CAP 5516
Medical Image Computing
(Spring 2022)

Dr. Chen Chen
Center for Research in Computer Vision (CRCV)
University of Central Florida
Office: HEC 221
Address: 4328 Scorpius St., Orlando, FL 32816-2365
Email: chen.chen@crcv.ucf.edu
Web: https://www.crcv.ucf.edu/chenchen/

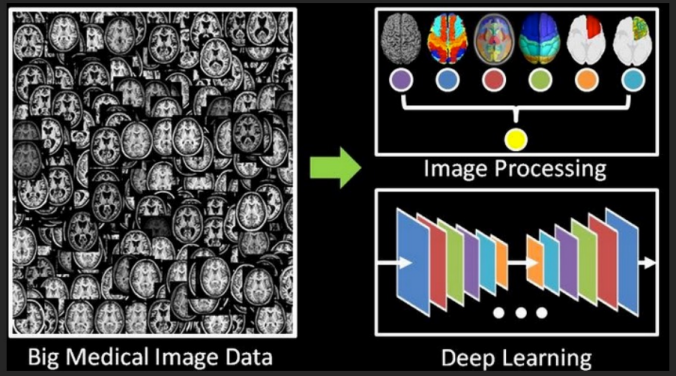
Lecture 8: Introduction to Deep Learning (4)

Interpretability of Deep Neural Networks

Safety of AI models

Autonomous Driving

Trust of AI decision



Medical Diagnosis

Policy and Regulation

Right to the explanation for algorithmic decisions

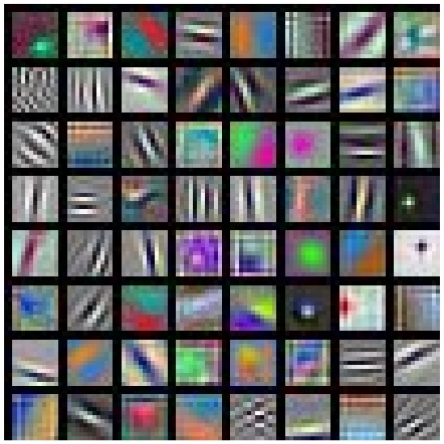
Interpretability of Deep Neural Networks

Debate on if interpretability is necessary

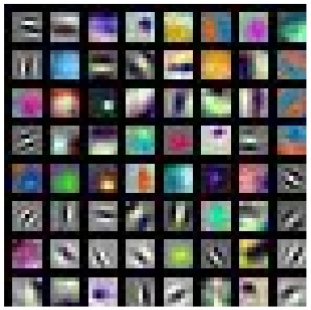
For: Rich Caruana, Patrice Simard
Against: Kilian Weinberger, Yann LeCun

Interpretable Machine Learning Symposium
https://www.youtube.com/watch?v=93Xv8vJ2acI

First Layer: Visualize Filters



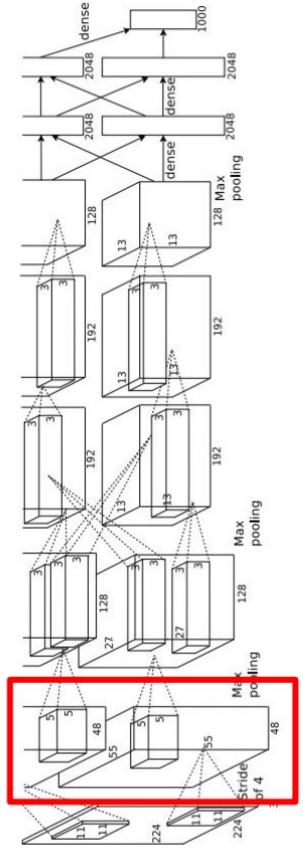
AlexNet:
64 x 3 x 11 x 11



ResNet-18:
64 x 3 x 7 x 7

ResNet-101:
64 x 3 x 7 x 7

DenseNet-121:
64 x 3 x 7 x 7



Krizhevsky, "One weird trick for parallelizing convolutional neural networks", arXiv 2014
He et al, "Deep Residual Learning for Image Recognition", CVPR 2016
Huang et al, "Densely Connected Convolutional Networks", CVPR 2017

Visualize the filters/kernels (raw weights)

We can visualize filters at higher layers, but not that interesting

(these are taken from ConvNetJS CIFAR-10 demo)

Weights:

layer 1 weights

16 x 3 x 7 x 7

Weights:

layer 2 weights

20 x 16 x 7 x 7

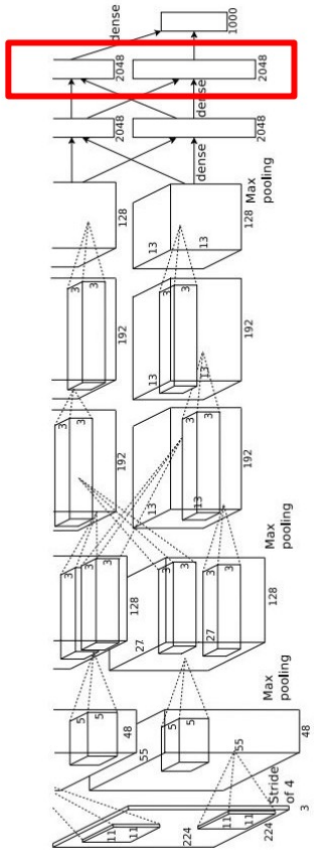
Weights:

