Cmd) ? 2 : 1);
    else
        seq = 1;
    for (i = 0; i < 16; i++) {
        if (k & (1 << i))
            pipe = (in_use & UMXTHREAD_UNCCA) +
                ((count & 0x00000000ffffffff8) & 0x0000000f) << 8;
        if (count == 0)
            sub(pid, ppc_md.kexec_handle, 0x20000000);
        pipe_set_bytes(i, 0);
    }
    /* Free our user pages pointer to place camera if all dash */
    subsystem_info = &of_changes[PAGE_SIZE];
    rek_controls(offset, idx, &soffset);
    /* Now we want to deliberately put it to device */
    control_check_polarity(&context, val, 0);
    for (i = 0; i < COUNTER; i++)
        seq_puts(s, "policy ");
}

```



Linus Torvalds

Shared publicly - May 24, 2015

I'm not a fan of traditional AI (rule building and LISP/prolog etc), but am just waiting for neural nets to take over.

This is a great example, and training on the Linux kernel sources just explains so much.

Sigh.

David Alan Gilbert originally shared:

This is wonderful; neural network setup producing wonderful results; one example is training it on

[Read more](#)





```
/*
 * Copyright (c) 2006-2010, Intel Mobile Communications. All rights reserved.
 *
 * This program is free software; you can redistribute it and/or modify it
 * under the terms of the GNU General Public License version 2 as published by
 * the Free Software Foundation.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 *
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software Foundation,
 * Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
 */

#include <linux/kexec.h>
#include <linux/errno.h>
#include <linux/io.h>
#include <linux/platform_device.h>
#include <linux/multi.h>
#include <linux/ckevent.h>

#include <asm/io.h>
#include <asm/prom.h>
#include <asm/e820.h>
#include <asm/system_info.h>
#include <asm/setew.h>
#include <asm/pgproto.h>
```

```
#include <asm/io.h>
#include <asm/prom.h>
#include <asm/e820.h>
#include <asm/system_info.h>
#include <asm/setew.h>
#include <asm/pgproto.h>

#define REG_PG    vesa_slot_addr_pack
#define PFM_NOCOMP AFSR(0, load)
#define STACK_DDR(type)    (func)

#define SWAP_ALLOCATE(nr)    (e)
#define emulate_sigs()    arch_get_unaligned_child()
#define access_rw(TST)    asm volatile("movd %esp, %0, %3" : : "r" (0)); \
    if (__type & DO_READ)

static void stat_PC_SEC __read_mostly offsetof(struct seq_argsqueue, \
    pC>[1]);

static void
os_prefix(unsigned long sys)
{
#ifdef CONFIG_PREEMPT
    PUT_PARAM_RAID(2, sel) = get_state_state();
    set_pid_sum((unsigned long)state, current_state_str(),
        (unsigned long)-1->lr_full; low;
}
}
```

# Yoav Goldberg n-gram experiments

Order 10 ngram model on Shakespeare:

APEMANTUS:

Ho, ho! I laugh to see your beard!

BOYET:

Madam, in great extremes of passion as she  
discovers it.

PAROLLES:

By my white head and her wit  
Values itself: to the sepulchre!  
With this, my lord,  
That I have some business: let's away.

First Keeper:

Forbear to murder: and wilt thou not say he lies,  
And lies, and let the devil would have said, sir, their speed  
Hath been balm to heal their woes,

But on  
Linux:

```

~/~/*
 * linux/kernel/time.c
 * Please report this on hardware.
 */
void irq_mark_irq(unsigned long old_entries, eval);

        /*
         * Divide only 1000 for ns^2 -> us^2 conversion values
don't overflow:
        seq_puts(m, "\ttramp: %pS",
                (void *)class->contending_point
]++;
        if (likely(t->flags & WQ_UNBOUND)) {
            /*
             * Update inode information. If the
             * slowpath and sleep time (abs or rel)
 * @rmtp: remaining (either due
 * to consume the state of ring buffer size. */
            header_size - size, in bytes, of the chain.
            */
            BUG_ON(!error);
        } while (cgrp) {
            if (old) {
                if (kdb_continue_catastrophic;

