ITCS 6156/8156 Fall 2023
Machine Learning

Recurrent Neural Networks

Instructor: Hongfei Xue
Email: hongfei.xue@charlotte.edu
Class Meeting: Mon & Wed, 4:00 PM – 5:15 PM, CHHS 376

UNIVERSITY OF NORTH CAROLINA CHARLOTTE

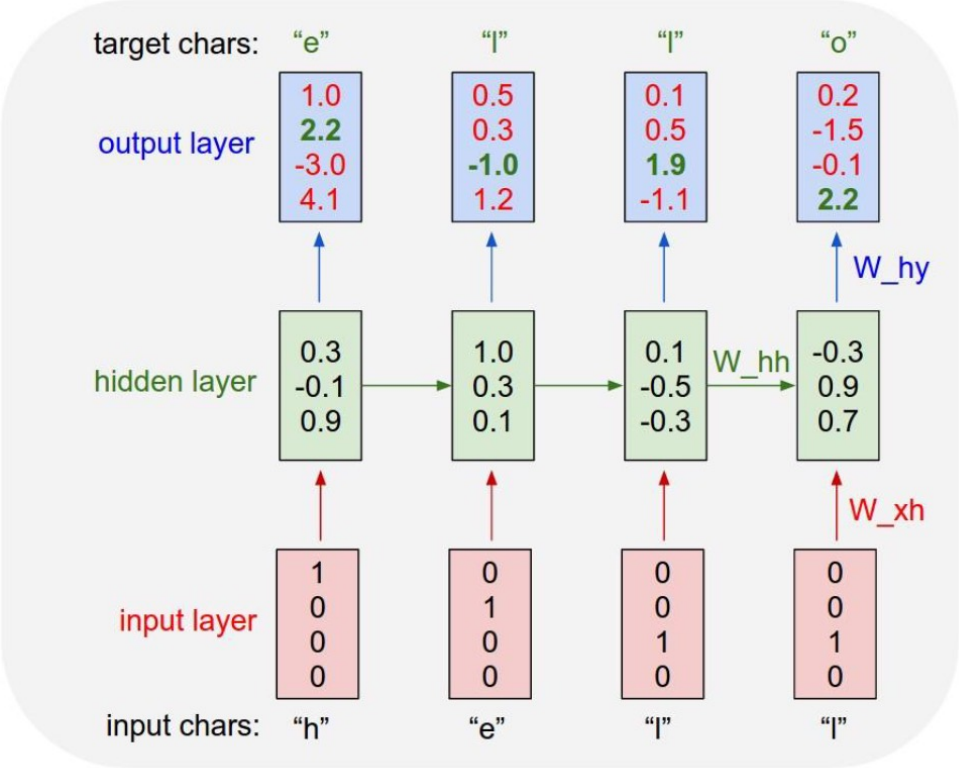
Some content in the slides is based on DeepMind's and Dr. Fei-Fei Li's lectures

Example: Character-level Language Model

Example: Character-level Language Model

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

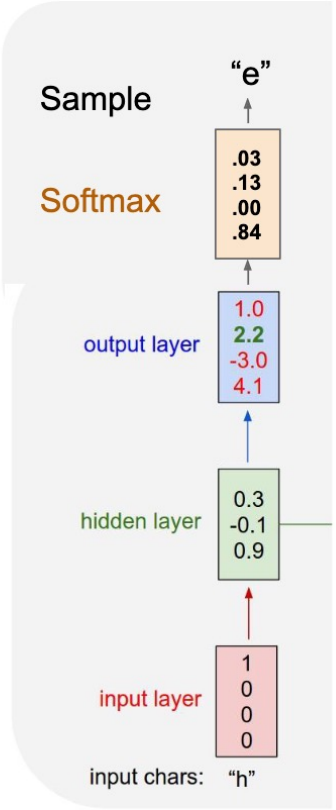


Example: Character-level Language Model

Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

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

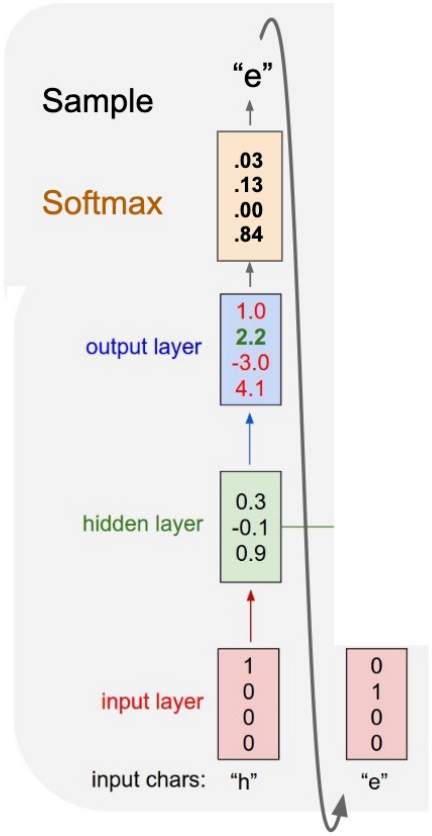


Example: Character-level Language Model

Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

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

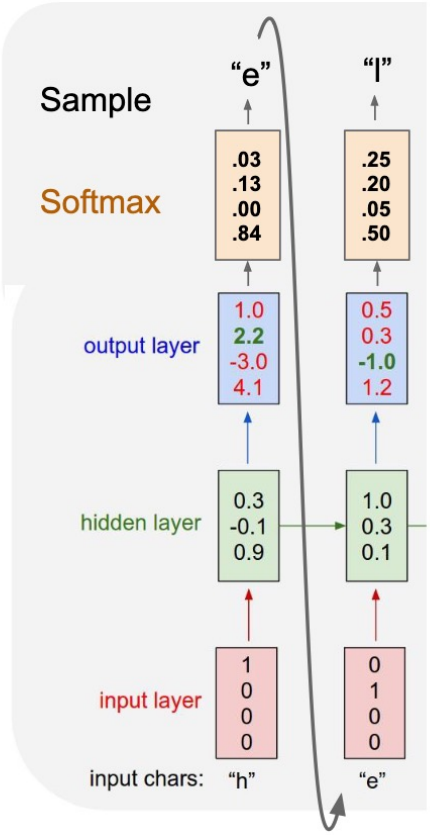


Example: Character-level Language Model

Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

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

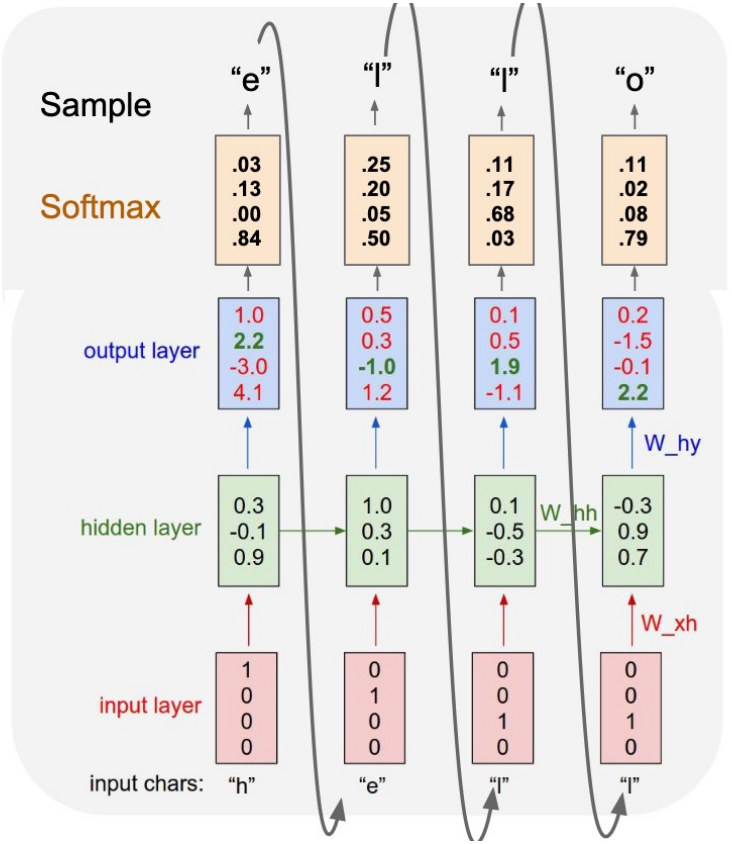


Example: Character-level Language Model

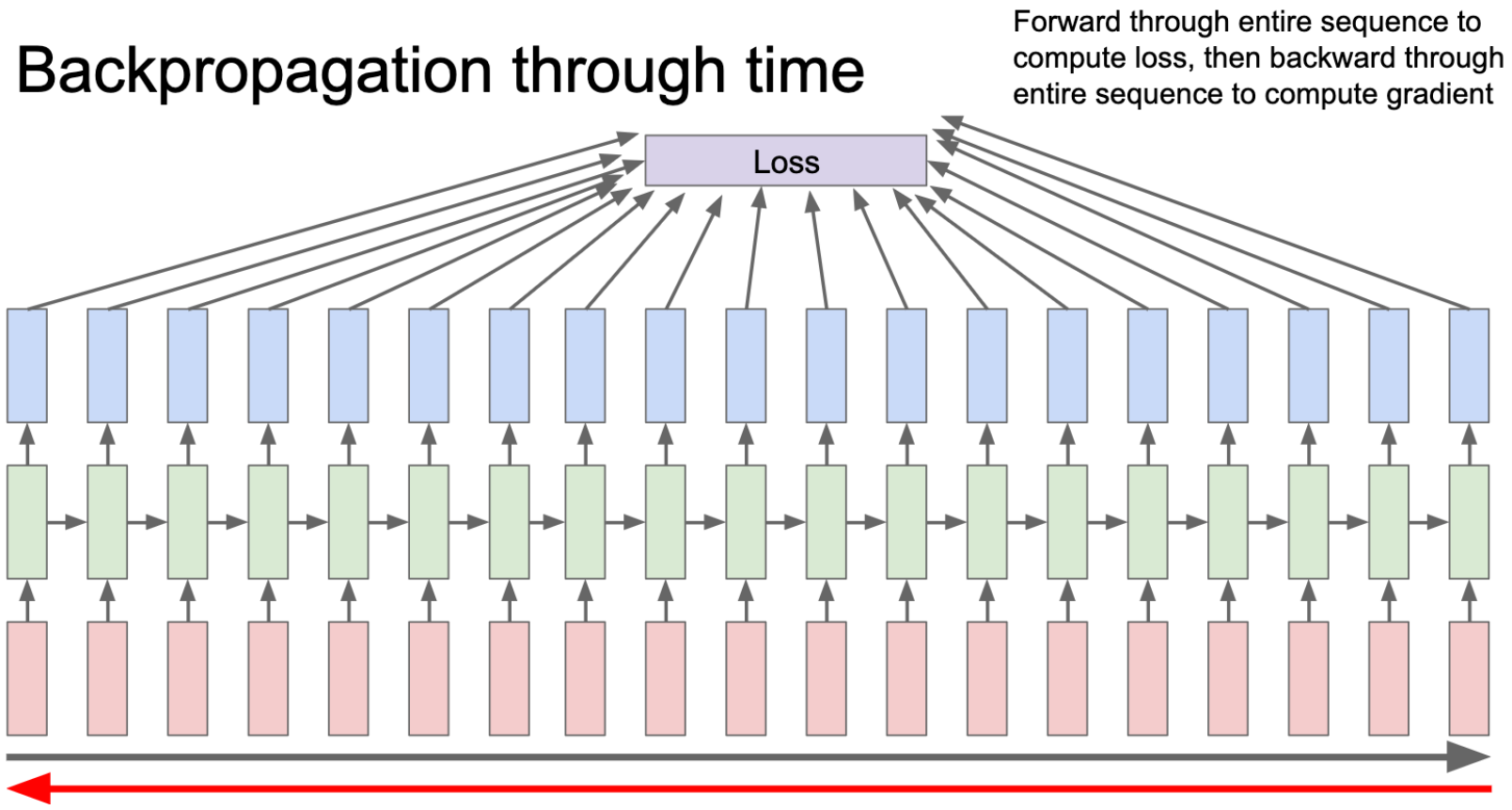
Example: Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