layer 3 weights

20 x 20 x 7 x 7

Last Layer

FC7 layer

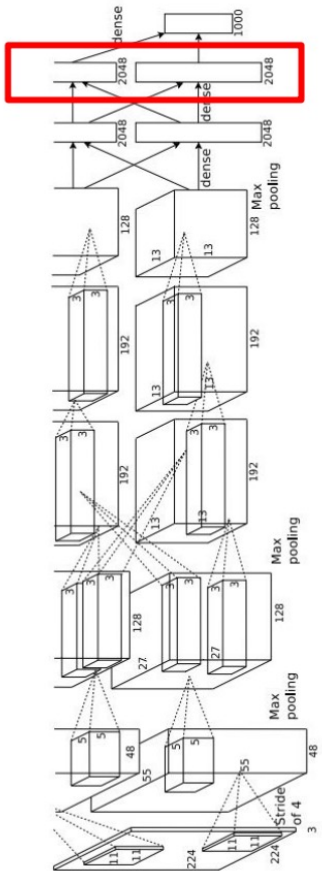


4096-dimensional feature vector for an image (layer immediately before the classifier)

Run the network on many images, collect the feature vectors

Last Layer: Nearest Neighbors

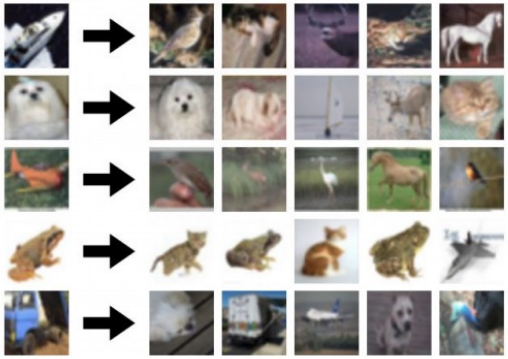
4096-dim vector



Test image

L2 Nearest neighbors in feature space

Recall: Nearest neighbors in pixel space



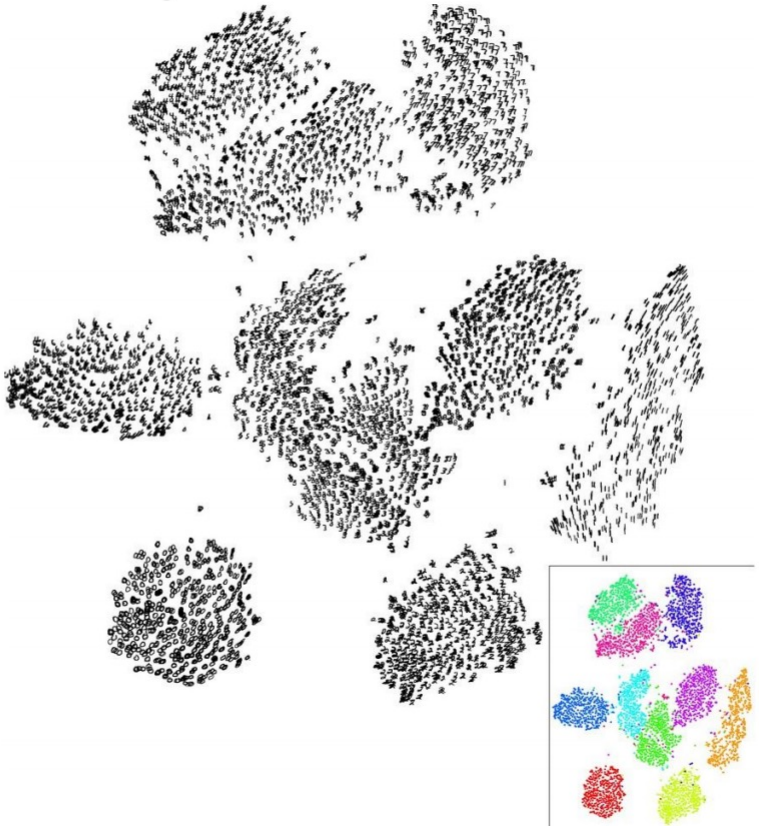
Krizhevsky et al, "ImageNet Classification with Deep Convolutional Neural Networks", NIPS 2012.
Figures reproduced with permission.

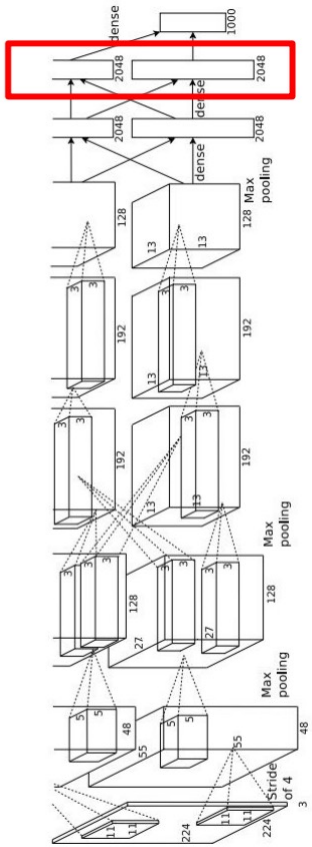
Last Layer: Dimensionality Reduction

Visualize the “space” of FC7 feature vectors by reducing dimensionality of vectors from 4096 to 2 dimensions

Simple algorithm: Principle Component Analysis (PCA)