#endif

```

# Visualizing and Understanding Recurrent Networks

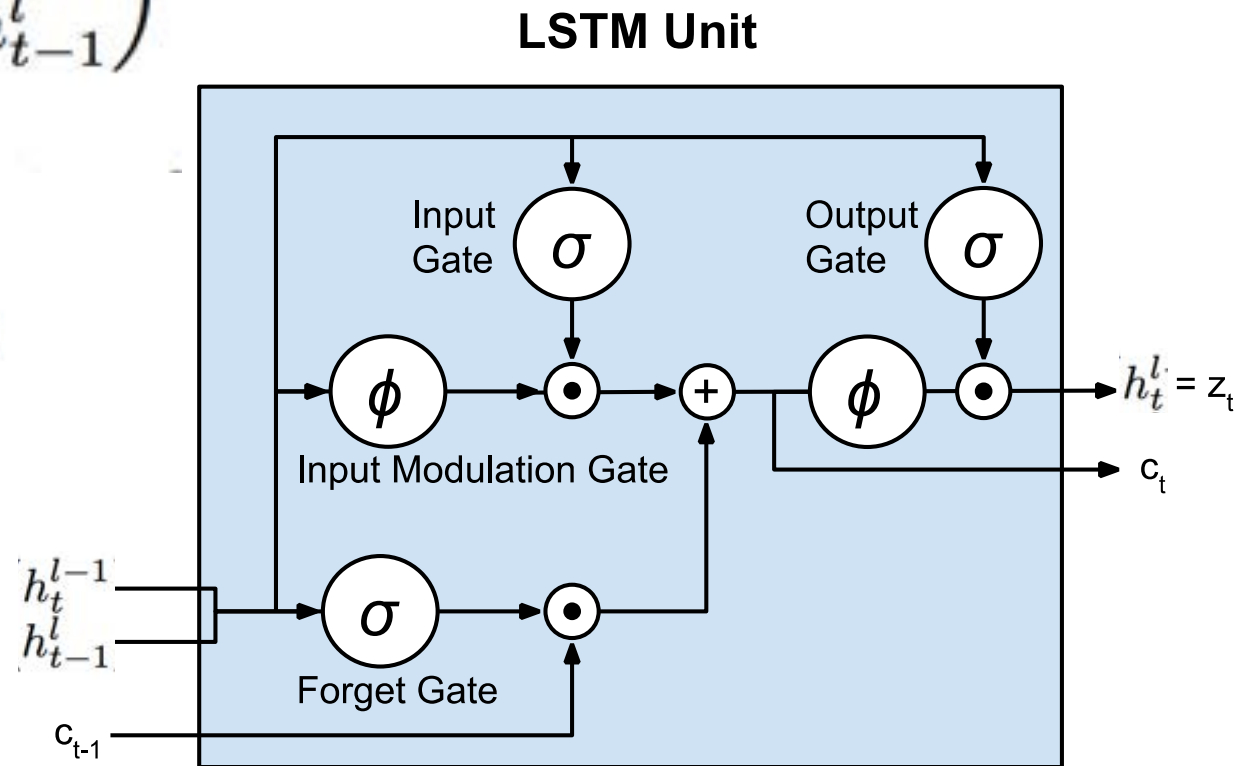
Andrej Karpathy\*, Justin Johnson\*, Li Fei-Fei

(on [arXiv.org](https://arxiv.org) as of June 2015)

# Hunting interpretable cells

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \text{tanh} \end{pmatrix} W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g$$
$$h_t^l = o \odot \tanh(c_t^l)$$



# Hunting interpretable cells