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

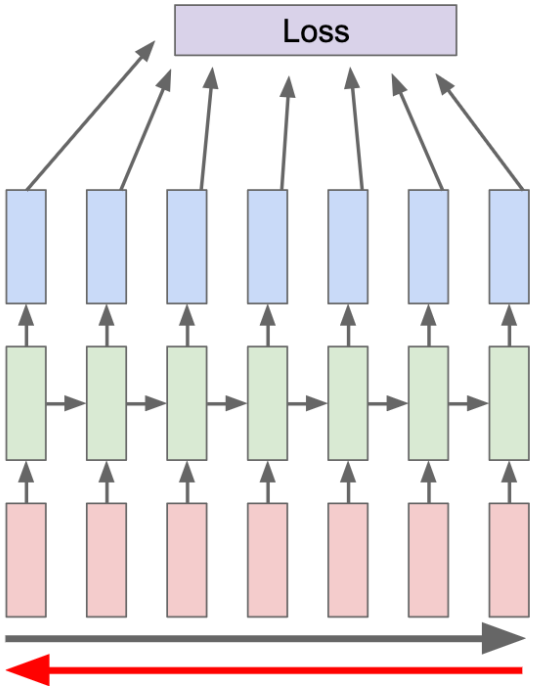


Backpropagation Through Time



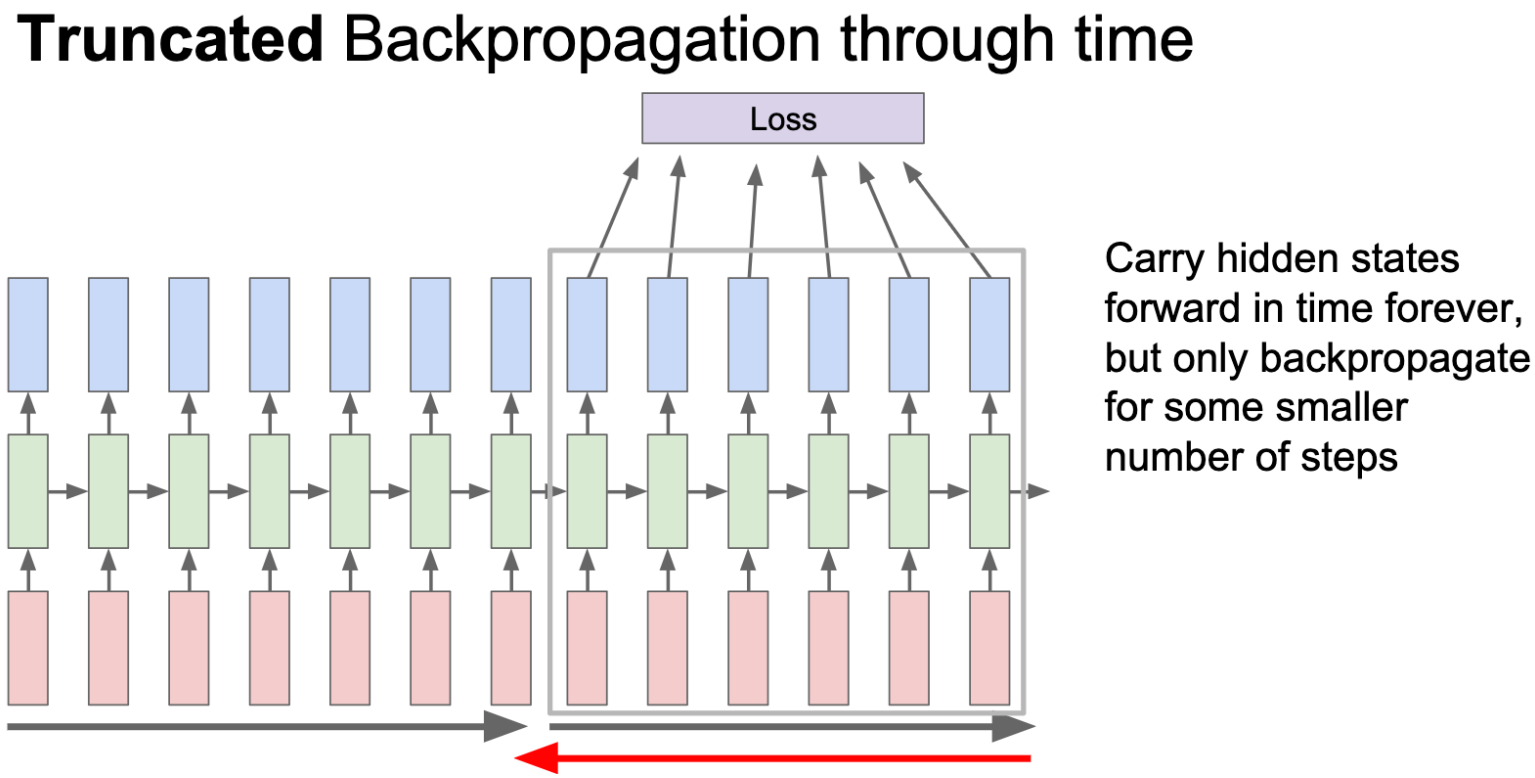
Truncated Backpropagation Through Time

Truncated Backpropagation through time

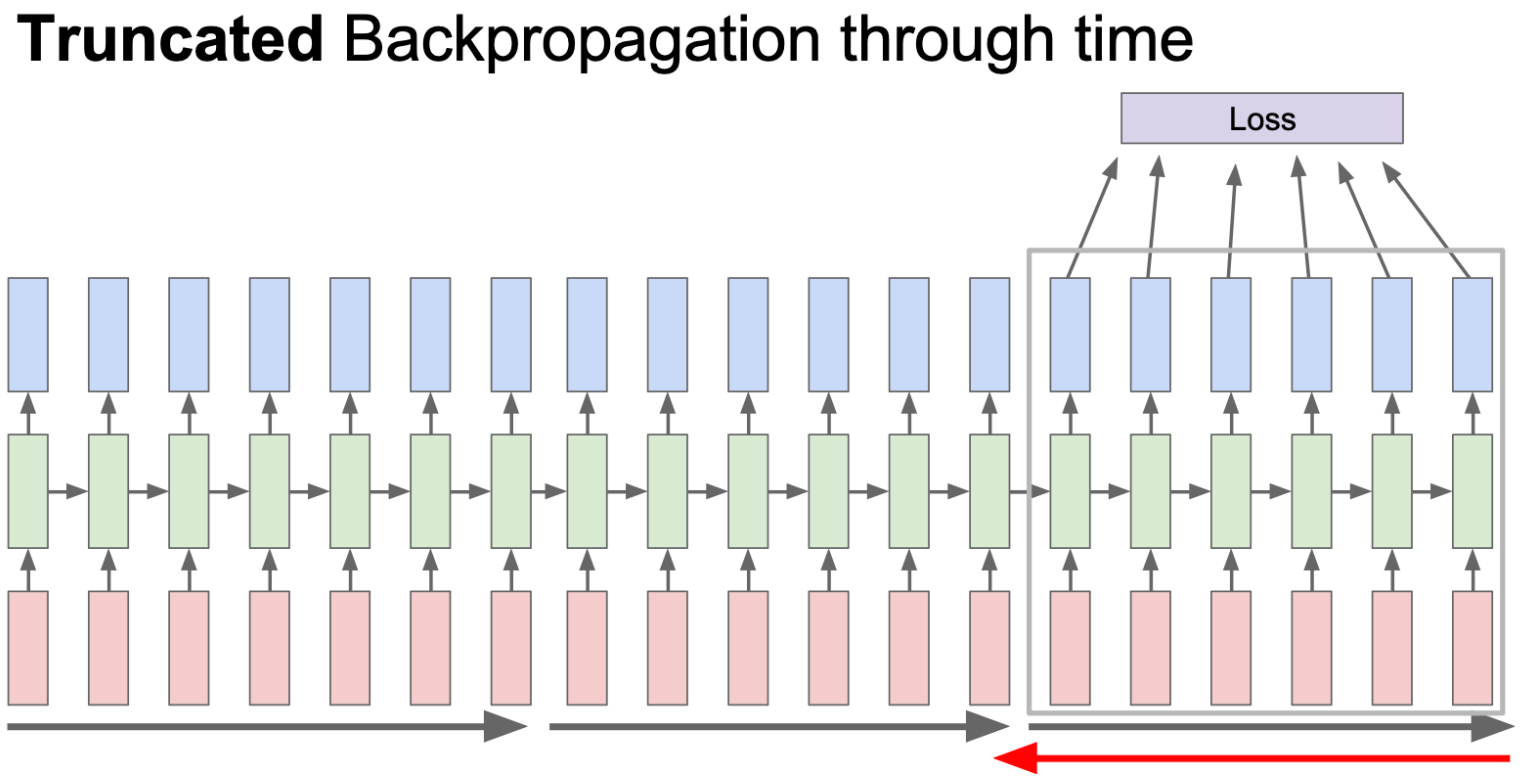


Run forward and backward through chunks of the sequence instead of whole sequence

Truncated Backpropagation Through Time

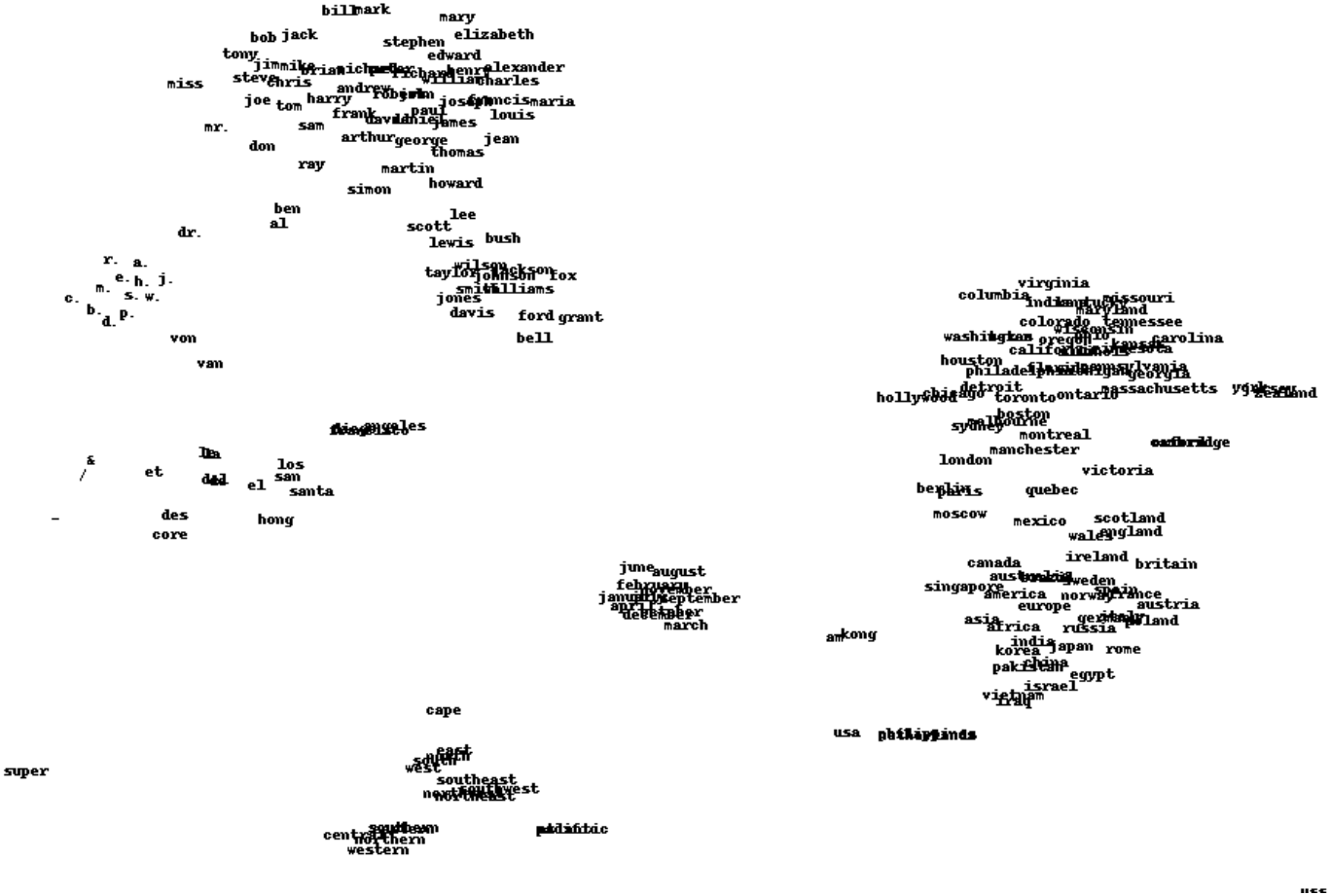


Truncated Backpropagation Through Time

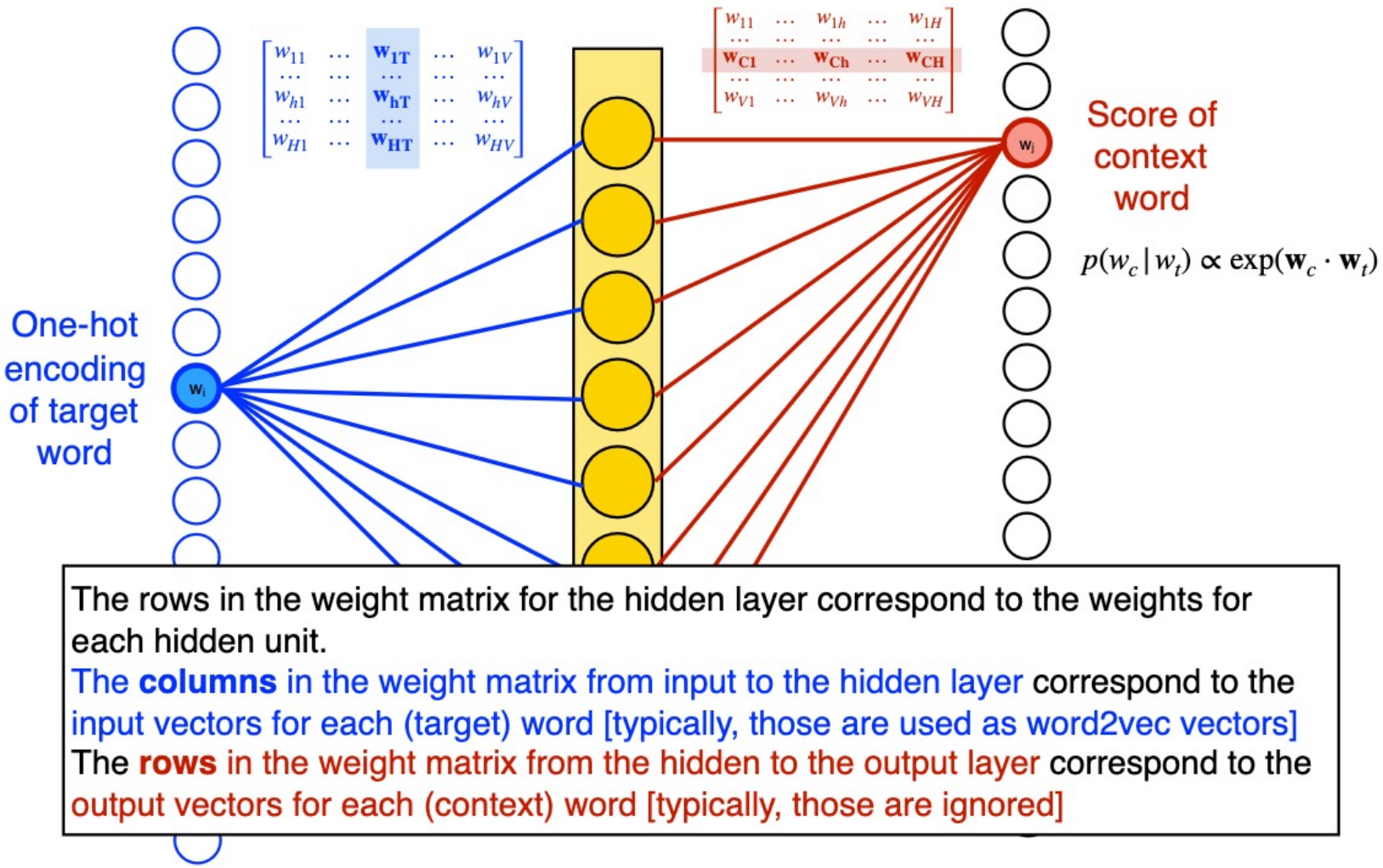


Word Embedding

- Can we represent words as vectors in space?



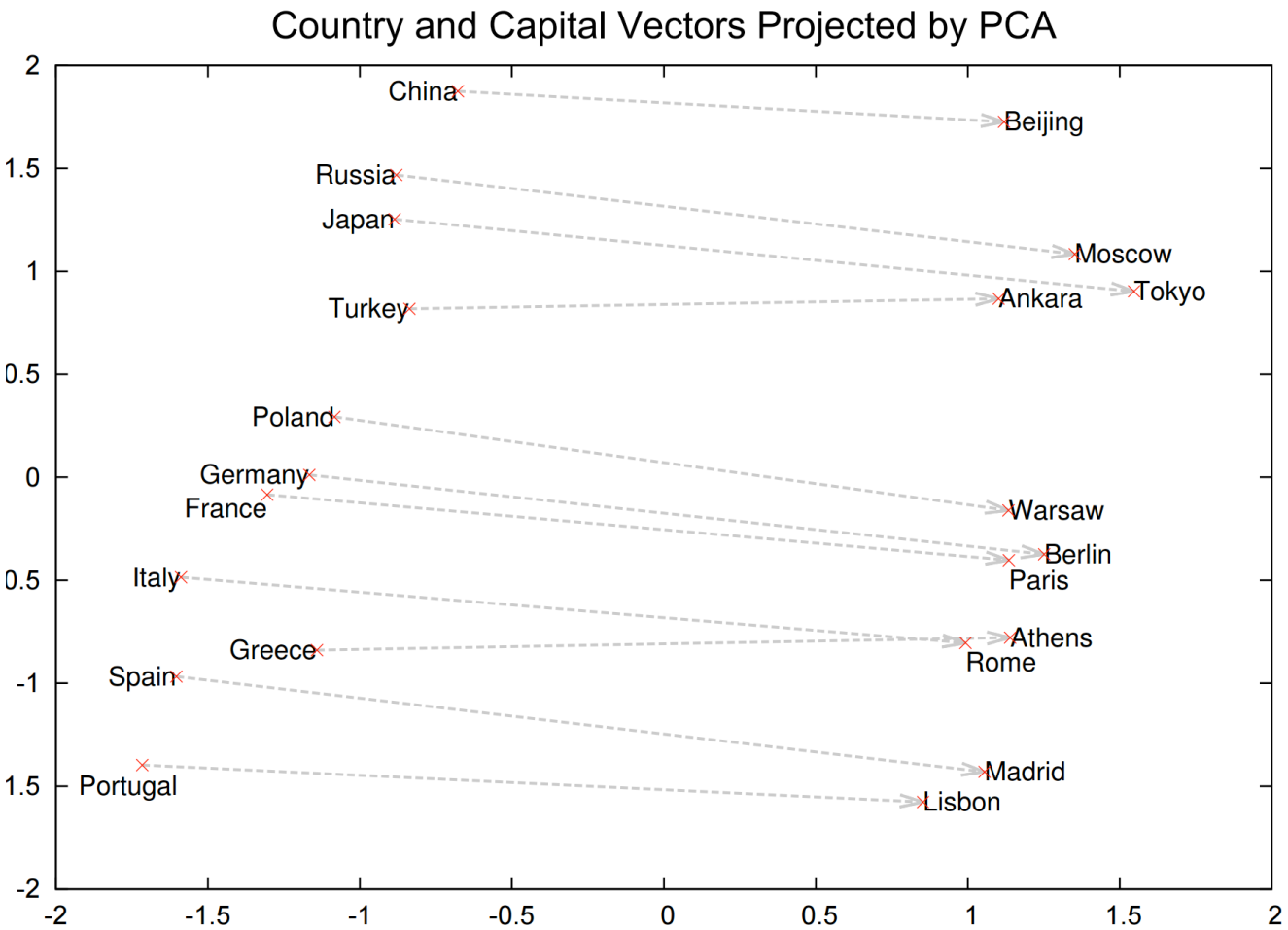
Word Embedding



The rows in the weight matrix for the hidden layer correspond to the weights for each hidden unit.
The **columns** in the weight matrix from input to the hidden layer correspond to the input vectors for each (target) word [typically, those are used as word2vec vectors]
The **rows** in the weight matrix from the hidden to the output layer correspond to the output vectors for each (context) word [typically, those are ignored]

Word Embedding

- Word Analogies:



Long-term Dependencies are Important

... Finally, Tim was planning to visit France on the final week of his journey. He was quite excited to try the local delicacies and had lots of recommendations for good restaurants and exhibitions. His first stop was, of course, the capital where he would meet his long-time Friend Jean-Pierre. In order to arrive for breakfast he took the early 5 AM train from London to ...

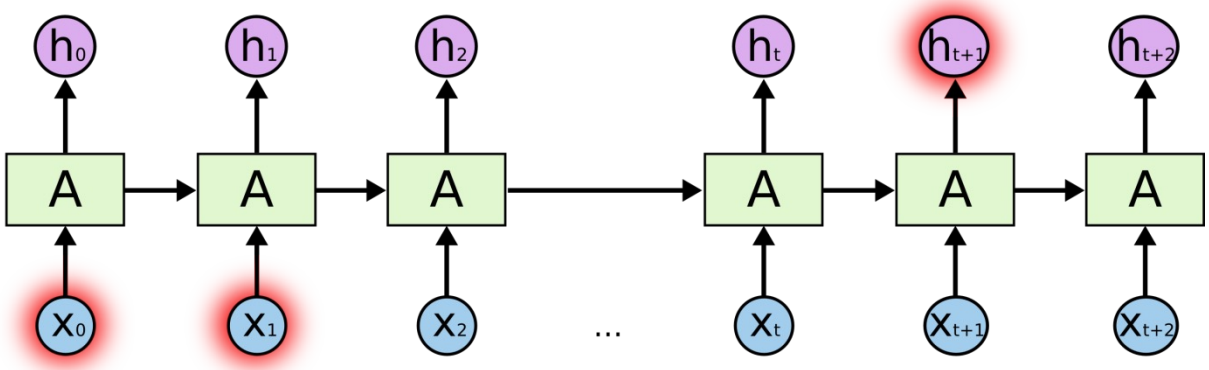
Long-term Dependencies are Important

... Finally, Tim was planning to visit **France** on the final week of his journey. He was quite excited to try the local delicacies and had lots of recommendations for good restaurants and exhibitions. His first stop was, of course, the **capital** where he would meet his long-time Friend Jean-Pierre. In order to arrive for breakfast he took the early 5 AM train from London to ...

PARIS!

Long Distance Dependencies

- It is very difficult to train RNNs to retain information over many time steps
- This make is very difficult to learn RNNs that handle long-distance dependencies, such as subject-verb agreement.