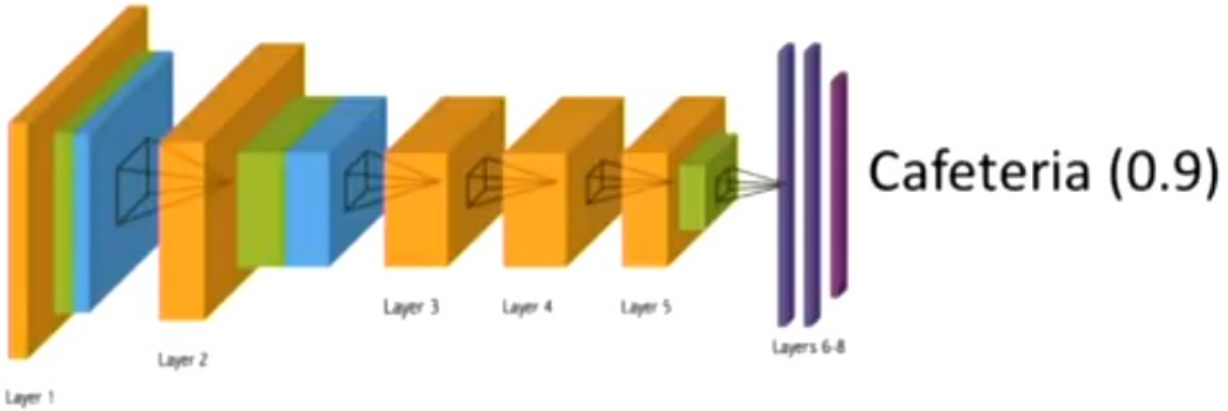
More complex: **t-SNE**





Van der Maaten and Hinton, “Visualizing Data using t-SNE”, JMLR 2008

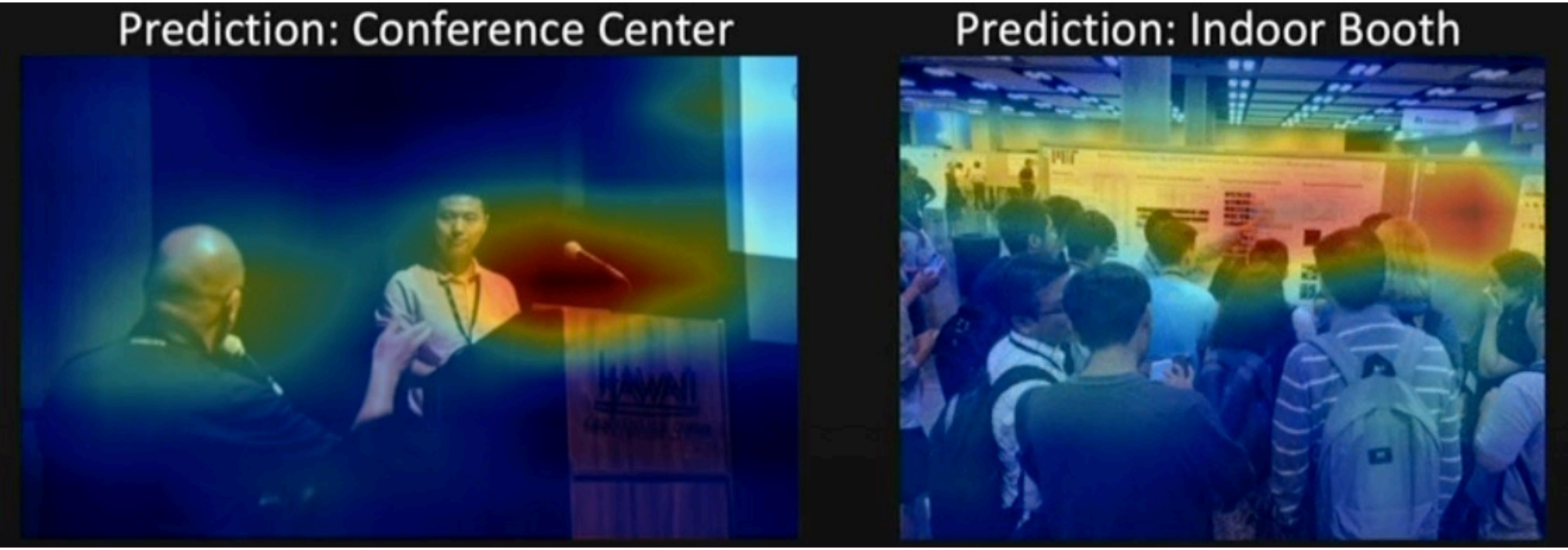
Why the network gives such prediction?