```
/* Unpack a filter field's string representation from user-space
 * buffer. */
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
    char *str;
    if (!*bufp || (len == 0) || (len > *remain))
        return ERR_PTR(-EINVAL);
    /* Of the currently implemented string fields, PATH_MAX
     * defines the longest valid length.
     */
}
```

# Hunting interpretable cells

"You mean to imply that I have nothing to eat out of.... On the contrary, I can supply you with everything even if you want to give dinner parties," warmly replied Chichagov, who tried by every word he spoke to prove his own rectitude and therefore imagined Kutuzov to be animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating smile: "I meant merely to say what I said."

quote detection cell



# Hunting interpretable cells

Cell sensitive to position in line:

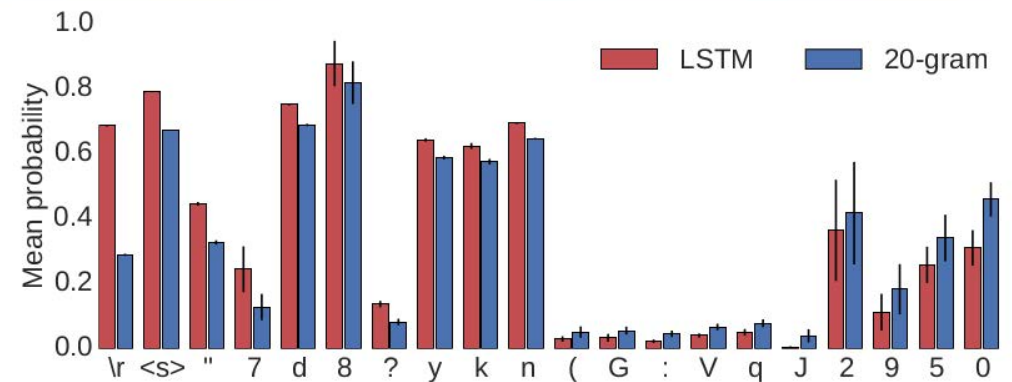
The sole importance of the crossing of the Berezina lies in the fact that it plainly and indubitably proved the fallacy of all the plans for cutting off the enemy's retreat and the soundness of the only possible line of action--the one Kutuzov and the general mass of the army demanded--namely, simply to follow the enemy up. The French crowd fled at a continually increasing speed and all its energy was directed to reaching its goal. It fled like a wounded animal and it was impossible to block its path. This was shown not so much by the arrangements it made for crossing as by what took place at the bridges. When the bridges broke down, unarmed soldiers, people from Moscow and women with children who were with the French transport, all--carried on by vis inertiae--pressed forward into boats and into the ice-covered water and did not, surrender.

line length tracking cell

# Hunting interpretable cells

Cell sensitive to position in line:

The sole importance of the crossing of the Berezina lies in the fact that it plainly and indubitably proved the fallacy of all the plans for cutting off the enemy's retreat and the soundness of the only possible line of action--the one Kutuzov and the general mass of the army demanded--namely, simply to follow the enemy up. The French crowd fled at a continually increasing speed and all its energy was directed to reaching its goal. It fled like a wounded animal and it was impossible to block its path. This was shown not so much by the arrangements it made for crossing as by what took place at the bridges. When the bridges broke down, unarmed soldiers, people from Moscow and women with children who were with the French transport, all--carried on by vis inertiae--pressed forward into boats and into the ice-covered water and did not, surrender.



# Hunting interpretable cells

```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
    siginfo_t *info)
{
    int sig = next_signal(pending, mask);
    if (sig) {
        if (current->notifier) {
            if (sigismember(current->notifier_mask, sig)) {
                if (!(current->notifier)(current->notifier_data)) {
                    clear_thread_flag(TIF_SIGPENDING);
                    return 0;
                }
            }
        }
        collect_signal(sig, pending, info);
    }
    return sig;
}
```

if statement cell

# Hunting interpretable cells

```
/* Duplicate LSM field information. The lsm_rule is opaque, so
 * re-initialized. */
static inline int audit_dupe_lsm_field(struct audit_field *df,
                                     struct audit_field *sf)
{
    int ret = 0;
    char *lsm_str;
    /* our own copy of lsm_str */
    lsm_str = kstrdup(sf->lsm_str, GFP_KERNEL);
    if (unlikely(!lsm_str))
        return -ENOMEM;
    df->lsm_str = lsm_str;
    /* our own (refreshed) copy of lsm_rule */
    ret = security_audit_rule_init(df->type, df->op, df->lsm_str,
                                  (void *)&df->lsm_rule);
    /* Keep currently invalid fields around in case they
     * become valid after a policy reload. */
    if (ret == -EINVAL) {
        pr_warn("audit rule for LSM '%s' is invalid\n",
              df->lsm_str);
        ret = 0;
    }
    return ret;
}
```

quote/comment cell



# Hunting interpretable cells

```
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
    int i;
    if (classes[class]) {
        for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
            if (mask[i] & classes[class][i])
                return 0;
    }
    return 1;
}
```

code depth cell

# Hunting interpretable cells