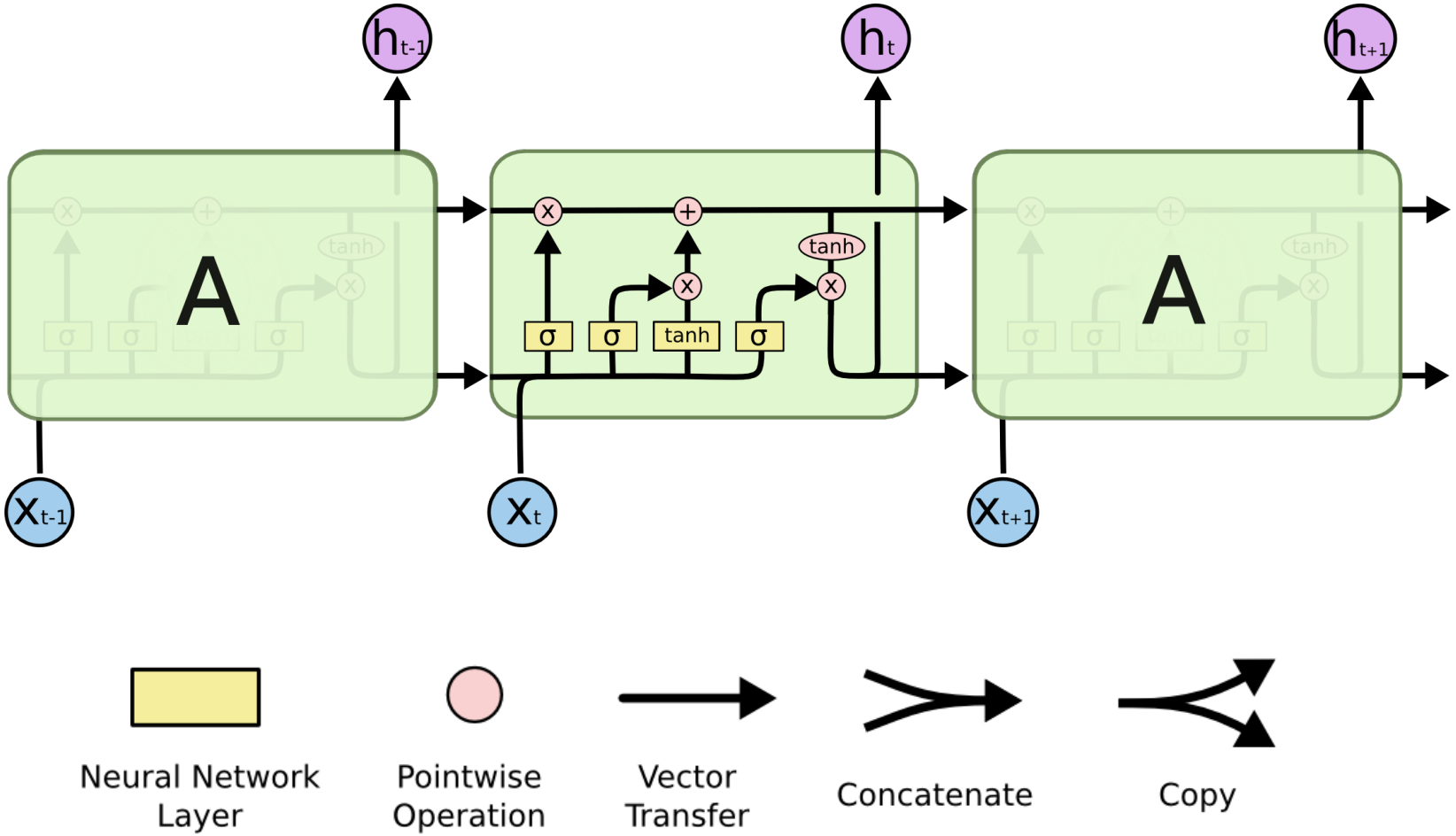


Long Short-Term Memory (LSTM) networks

- LSTM networks, add additional gating units in each memory cell.
  - Forget gate
  - Input gate
  - Output gate

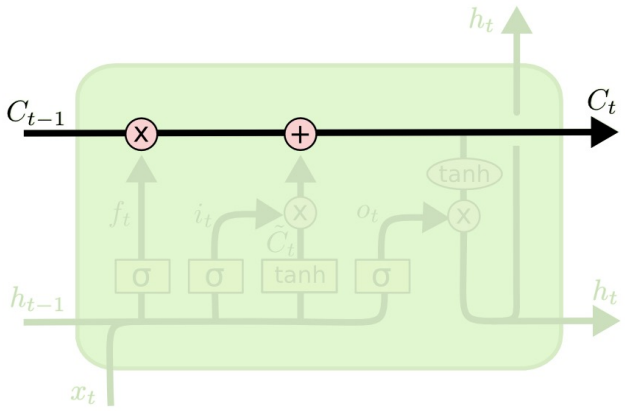
- Prevents vanishing/exploding gradient problem and allows network to retain state information over longer periods of time.

LSTM Network Architecture



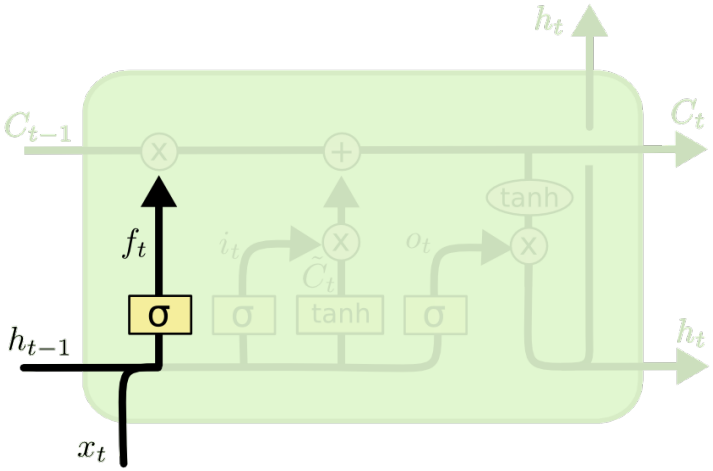
Cell State

- Maintains a vector C_t that is the same dimensionality as the hidden state, h_t
- Information can be added or deleted from this state vector via the forget and input gates.



Forget Gate

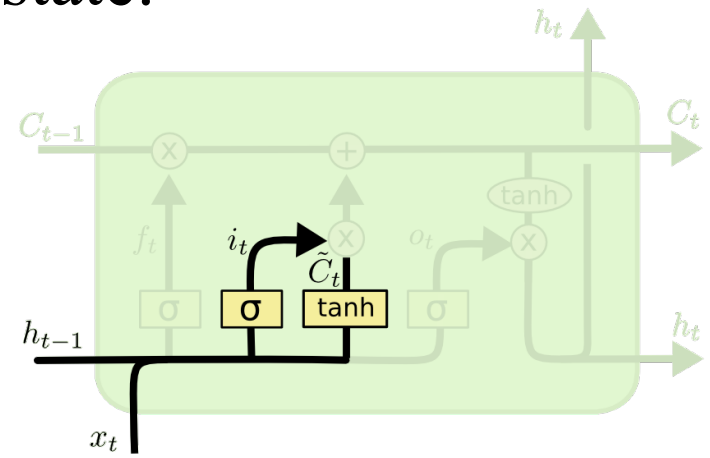
- Forget gate computes a 0-1 value using a logistic sigmoid output function from the input, x_t, and the current hidden state, h_t:
- Multiplicatively combined with cell state, "forgetting" information where the gate outputs something close to 0.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Input Gate

- First, determine which entries in the cell state to update by computing 0-1 sigmoid output.
- Then determine what amount to add/subtract from these entries by computing a tanh output (valued –1 to 1) function of the input and hidden state.

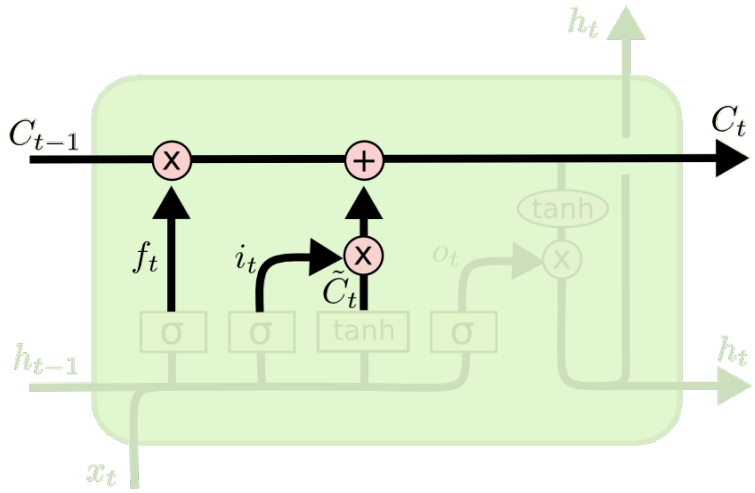


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Updating the Cell State

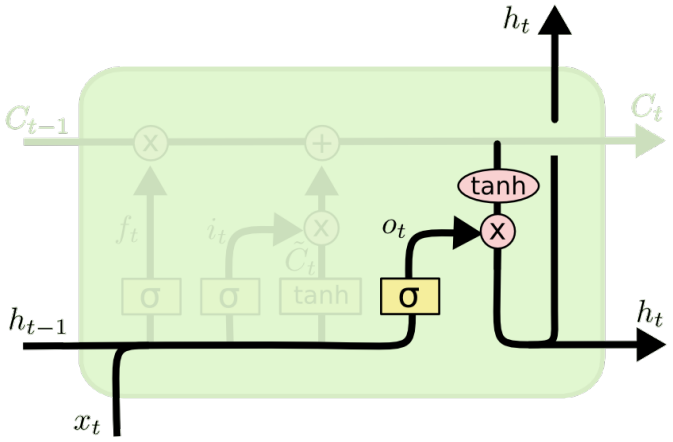
- Cell state is updated by using component-wise vector multiply to "forget" and vector addition to "input" new information.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Output Gate

- Hidden state is updated based on a "filtered" version of the cell state, scaled to –1 to 1 using tanh.
- Output gate computes a sigmoid function of the input and current hidden state to determine which elements of the cell state to "output".

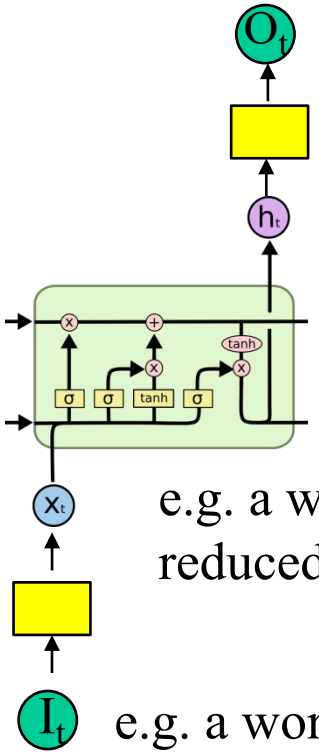


$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Overall Network Archtecture

- Single or multilayer networks can compute LSTM inputs from problem inputs and problem outputs from LSTM outputs.



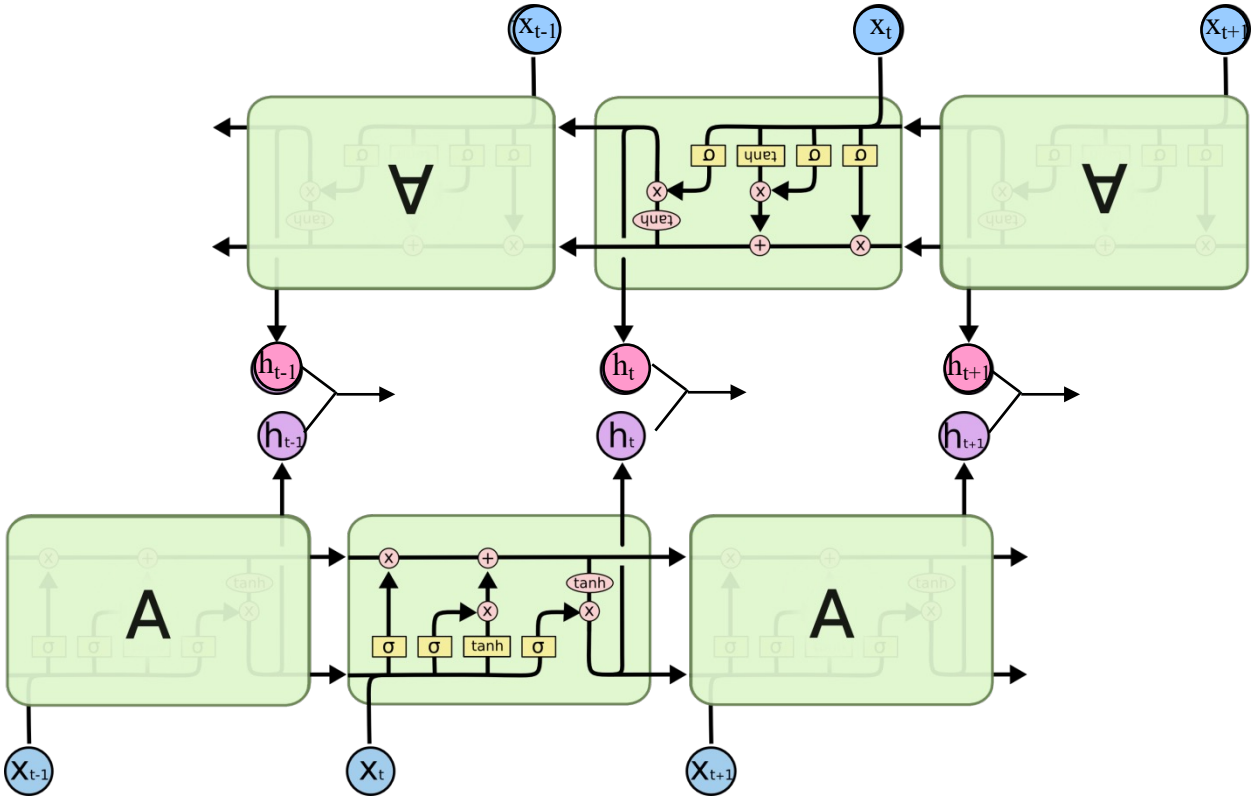
e.g. a word as a "one hot" vector

e.g. a word "embedding" with reduced dimensionality

e.g. a word as a "one hot" vector

Bi-directional LSTM (Bi-LSTM)

- Separate LSTMs process sequence forward and backward and hidden layers at each time step are concatenated to form the cell output.



Multilayer RNNs/LSTMs

Multilayer RNNs

$$h_t^l = \tanh W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$h \in \mathbb{R}^n$ $\quad W^l \ [n \times 2n]$

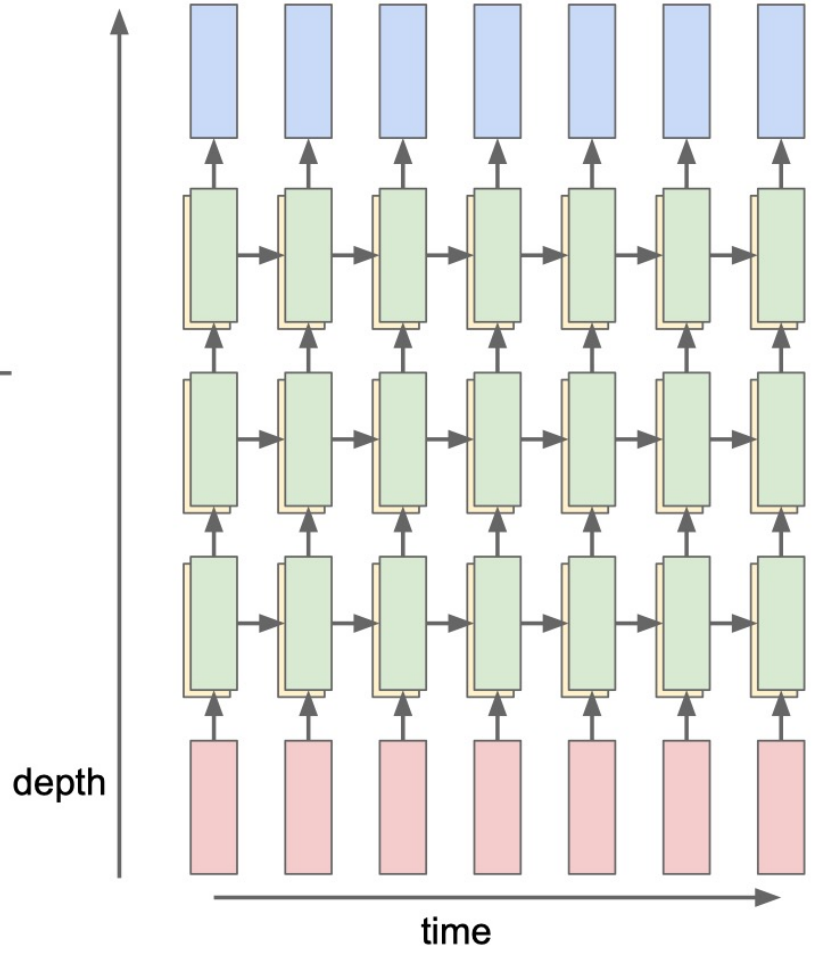
LSTM:

$W^l \ [4n \times 2n]$

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

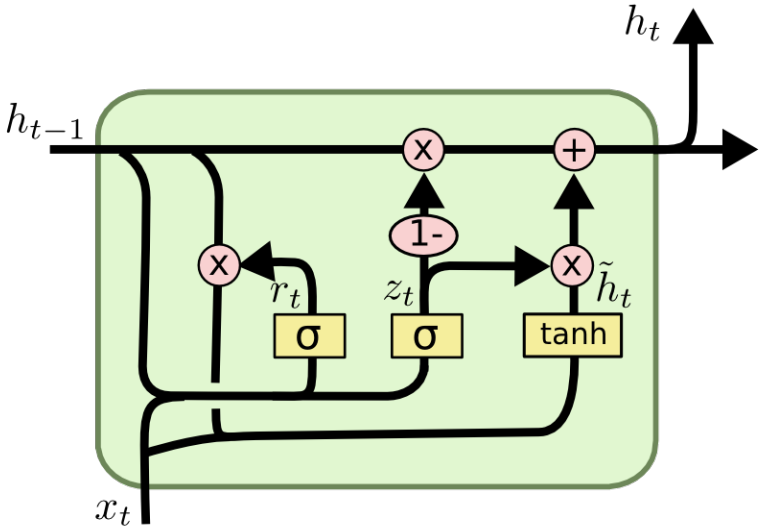
$$c_t^l = f \odot c_{t-1}^l + i \odot g$$

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



Gated Recurrent Unit (GRU)

- Alternative RNN to LSTM that uses fewer gates (Cho, et al., 2014)
  - Combines forget and input gates into "update" gate.
  - Eliminates cell state vector



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

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

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

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

GRU vs. LSTM

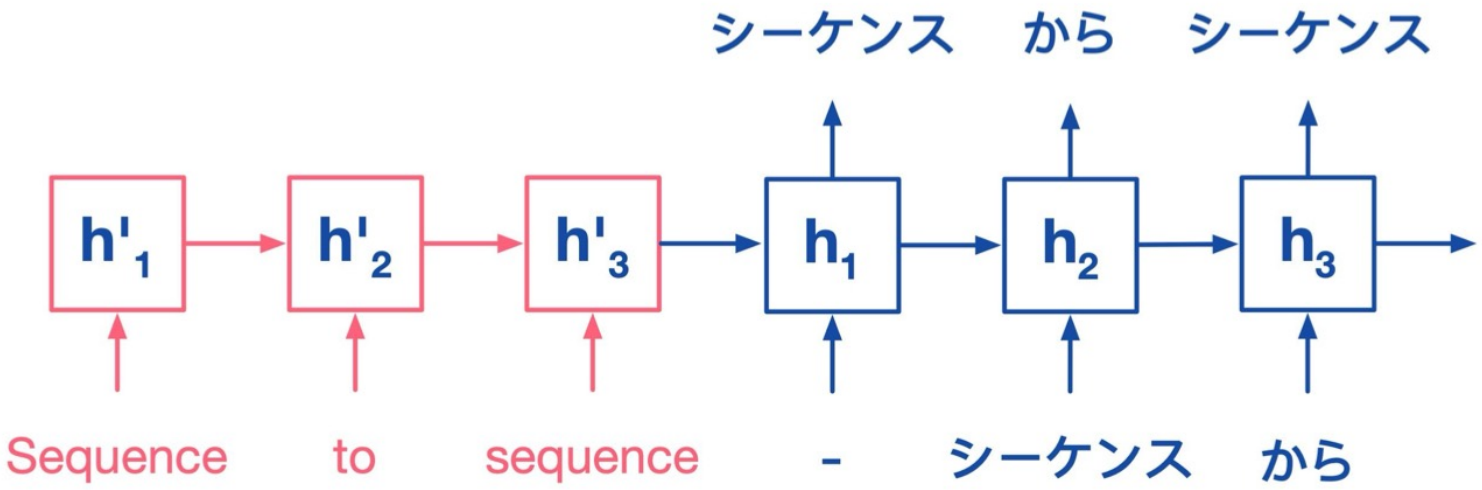
- GRU has significantly fewer parameters and trains faster.
- Experimental results comparing the two are still inconclusive, many problems they perform the same, but each has problems on which they work better.

Conclusions of LSTM

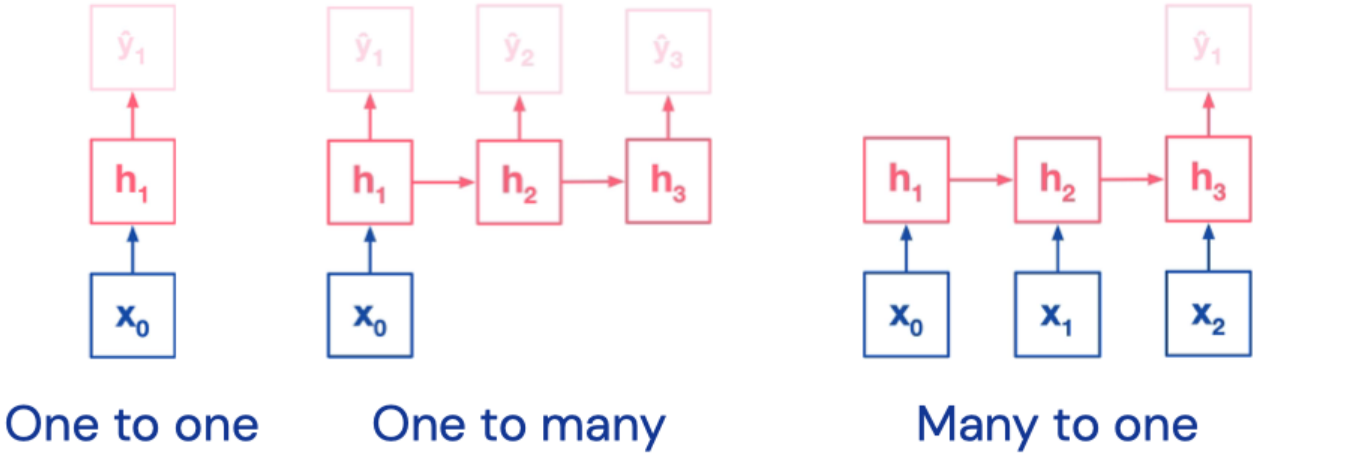
- By adding “gates” to an RNN, we can prevent the vanishing/exploding gradient problem.
- Trained LSTMs/GRUs can retain state information longer and handle long-distance dependencies.

Natural Language as Sequences

Sequence-to-sequence models



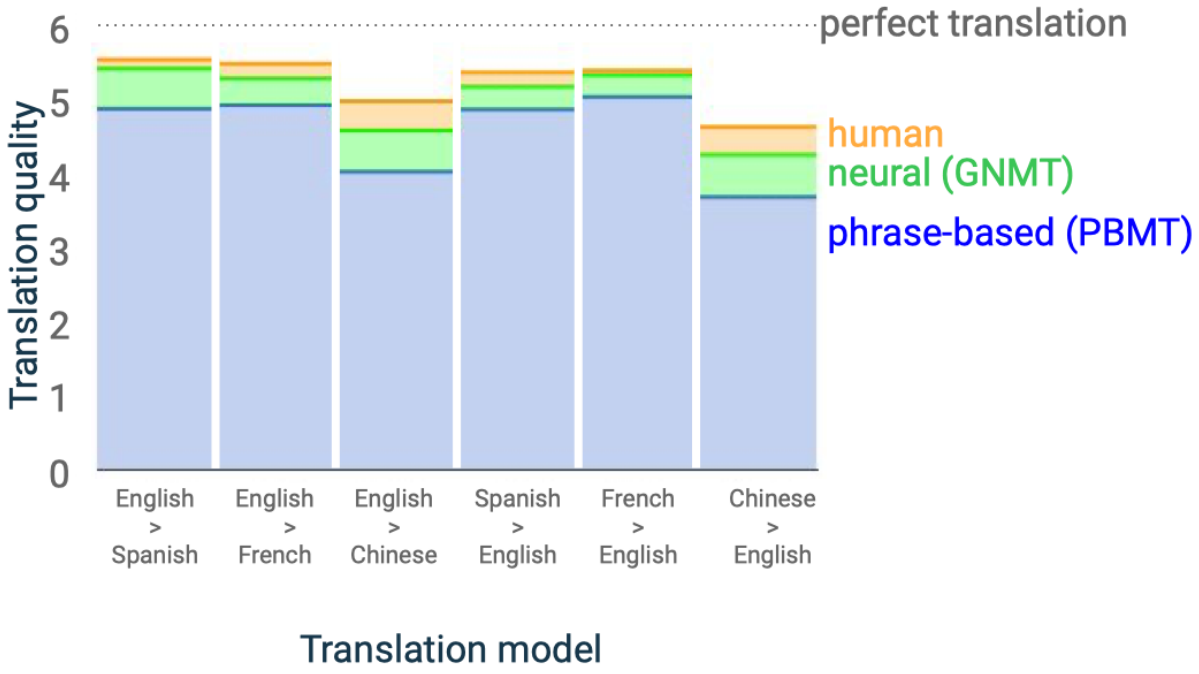
Flexible Sequence Mappings



Seq2seq has a wide range of applications

1. **MT** [Kalchbrenner et al, EMNLP 2013][Cho et al, EMLP 2014][Sutskever & Vinyals & Le, NIPS 2014][Luong et al, ACL 2015][Bahdanau et al, ICLR 2015]
2. **Image captions** [Mao et al, ICLR 2015][Vinyals et al, CVPR 2015][Donahue et al, CVPR 2015][Xu et al, ICML 2015]
3. **Speech** [Chorowsky et al, NIPS DL 2014][Chan et al, arxiv 2015]
4. **Parsing** [Vinyals & Kaiser et al, NIPS 2015]
5. **Dialogue** [Shang et al, ACL 2015][Sordoni et al, NAACL 2015][Vinyals & Le, ICML DL 2015]
6. **Video Generation** [Srivastava et al, ICML 2015]
7. **Geometry** [Vinyals & Fortunato & Jaitly, NIPS 2015]

Google Neural Machine Translation



Closes gap between old system and human-quality translation by 58% to 87%.

Image Captioning

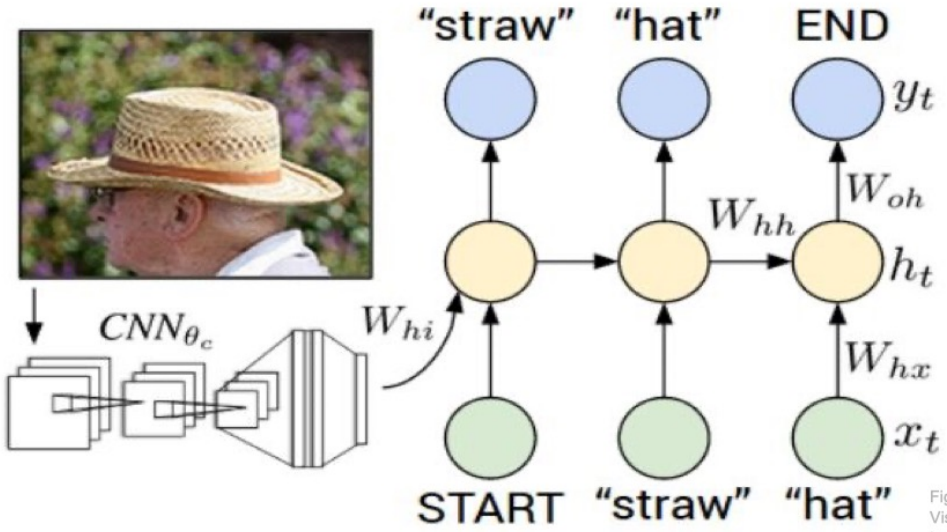


Figure from Karpathy et a, "Deep Visual-Semantic Alignments for Generating Image Descriptions", CVPR 2015; figure copyright IEEE, 2015. Reproduced for educational purposes.