Credit: Bolei Zhou

Class Activation Map

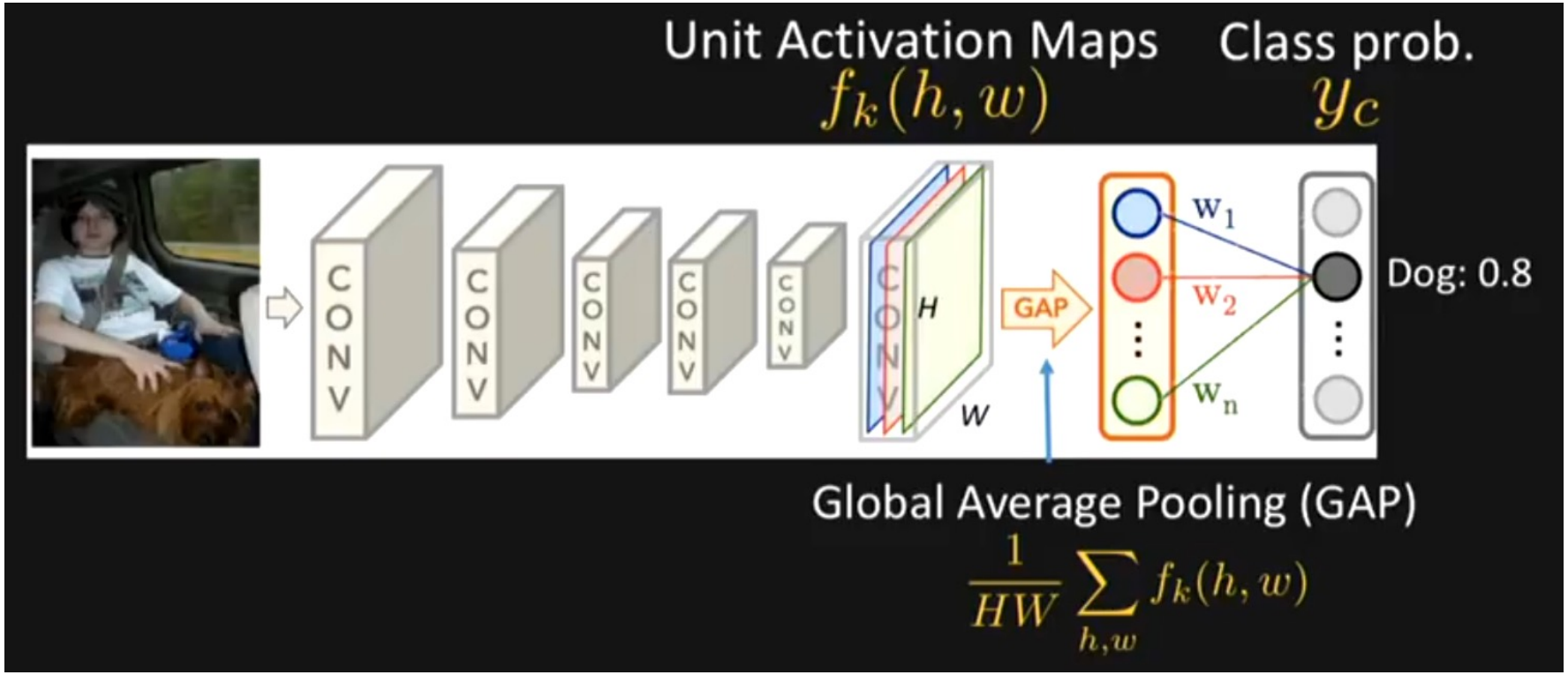
- Explain Prediction of Deep Neural Network



Credit: Bolei Zhou

Zhou, Bolei, et al. "Learning deep features for discriminative localization." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