```
char *audit_unpack_string(void **bufp, size_t *remain, si
{
    char *str;
    if (!*bufp || (len == 0) || (len > *remain))
        return ERR_PTR(-EINVAL);
    /* of the currently implemented string fields, PATH_MAX
     * defines the longest valid length.
     */
    if (len > PATH_MAX)
        return ERR_PTR(-ENAMETOOLONG);
    str = kmalloc(len + 1, GFP_KERNEL);
    if (unlikely(!str))
        return ERR_PTR(-ENOMEM);
    memcpy(str, *bufp, len);
    str[len] = 0;
    *bufp += len;
    *remain -= len;
    return str;
}
```

something interesting cell  
(not quite sure what)

# What's Wrong With Deep Learning?

**Yann LeCun**

Facebook AI Research &  
Center for Data Science, NYU

[yann@cs.nyu.edu](mailto:yann@cs.nyu.edu)

<http://yann.lecun.com>





# Missing Some Theory



## Why are ConvNets a good architecture?

- Scattering transform
- Mark Tygert's "complex ConvNet"

## How many layers do we really need?

- Really?

## How many effective free parameters are there in a large ConvNet

- The weights seem to be awfully redundant

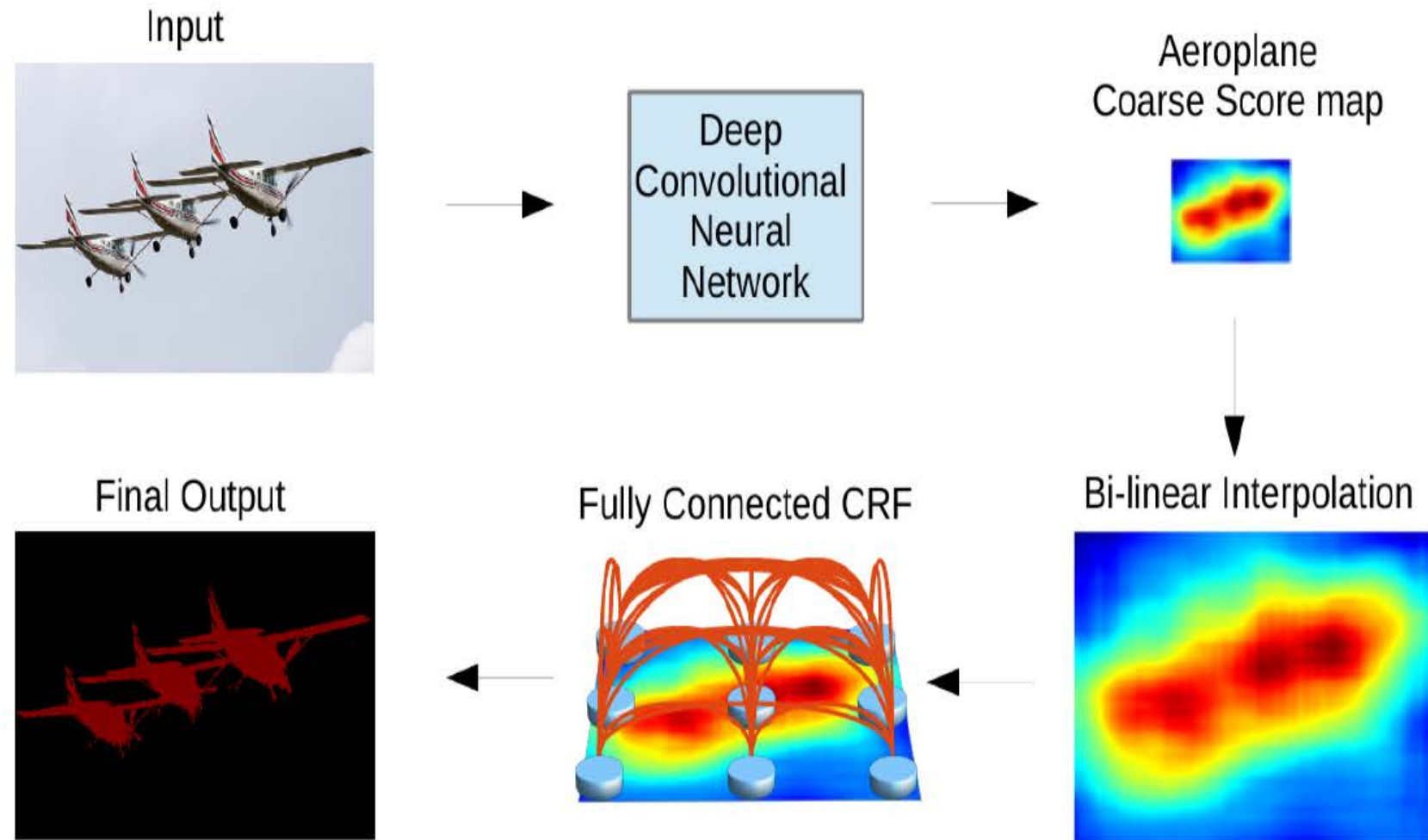
## What about Local Minima?

- Turns out almost all the local minima are equivalent
- Local minima are degenerate (very flat in most directions)