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

Image Captioning

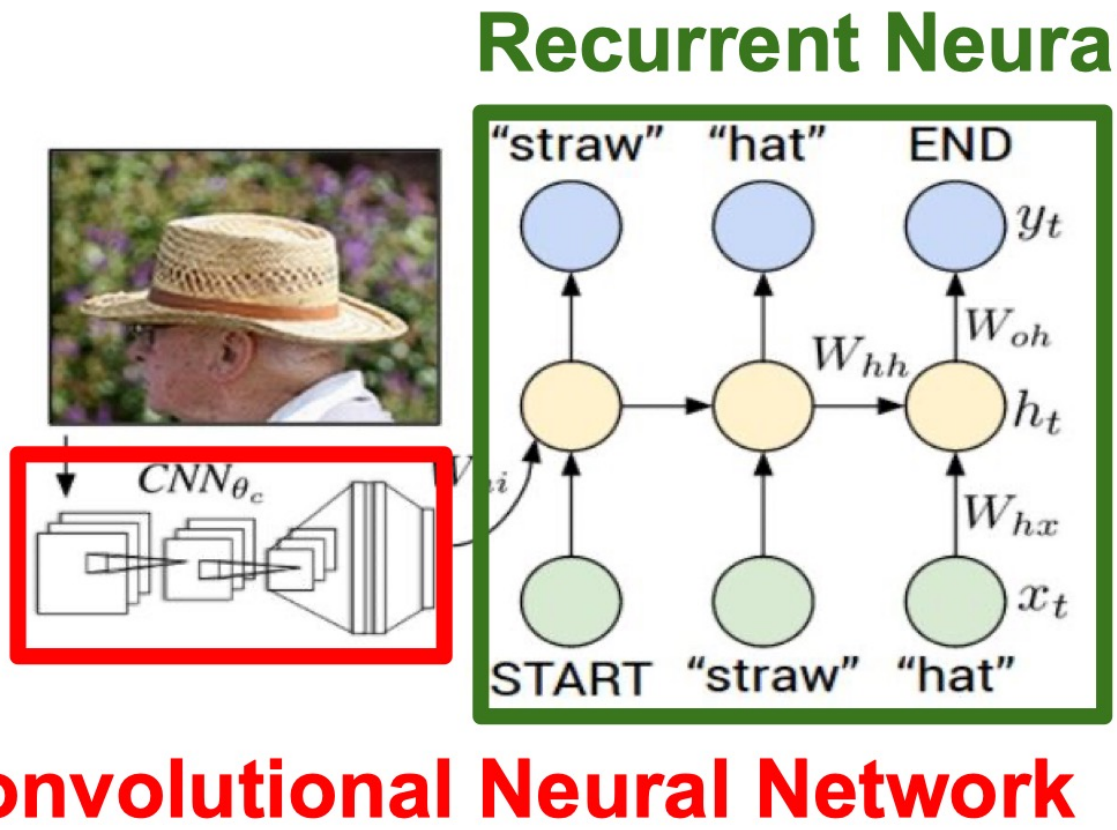


Image Captioning

This image is CC0 public domain

test image

Image Captioning

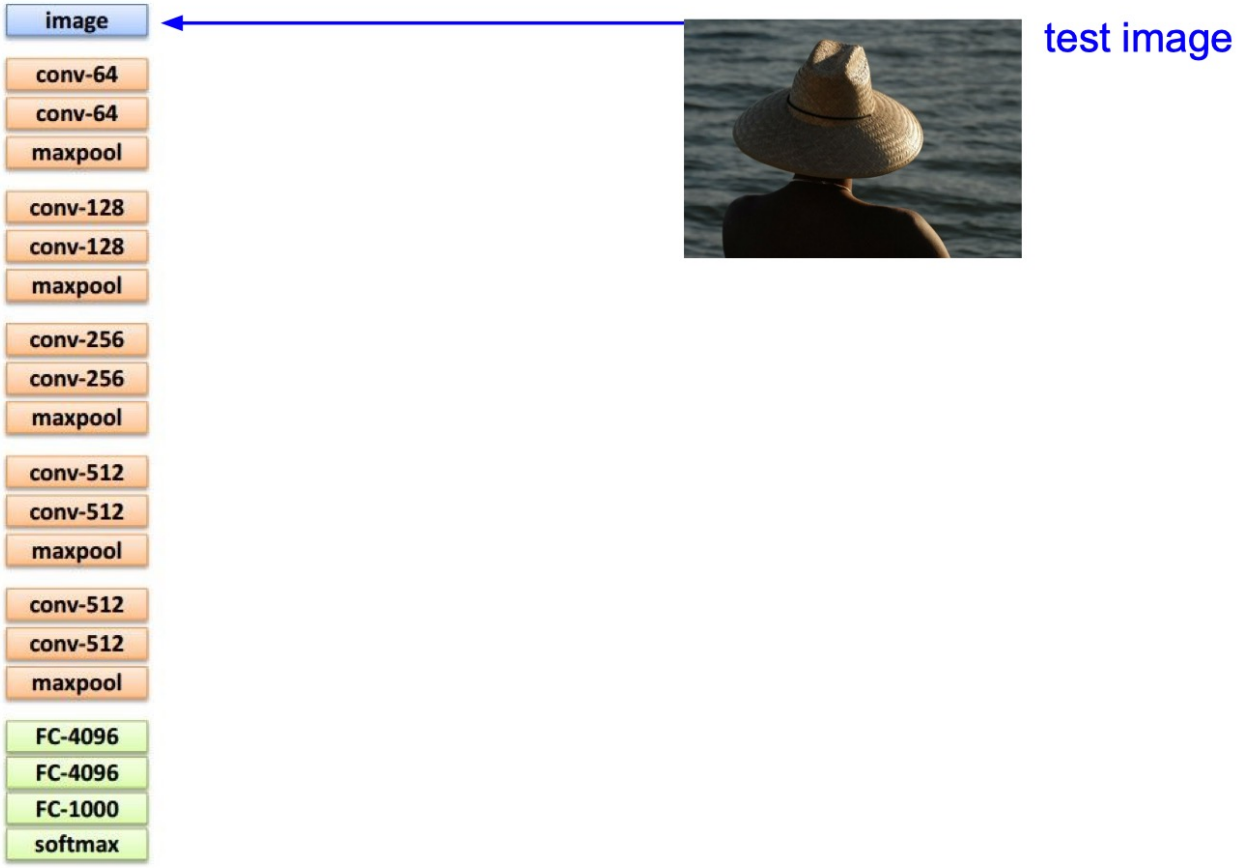


Image Captioning

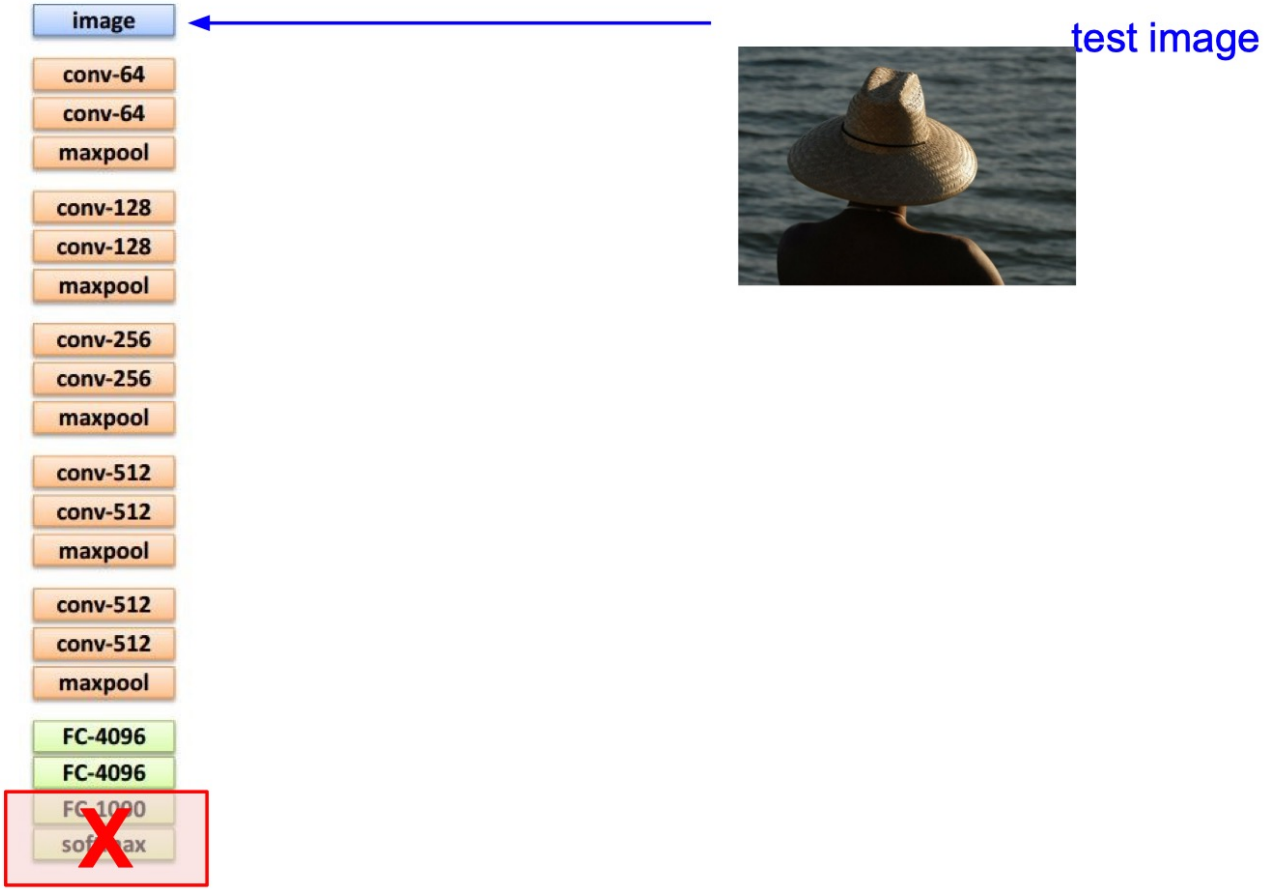


Image Captioning

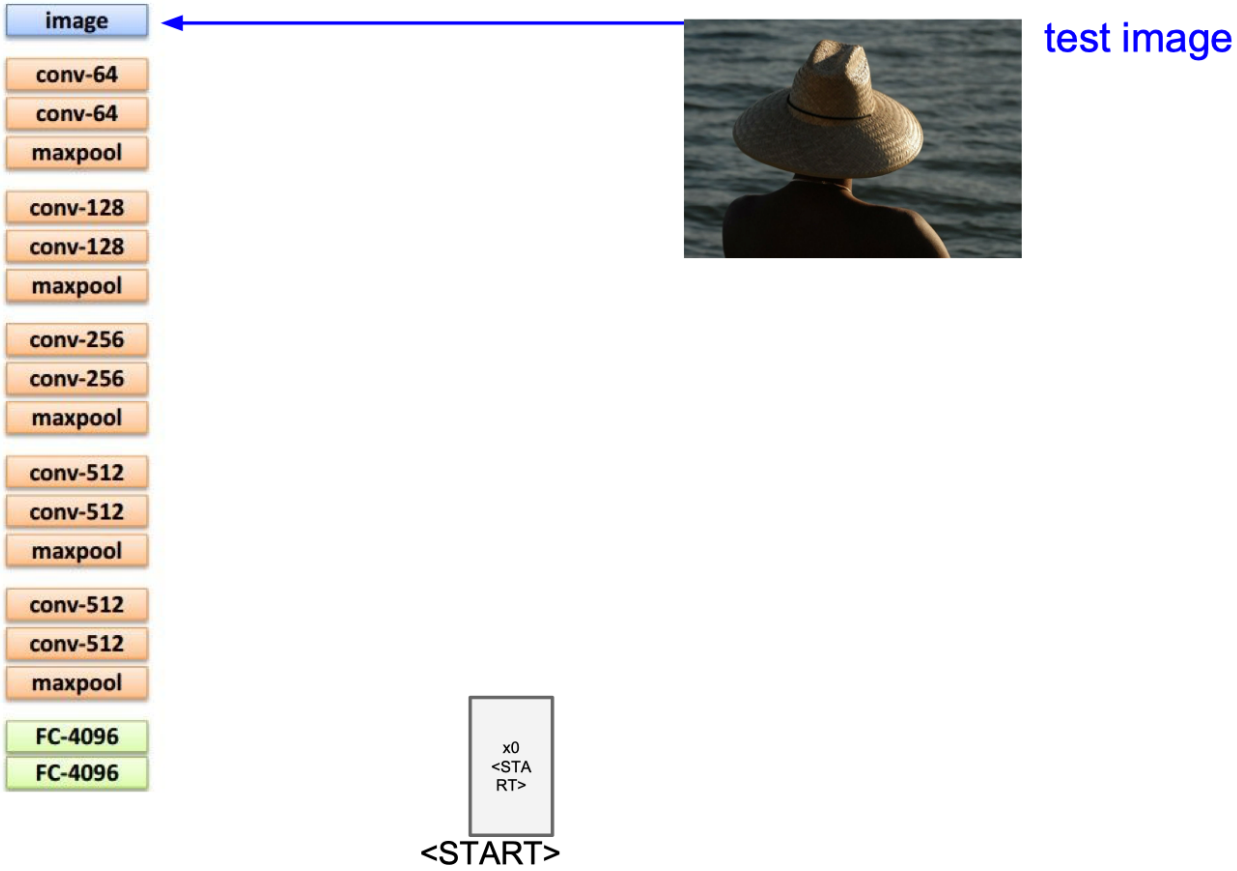
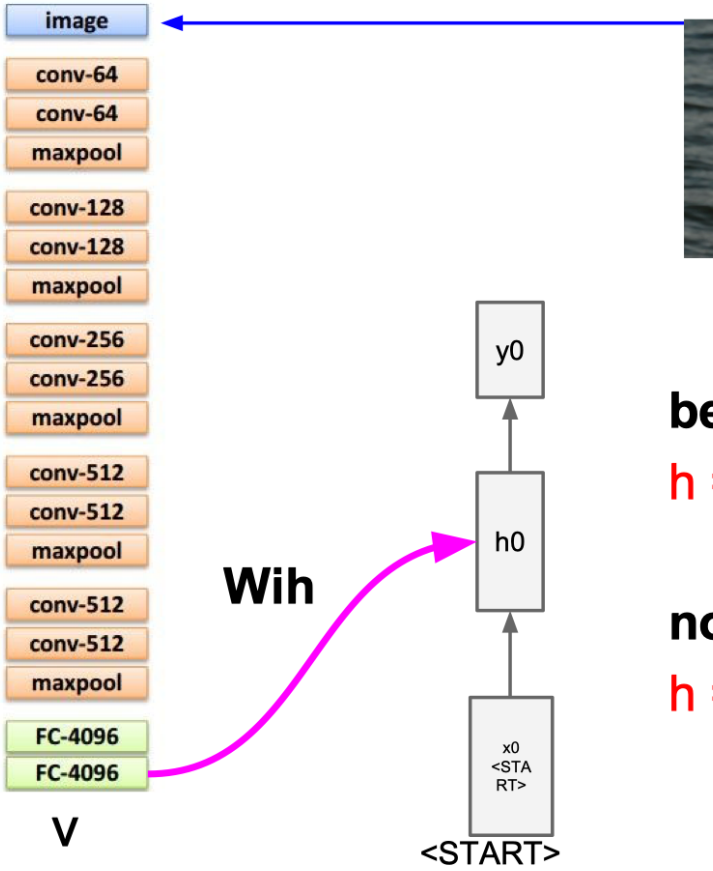


Image Captioning



test image

before:

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

now:

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

Image Captioning

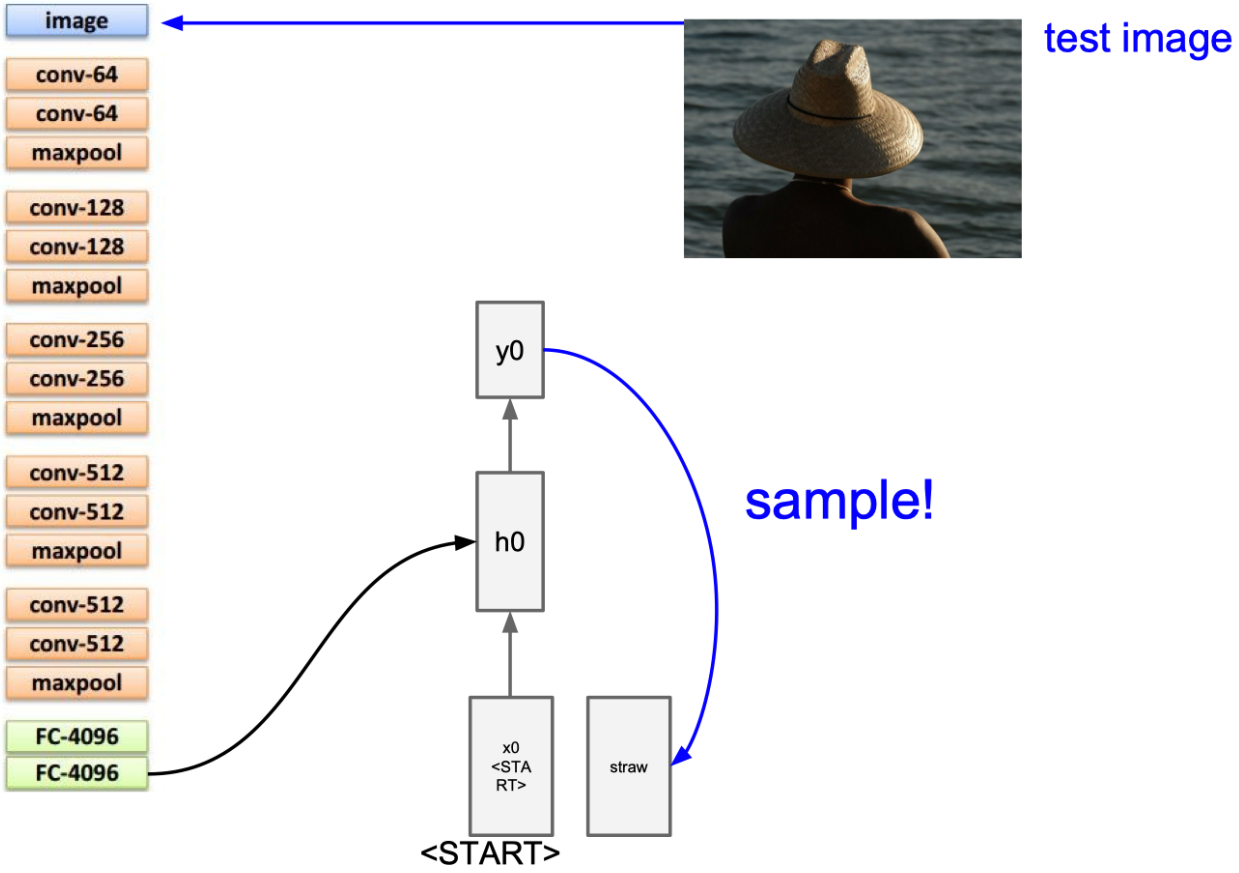


Image Captioning

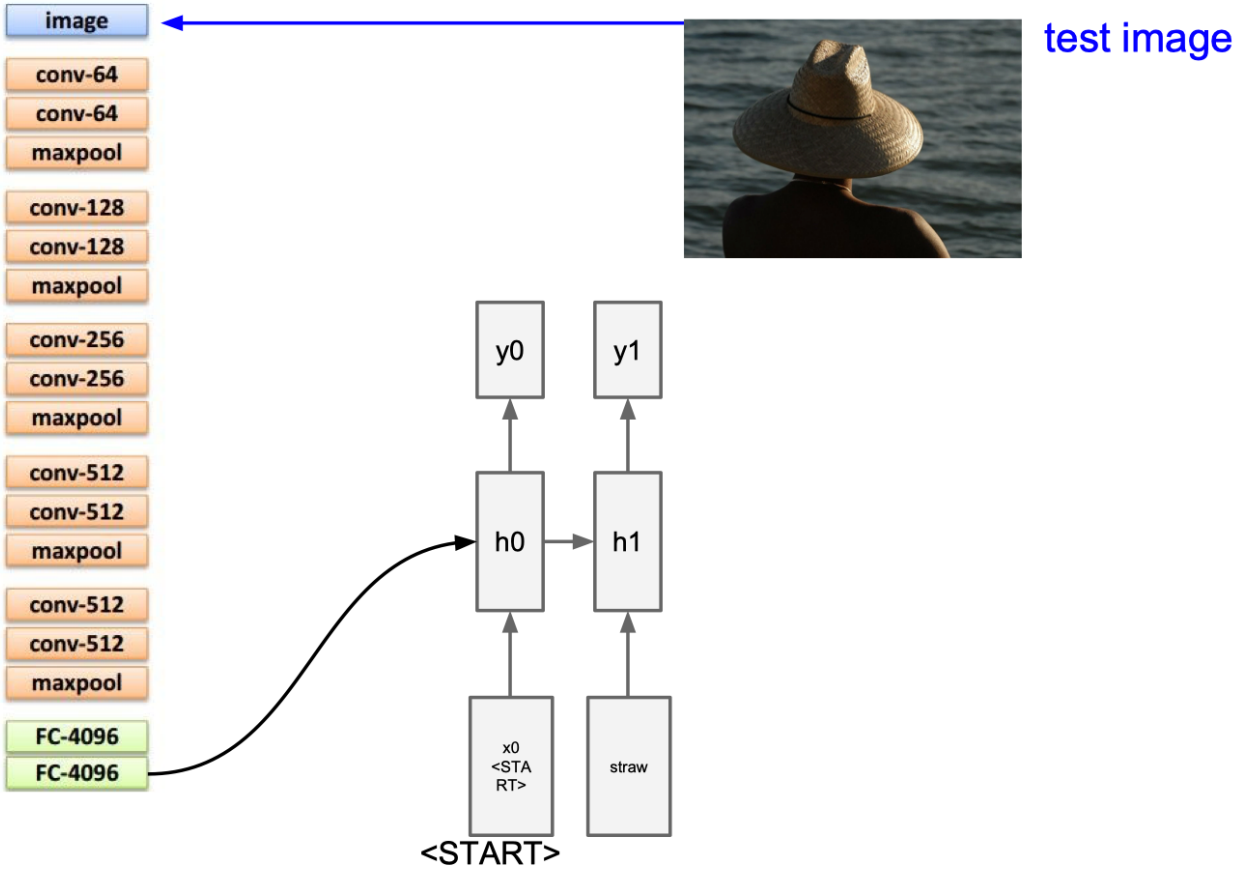


Image Captioning

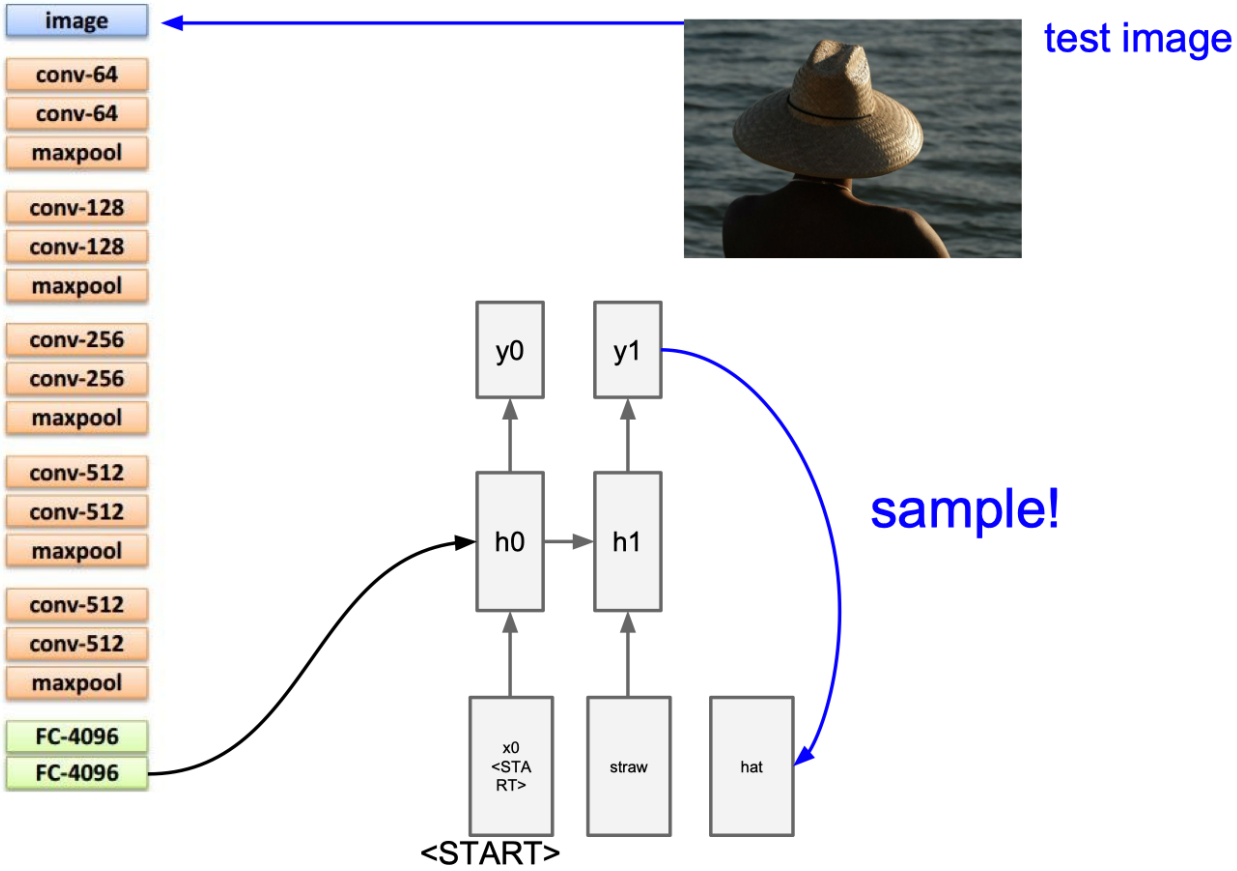


Image Captioning

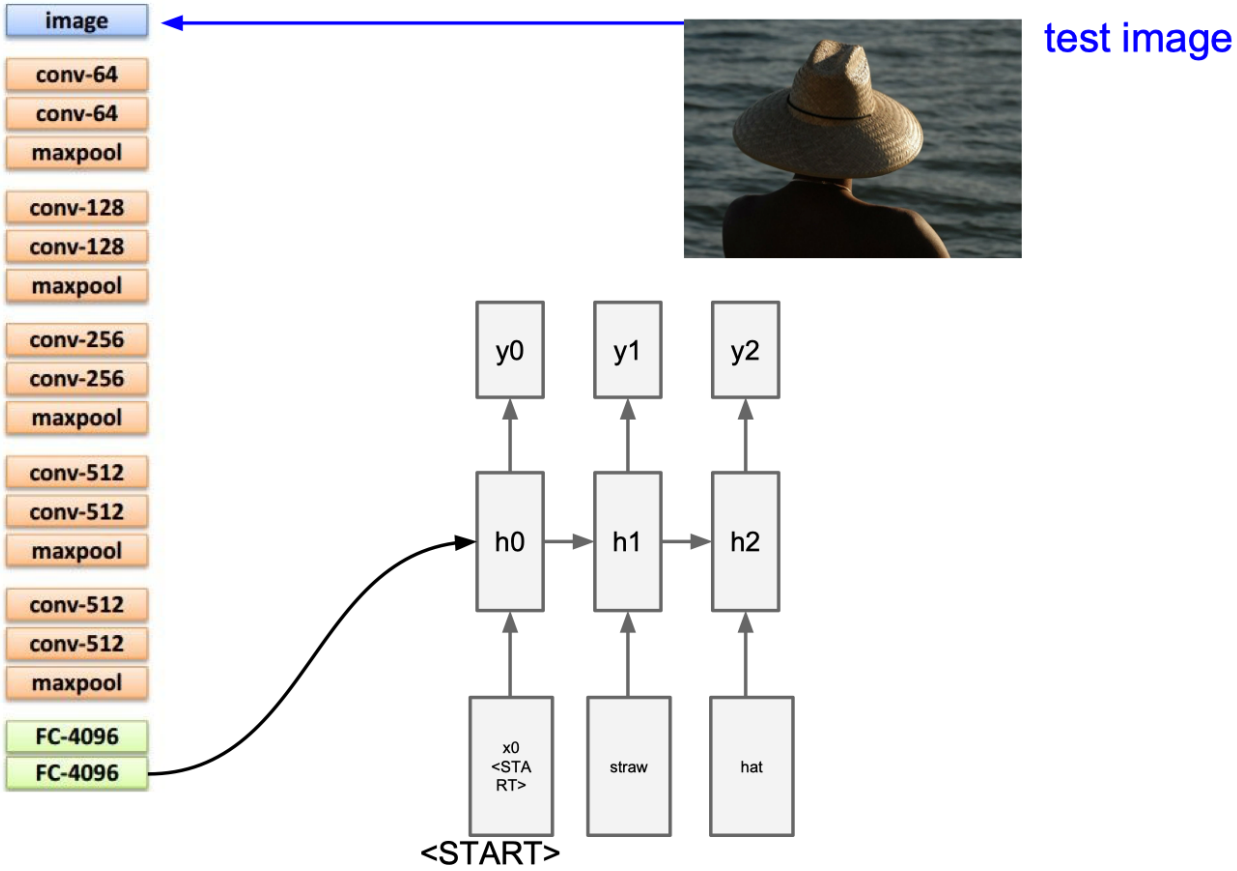


Image Captioning

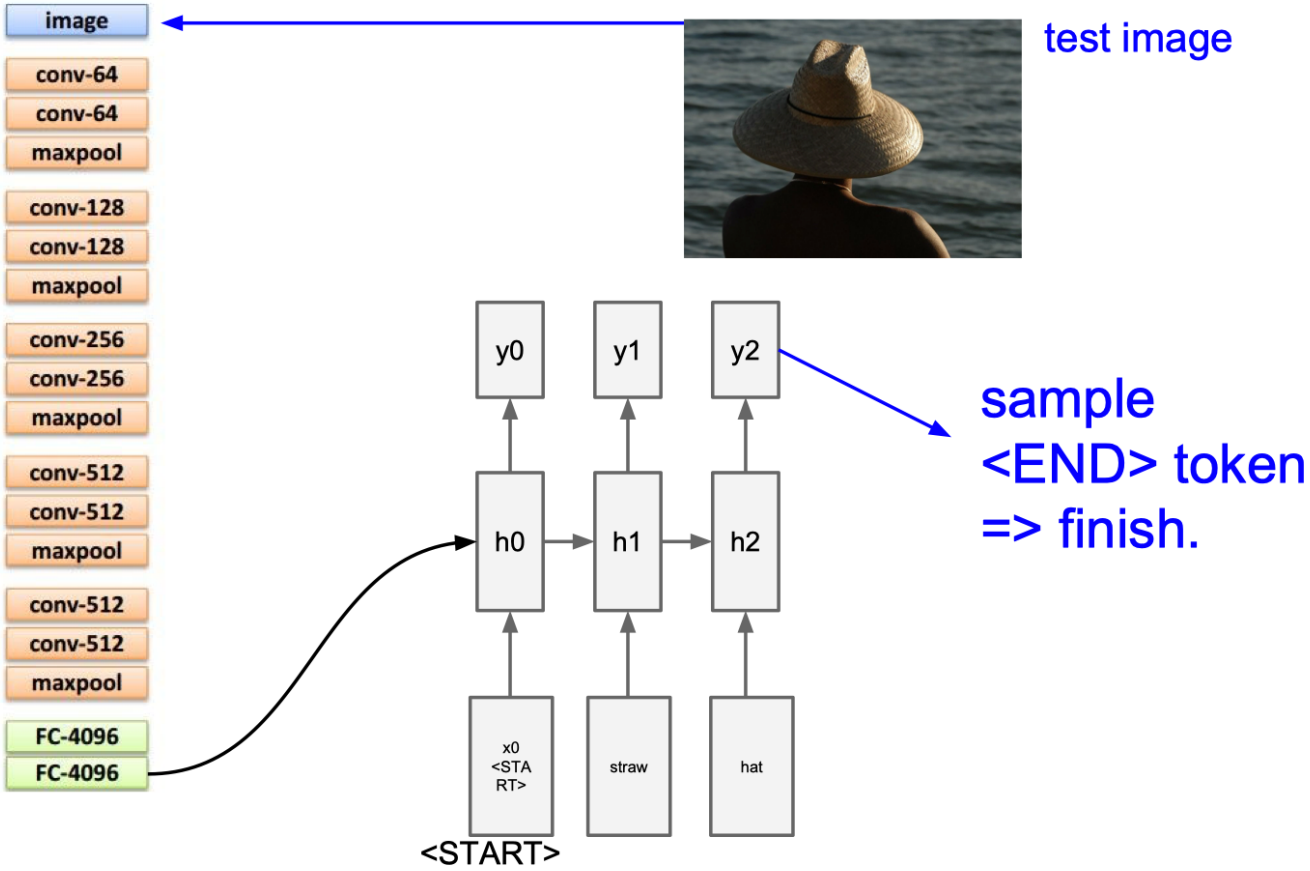


Image Captioning: Example Results

A cat sitting on a suitcase on the floor

A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

Two people walking on the beach with surfboards

A tennis player in action on the court

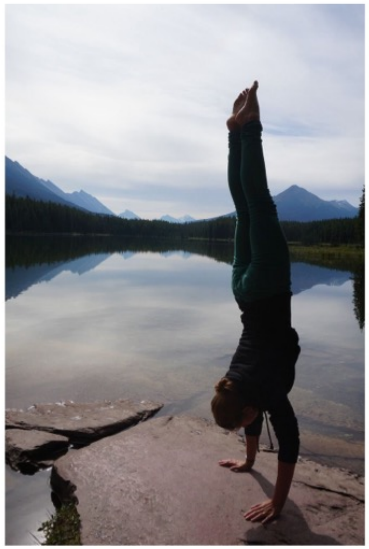
Two giraffes standing in a grassy field

A man riding a dirt bike on a dirt track

Image Captioning: Failure Cases

A woman is holding a cat in her hand

A person holding a computer mouse on a desk



A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

A man in a baseball uniform throwing a ball

Image Captioning with Attention

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

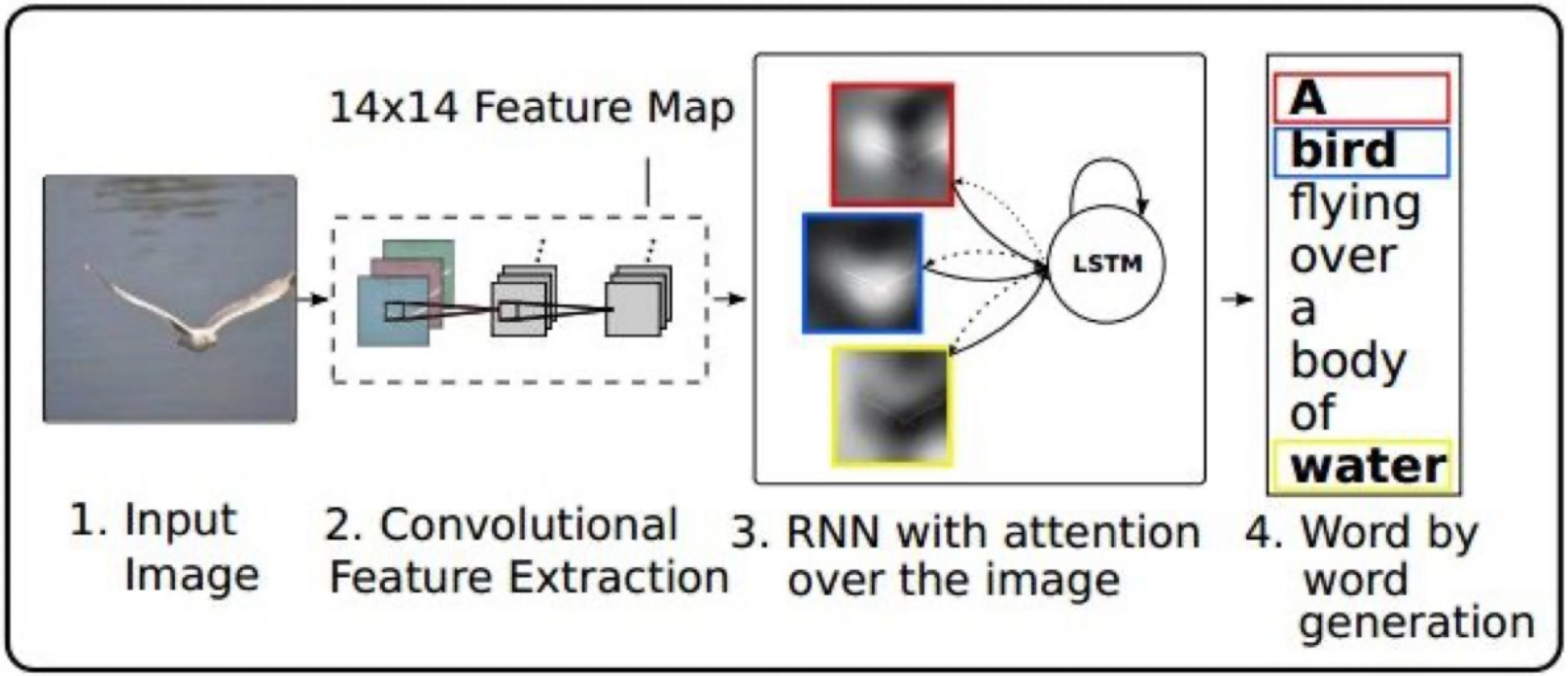
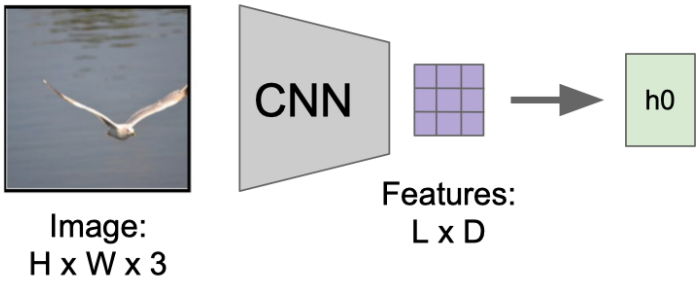
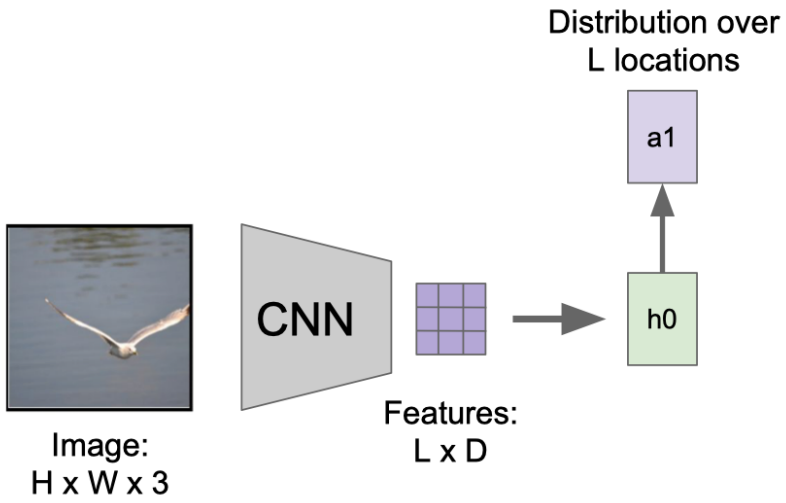


Image Captioning with Attention



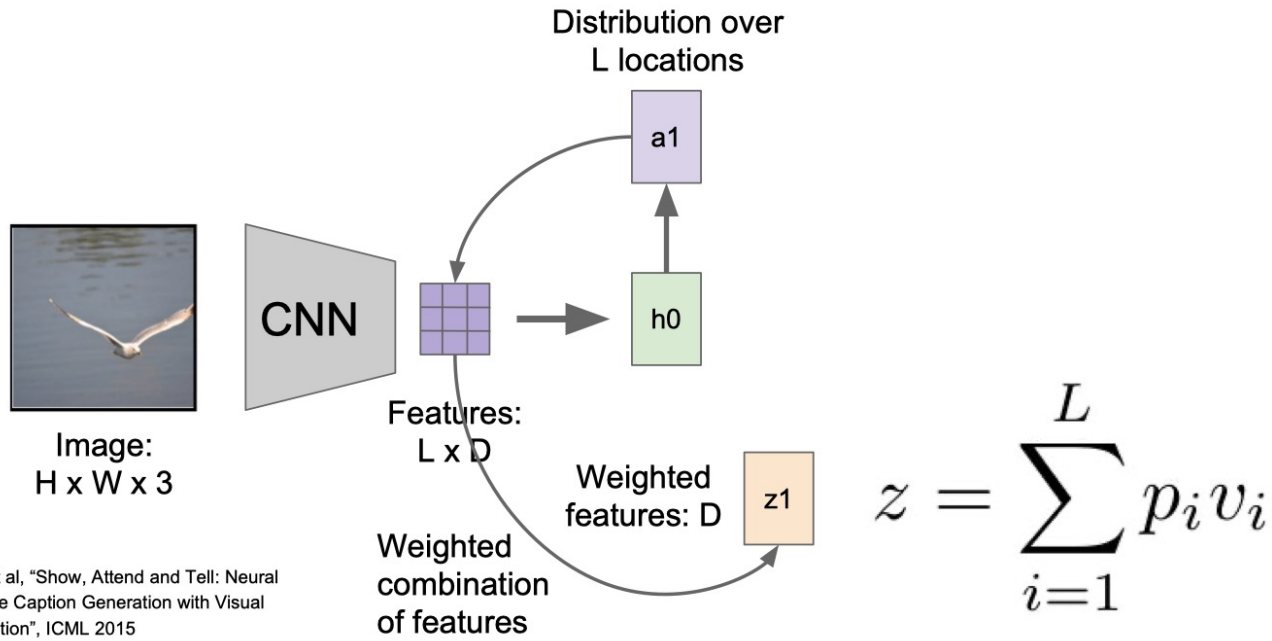
Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Image Captioning with Attention