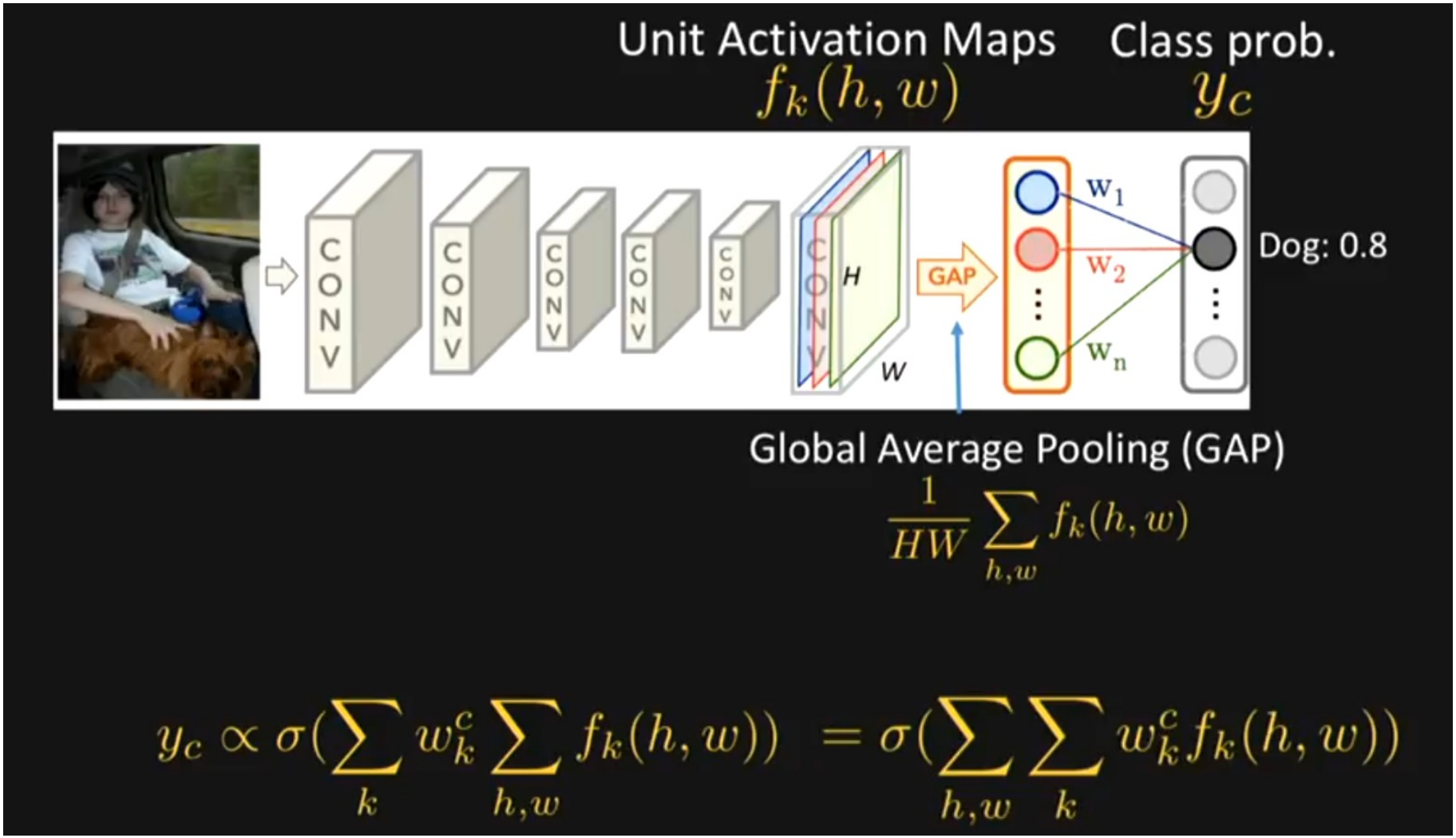
What the CNN is looking



Credit: Bolei Zhou

Zhou, Bolei, et al. "Learning deep features for discriminative localization." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

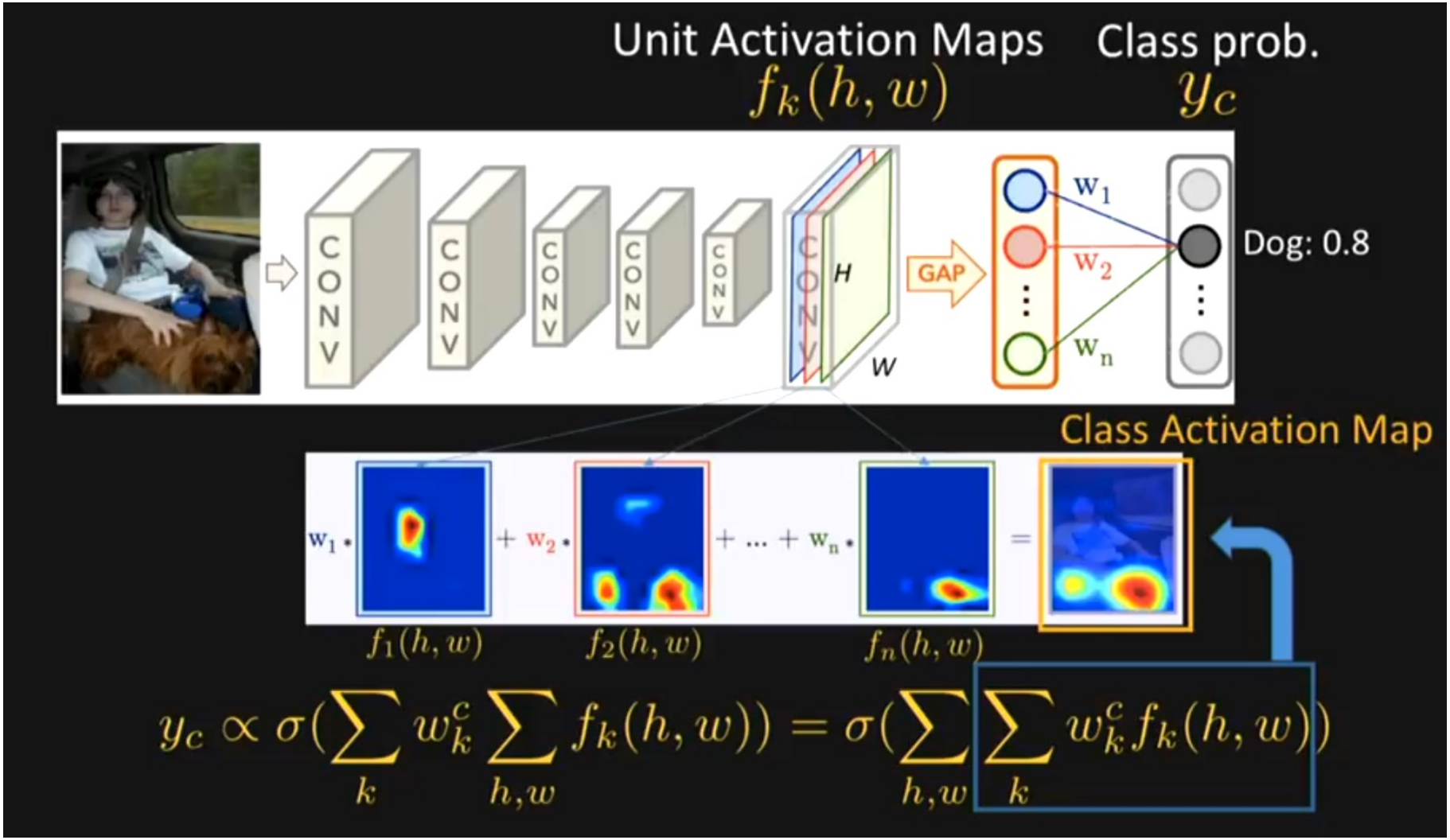
What the CNN is looking



Zhou, Bolei, et al. "Learning deep features for discriminative localization." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