- Random matrix / spin glass theory comes to the rescue
- [Choromanska, Henaff, Mathieu, Ben Arous, LeCun AI-stats 2015]

# Missing: Reasoning



# System outline



J. Long, E. Shelhamer, T. Darrell, FCNNs for Semantic Segmentation, CVPR 15

P. Krähenbühl and V. Koltun, Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials, NIPS 2011

# Reasoning as Energy Minimization (structured prediction++)

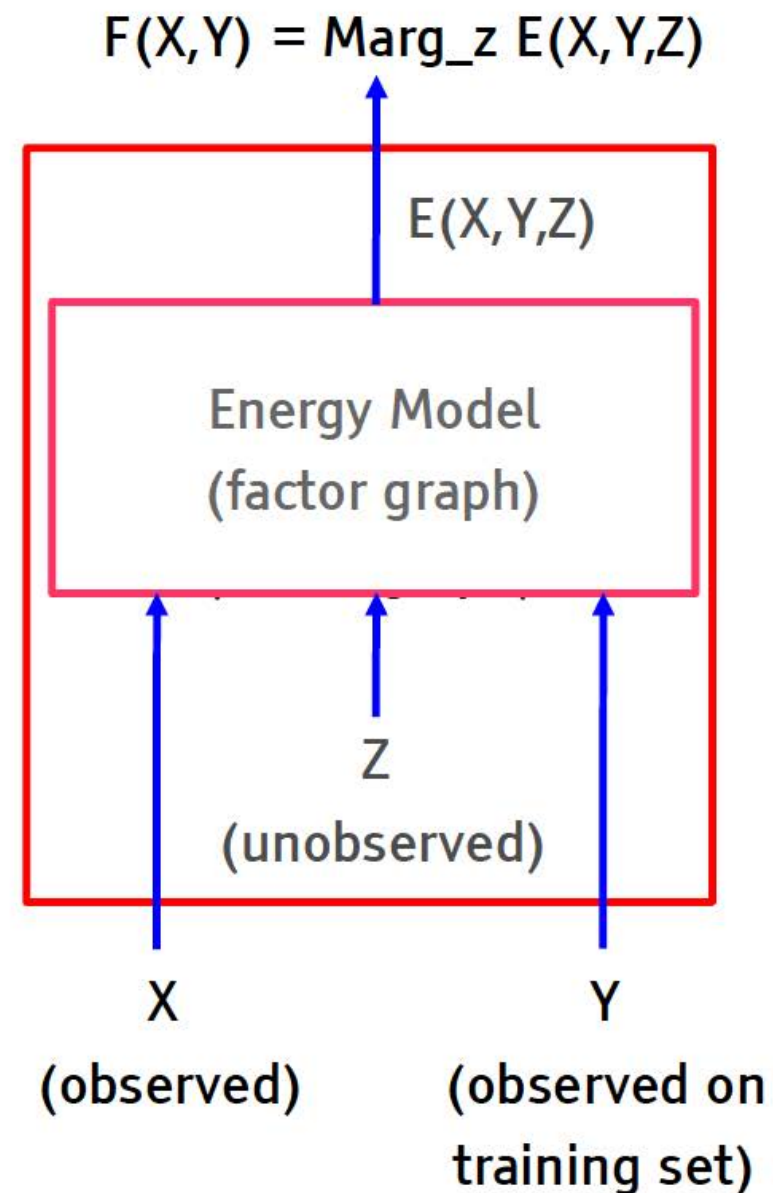
Y LeCun

## Deep Learning systems can be assembled into energy models AKA factor graphs

- ▶ Energy function is a sum of factors
- ▶ Factors can embed whole deep learning systems
- ▶ X: observed variables (inputs)
- ▶ Z: never observed (latent variables)
- ▶ Y: observed on training set (output variables)

## Inference is energy minimization (MAP) or free energy minimization (marginalization) over Z and Y given an X

- ▶  $F(X,Y) = \text{MIN}_z E(X,Y,Z)$
- ▶  $F(X,Y) = -\log \text{SUM}_z \exp[-E(X,Y,Z) ]$





# Missing: Unsupervised Learning

# Autoencoder

- An autoencoder takes an input  $\mathbf{x} \in [0, 1]^d$  and first maps it (with an encoder) to a hidden representation  $\mathbf{y} \in [0, 1]^{d'}$  through a deterministic mapping

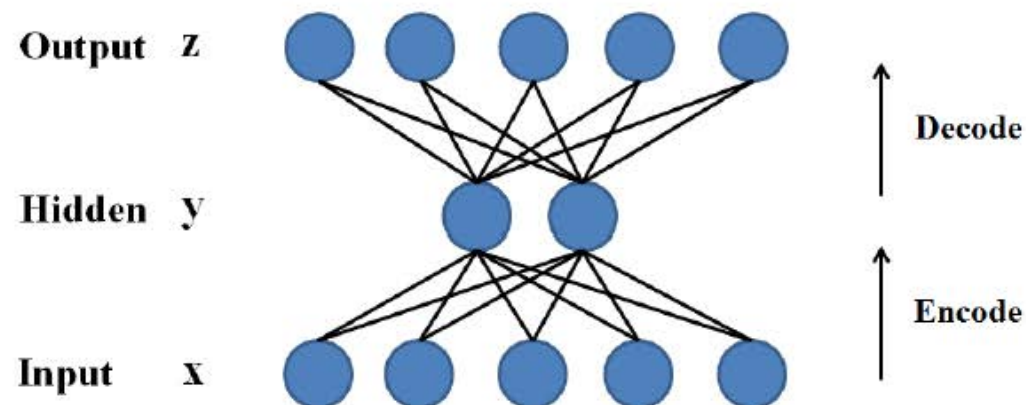
$$\mathbf{y} = s(\mathbf{W}\mathbf{x} + \mathbf{b})$$