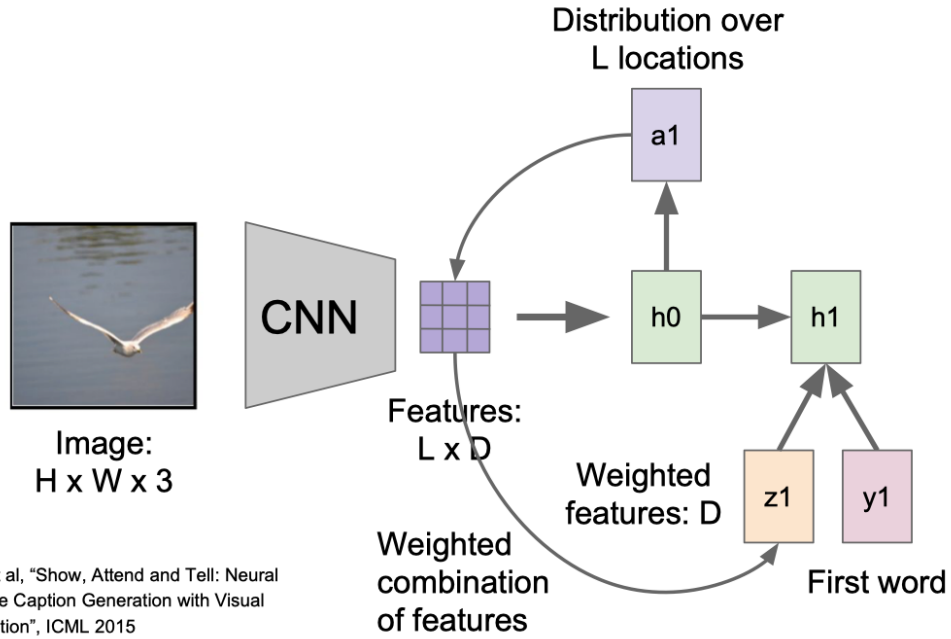
Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Image Captioning with Attention



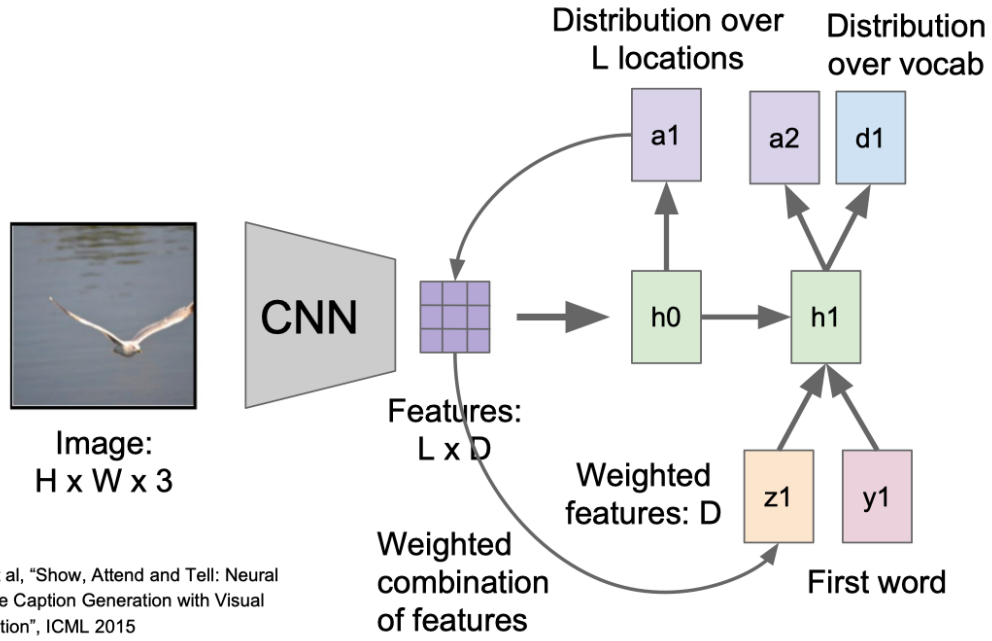
$$z = \sum_{i=1}^{L} p_i v_i$$

Image Captioning with Attention



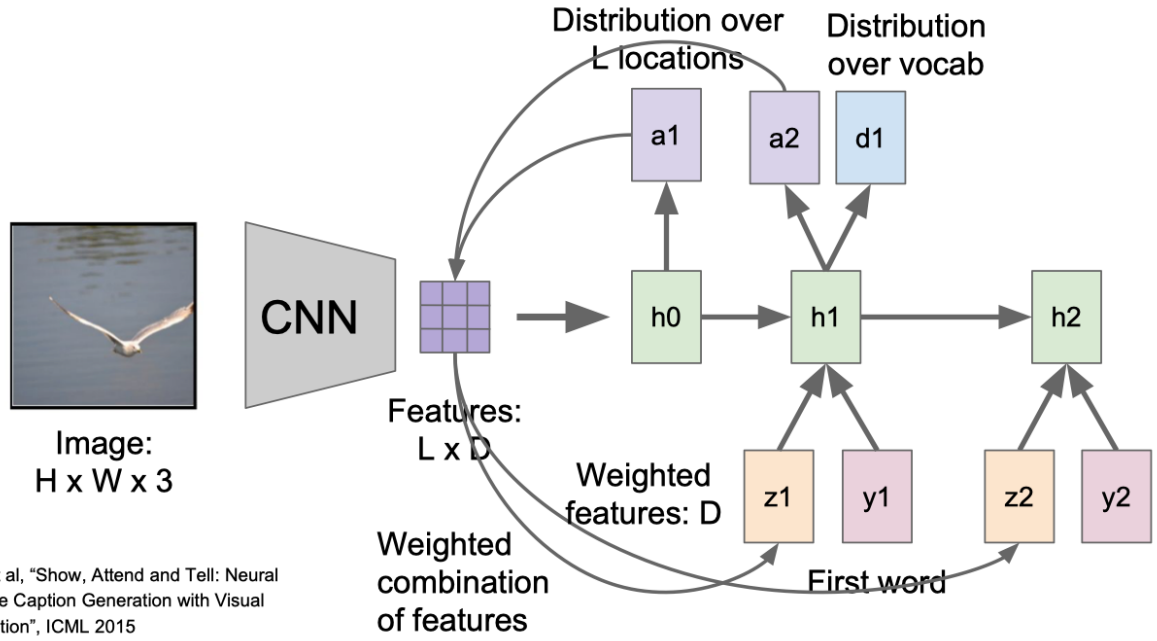
Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Image Captioning with Attention



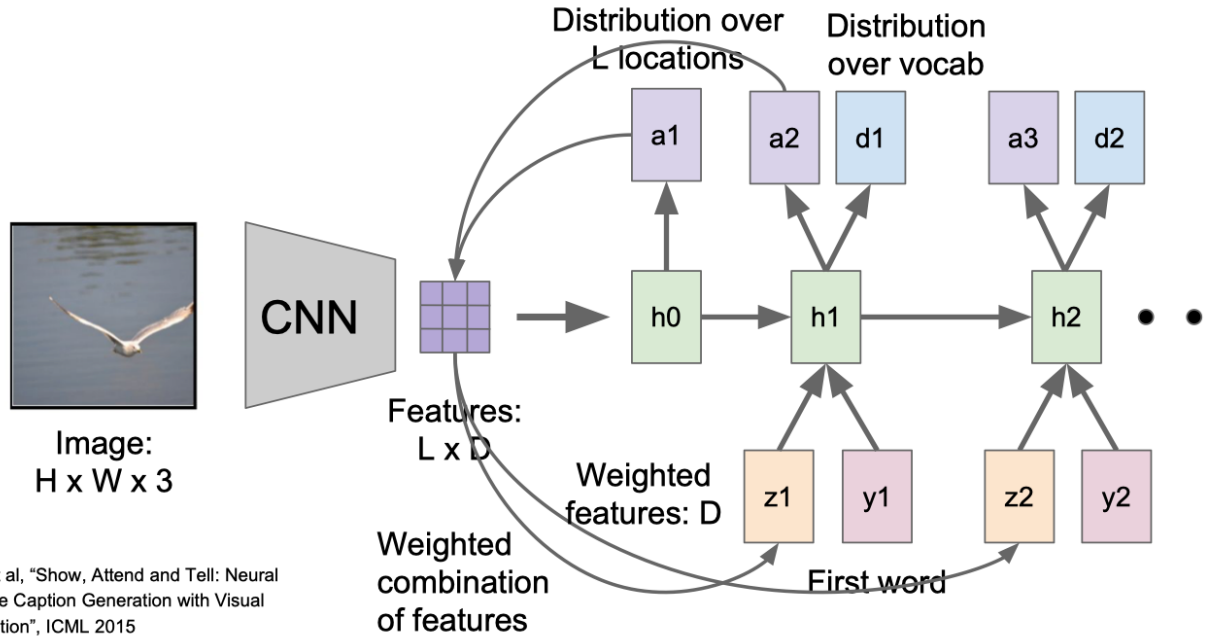
Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Image Captioning with Attention



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

Image Captioning with Attention



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

Image Captioning with Attention

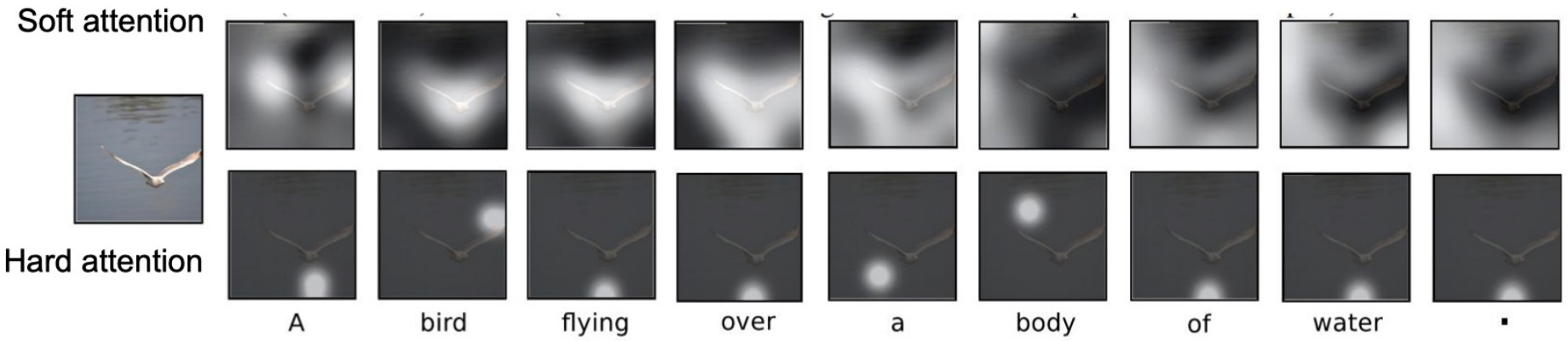
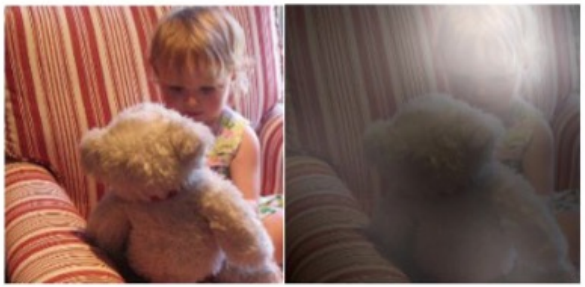


Image Captioning with Attention

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

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



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

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

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

Visual Question Answering

Q: What endangered animal is featured on the truck?

A: **A bald eagle.**
A: A sparrow.
A: A humming bird.
A: A raven.

Q: Where will the driver go if turning right?

A: **Onto 24 ¾ Rd.**
A: Onto 25 ¾ Rd.
A: Onto 23 ¾ Rd.
A: Onto Main Street.

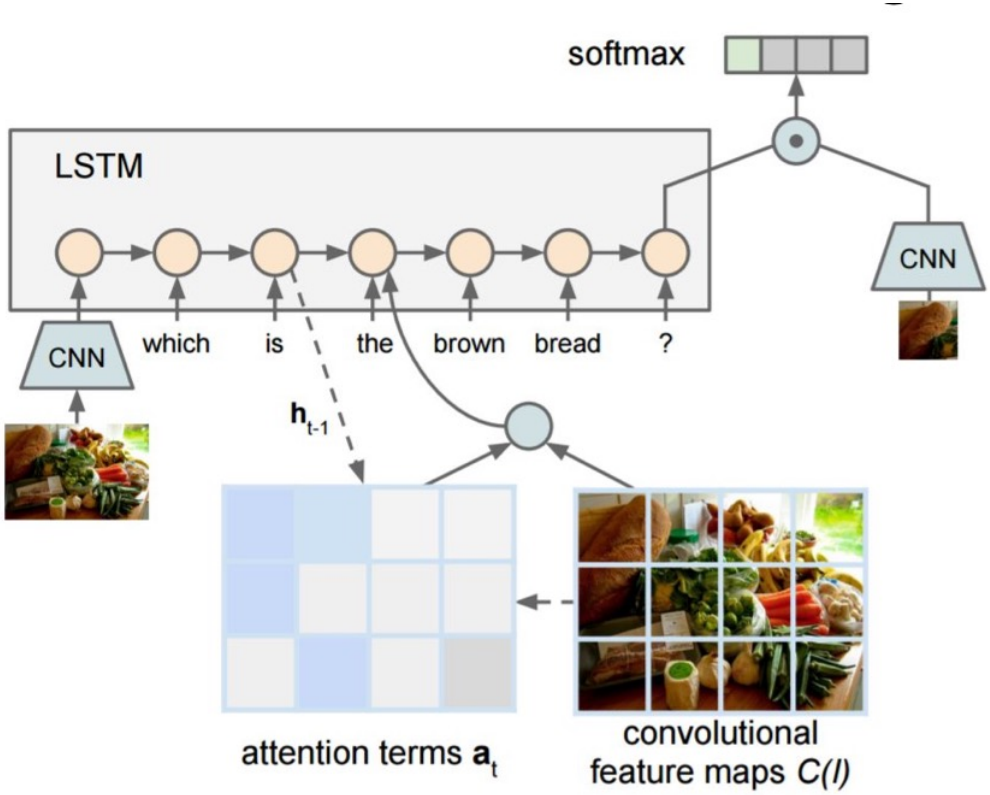
Q: When was the picture taken?

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

Q: Who is under the umbrella?

A: **Two women.**
A: A child.
A: An old man.
A: A husband and a wife.

Visual Question Answering



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

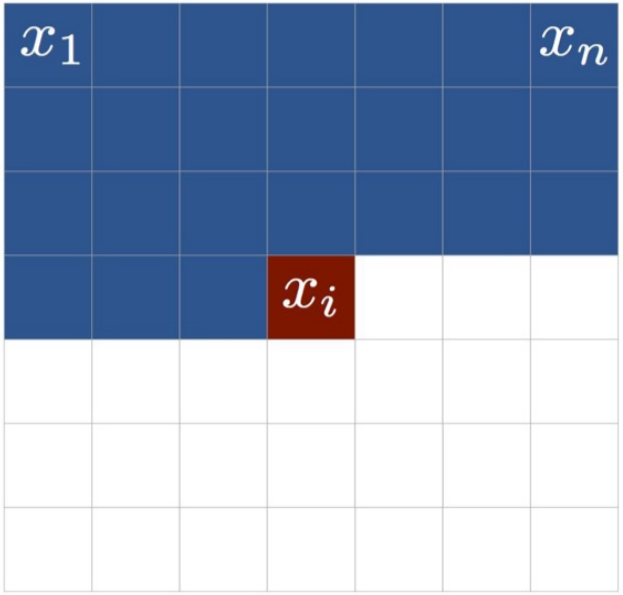
What kind of animal is in the photo?

A **cat**.

Why is the person holding a knife?