Credit: Bolei Zhou

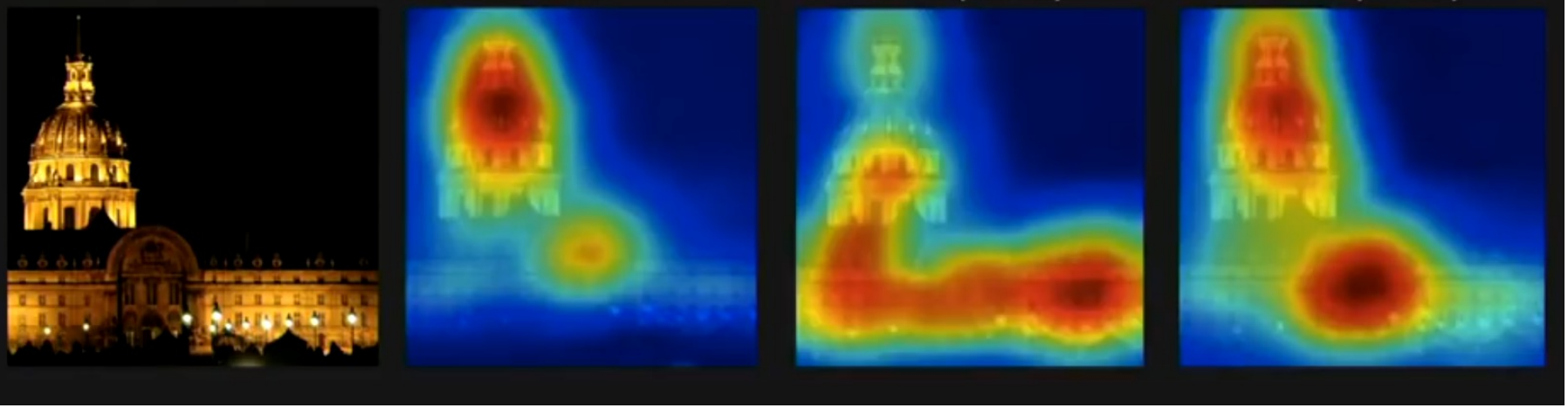
What the CNN is looking



Zhou, Bolei, et al. "Learning deep features for discriminative localization." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

Credit: Bolei Zhou

Class Activation Mapping: Explain Prediction of Deep Neural Network



Credit: Bolei Zhou

Credit: Bolei Zhou

Explain the failure cases

GT: House
Prediction: Sushi bar

Credit: Bolei Zhou

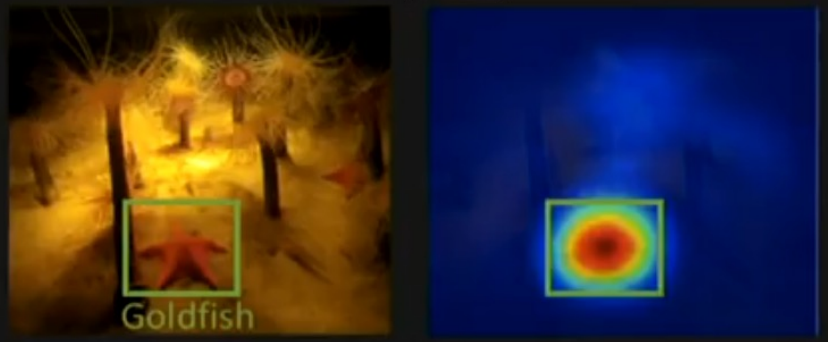
Explain the failure cases

Credit: Bolei Zhou

Using CAM for localization

Evaluation on Weakly-Supervised Localization

Prediction: Starfish (0.83)



Prediction: Tricycle (0.92)

Method	Supervision	Localization Accuracy(%)
Backpropagation	weakly	53.6
Our method	weakly	**62.9**
AlexNet	full	65.8

Result on ImageNet Localization Benchmark

Credit: Bolei Zhou

Limitation of CAM

- To apply CAM, any CNN-based network must change its architecture, where GAP is a must before the output layer
  - i.e., architectural changes and hence re-training is needed

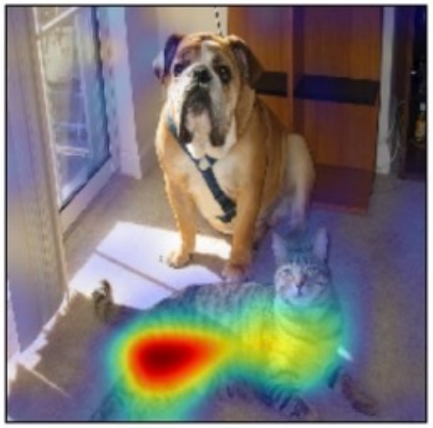
Grad-CAM (Gradient-weighted Class Activation Mapping)

- Grad-CAM (Gradient-weighted Class Activation Mapping) generalizes CAM for a wide variety of CNN-based architectures
  - i.e., without requiring architectural changes or re-training
- Characteristics
  - Without GAP layer, we need a way to define weights - w_k^c
  - Grad-CAM uses the gradients of any target class (e.g., “dog” in a classification network) flowing into the final convolutional layer, and derive summary statistics out of it to represent the weights (importance)

Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." Proceedings of the IEEE International Conference on Computer Vision. 2017.