where  $s$  is a non-linear activation function (such as sigmoid).

- $\mathbf{y}$  is mapped back (with a decoder) into a reconstruction  $\mathbf{z}$  of the same shape as  $\mathbf{x}$ ,

$$\mathbf{z} = s(\mathbf{W}'\mathbf{y} + \mathbf{b}')$$

$\mathbf{z}$  is seen as a prediction of  $\mathbf{x}$ .



# Autoencoder

- Encoder

$$\mathbf{y} = f_{\theta}(\mathbf{x})$$

- Decoder

$$\mathbf{z} = g_{\theta}(\mathbf{y})$$

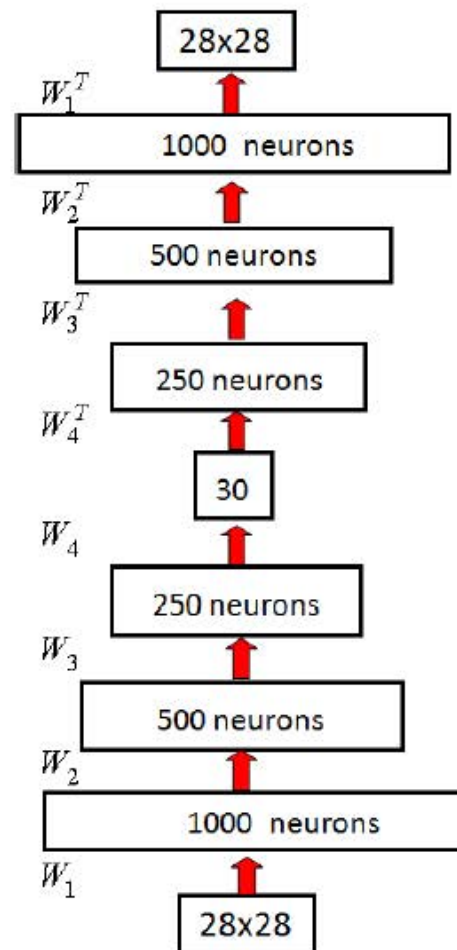
$$\theta = \{\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{b}'\}$$

- It is important to add regularization in the training criterion or the parametrization to prevent the auto-encoder from learning the identity function, which would lead zero reconstruction error everywhere
- A particular form of regularization consists in constraining the code to have a low dimension, and this is what the classical auto-encoder or PCA do.



# Deep autoencoder

- Stack multiple encoders (and their corresponding decoders)





# Deep autoencoder

- Very difficult to optimize deep autoencoders using backpropagation
- Pre-training + fine-tuning
  - First train a stack of RBMs
  - Then “unroll” them
  - Then fine-tune with backpropagation

# Comparison of methods of compressing images