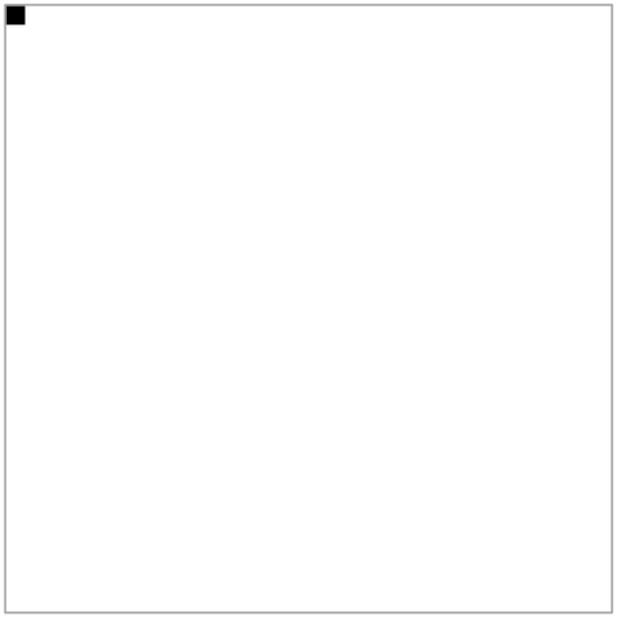
To cut the **cake** with.

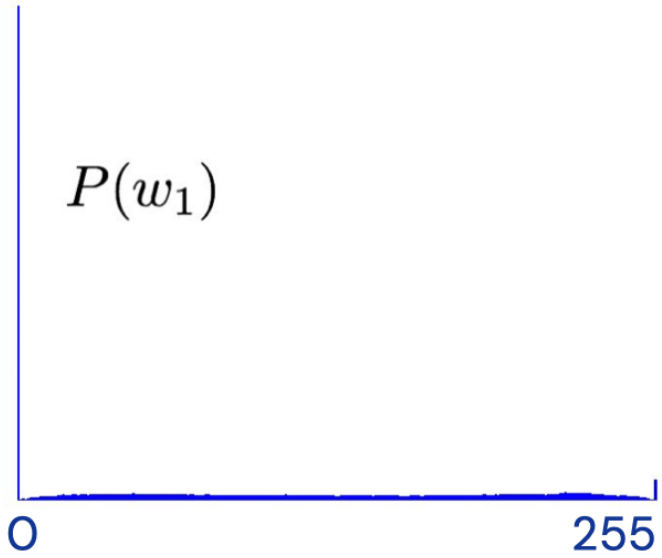
Images as sequences: PixelRNN



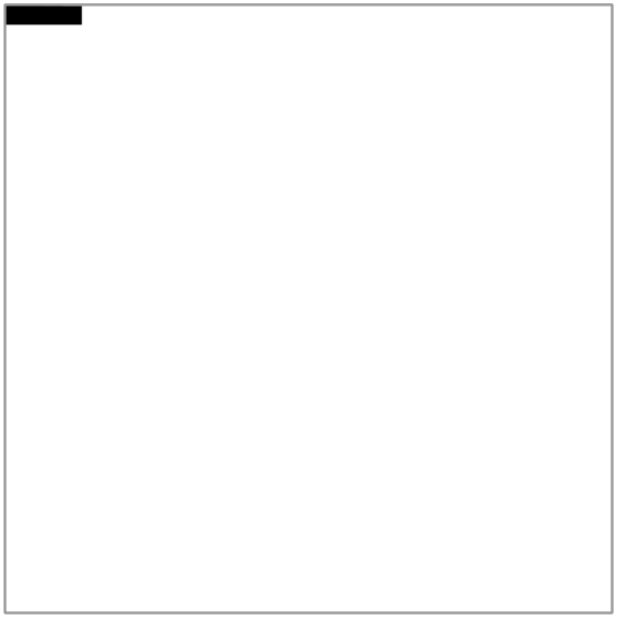
$$p(\mathbf{x}) = \prod_{i=1}^{n^2} p(x_i | x_1, \ldots, x_{i-1})$$

Images as sequences: PixelRNN



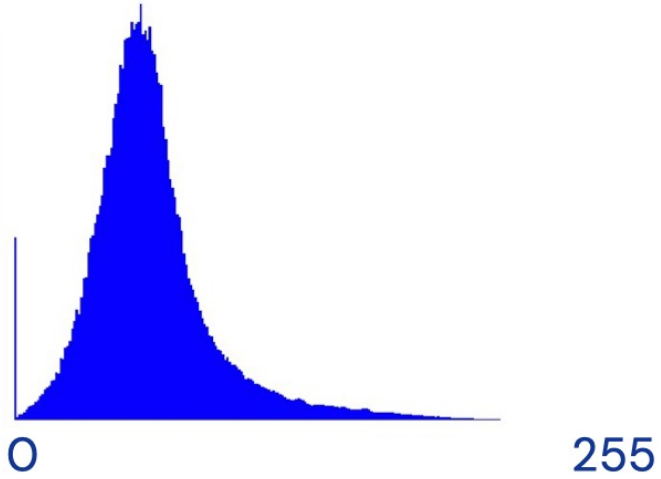


Images as sequences: PixelRNN

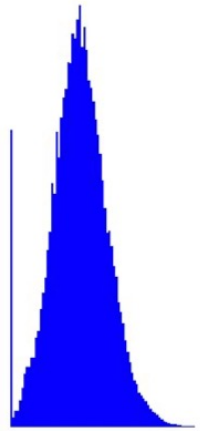


$$P(w_1)$$
$$P(w_2|w_1)$$
$$P(w_3|w_2, w_1)$$
$$P(w_4|w_3, w_2, w_1)$$

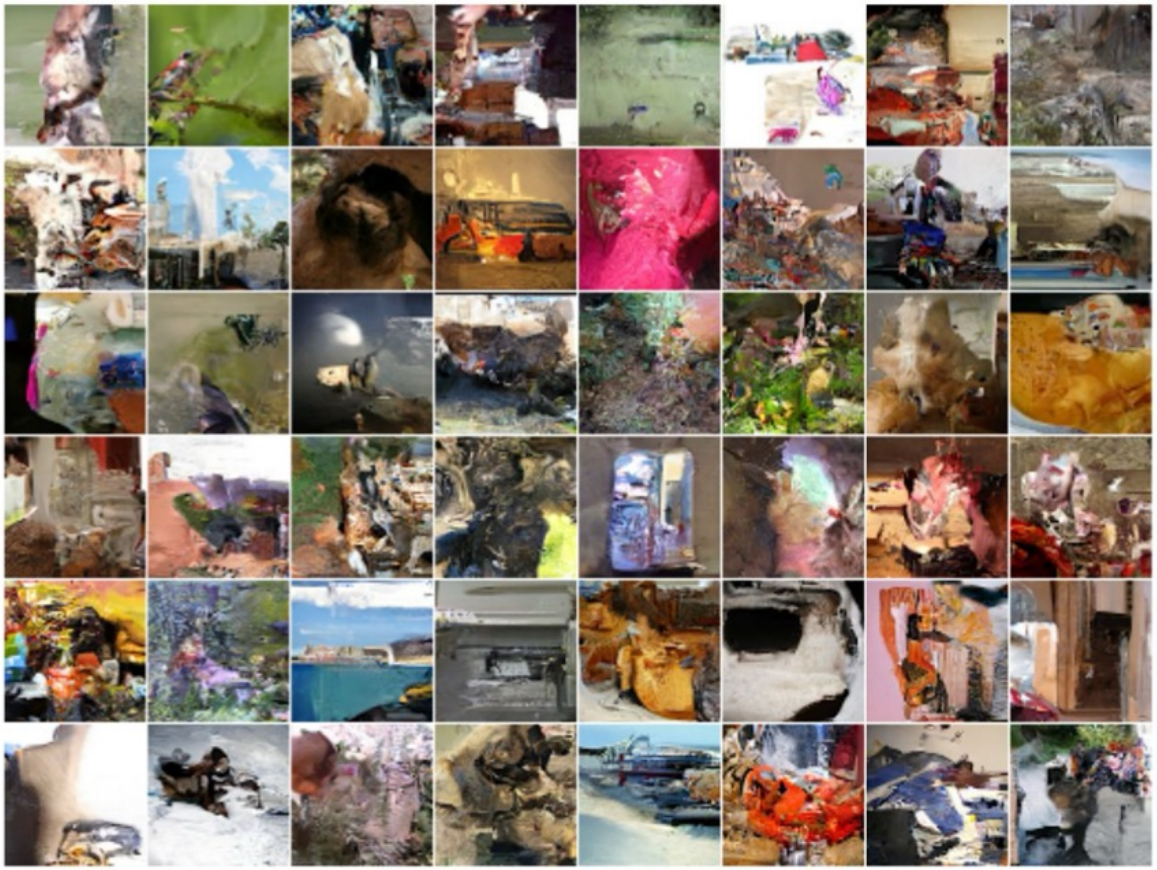
Images as sequences: PixelRNN



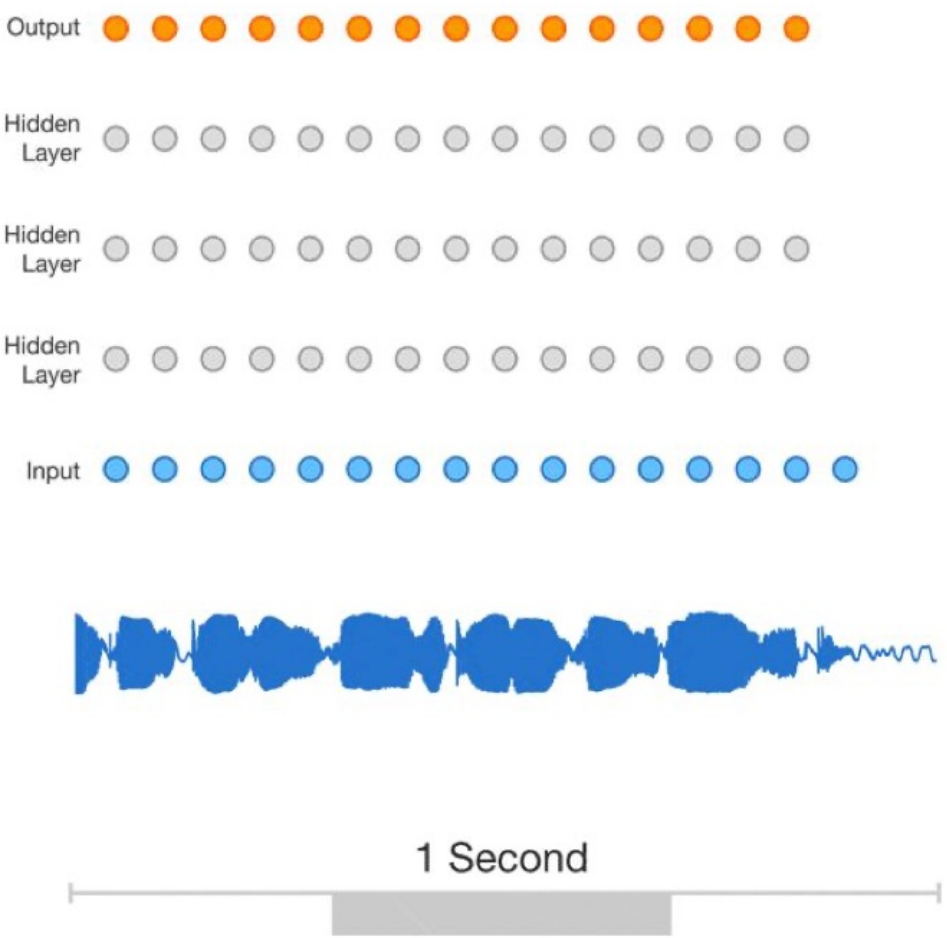
Images as sequences: PixelRNN



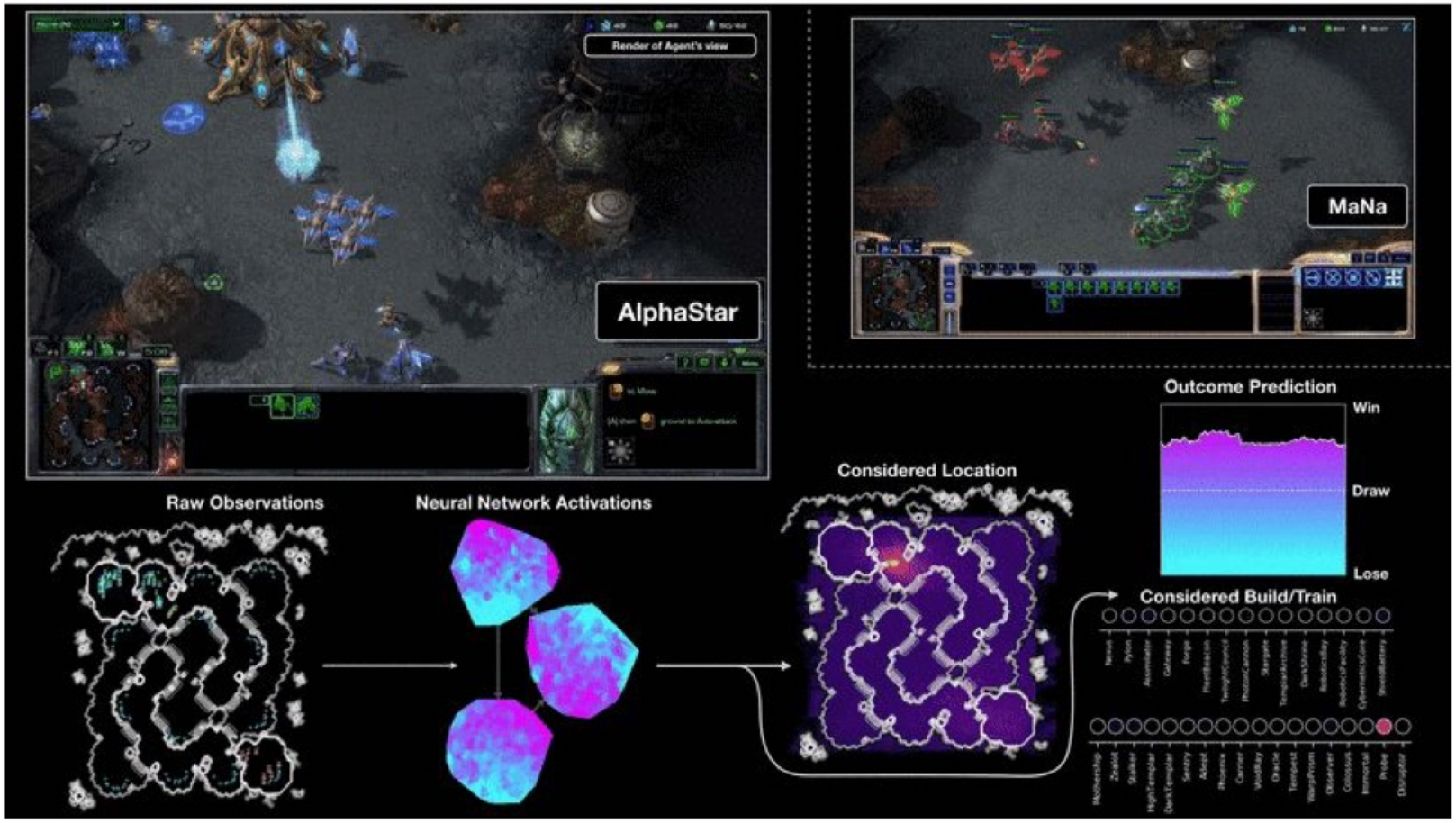
Images as sequences: PixelRNN



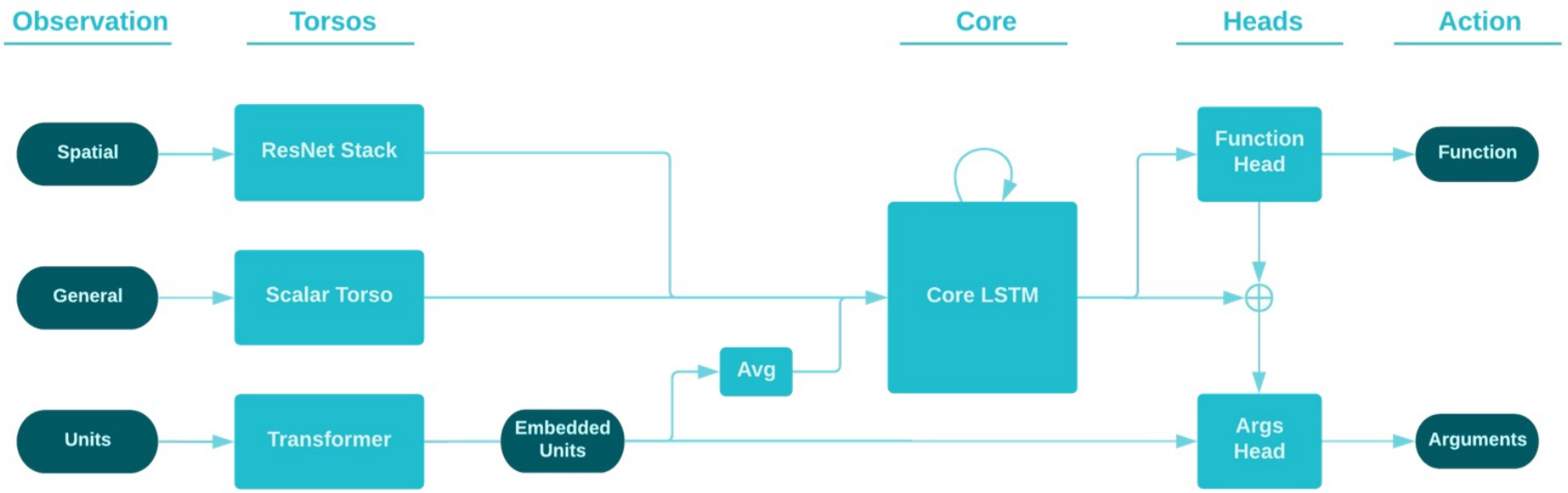
Audio waves as sequences



Policies as Sequences: Alphastar



AlphaStar Architecture



Questions?

UNIVERSITY OF NORTH CAROLINA CHARLOTTE