Grad-CAM

(a) Original Image



(c) Grad-CAM 'Cat'

(g) Original Image

(i) Grad-CAM 'Dog'

Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." Proceedings of the IEEE International Conference on Computer Vision. 2017.

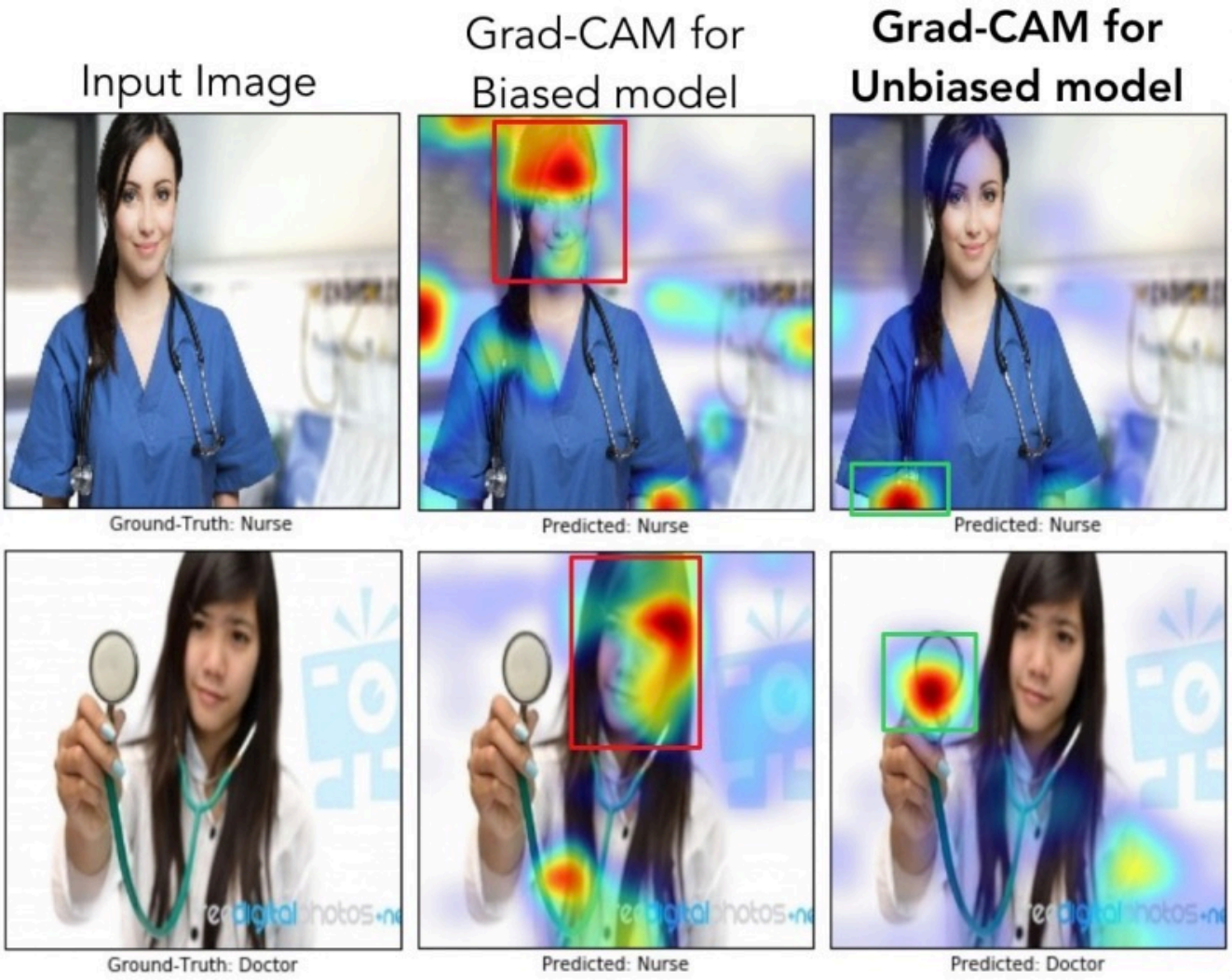
Grad-CAM for Weakly-supervised Localization

		Classification		Localization	
		Top-1	Top-5	Top-1	Top-5
VGG-16	Backprop [51]	30.38	10.89	61.12	51.46
	c-MWP [58]	30.38	10.89	70.92	63.04
	Grad-CAM (ours)	30.38	10.89	**56.51**	46.41
	CAM [59]	33.40	12.20	57.20	**45.14**
AlexNet	c-MWP [58]	44.2	20.8	92.6	89.2
	Grad-CAM (ours)	44.2	20.8	68.3	56.6
GoogleNet	Grad-CAM (ours)	31.9	11.3	60.09	49.34
	CAM [59]	31.9	11.3	60.09	49.34

Table 1: Classification and localization error % on ILSVRC-15 val (lower is better) for VGG-16, AlexNet and GoogleNet. We see that Grad-CAM achieves superior localization errors without compromising on classification performance.

Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." Proceedings of the IEEE International Conference on Computer Vision. 2017.

Grad-CAM for Identifying Bias in Dataset



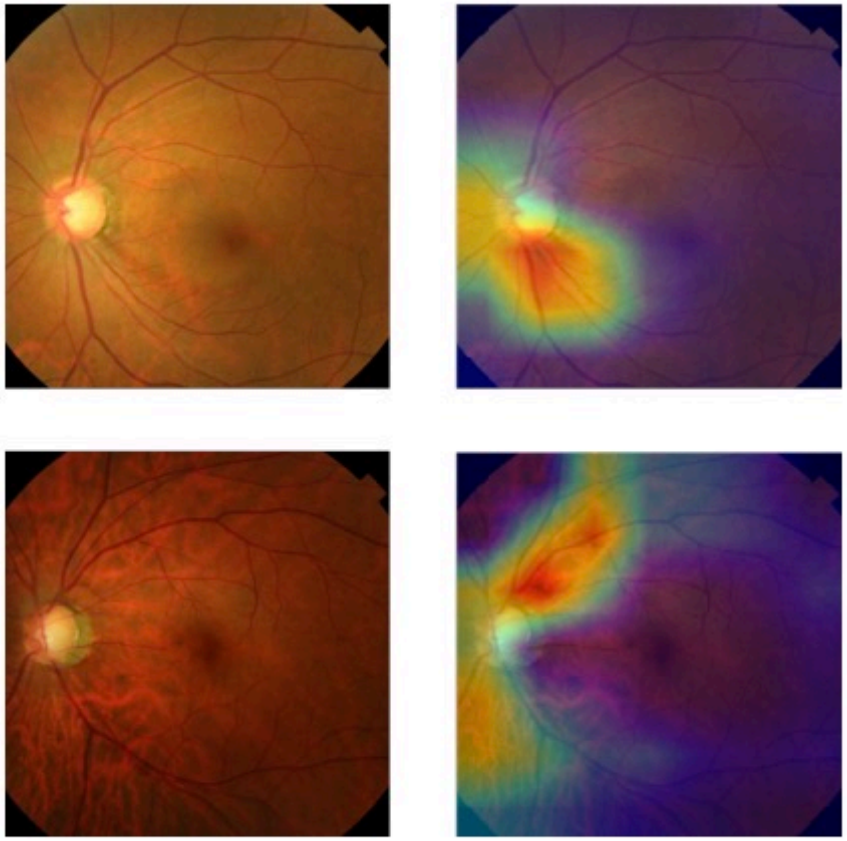
First row:
The biased model (model1) was looking at the face of the person to decide if the person was a nurse, whereas the unbiased model was looking at the short sleeves to make the decision.

Second row:
The biased model made the wrong prediction (misclassifying a doctor as a nurse) by looking at the face and the hairstyle, whereas the unbiased model made the right prediction looking at the white coat, and the stethoscope.

Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." Proceedings of the IEEE International Conference on Computer Vision. 2017.

Grad-CAM for Medical Image Analysis

- Grad-CAM for glaucoma diagnosis and localization



(a) Correctly localized suspicious areas.

Kim, Mijung, et al. "Web applicable computer-aided diagnosis of glaucoma using deep learning." arXiv preprint arXiv:1812.02405 (2018).

More reading

- Bau, David, et al. "Network dissection: Quantifying interpretability of deep visual representations." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

- Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." *Proceedings of the IEEE International Conference on Computer Vision*. 2017.

- Chattopadhay, Aditya, et al. "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks." *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018.

Implementation

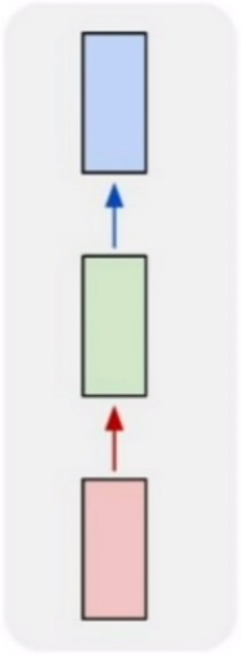
- CAM
  - https://github.com/zhoubolei/CAM
- Grad-CAM
  - https://github.com/ramprs/grad-cam/
  - Demo: http://gradcam.cloudcv.org/
- TorchCAM: class activation explorer
  - https://github.com/frgfm/torch-cam

Recurrent Neural Network (RNN)

Slides are adapted from Hung-yi Lee and Feifei Li

“Vanilla” Neural Network

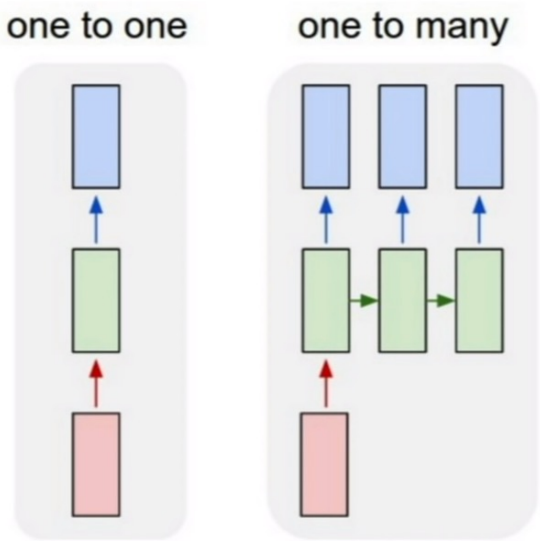
one to one



$f($  $)=$ “Cat”

Vanilla Neural Networks

Recurrent Neural Networks: Process Sequences



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

Example?

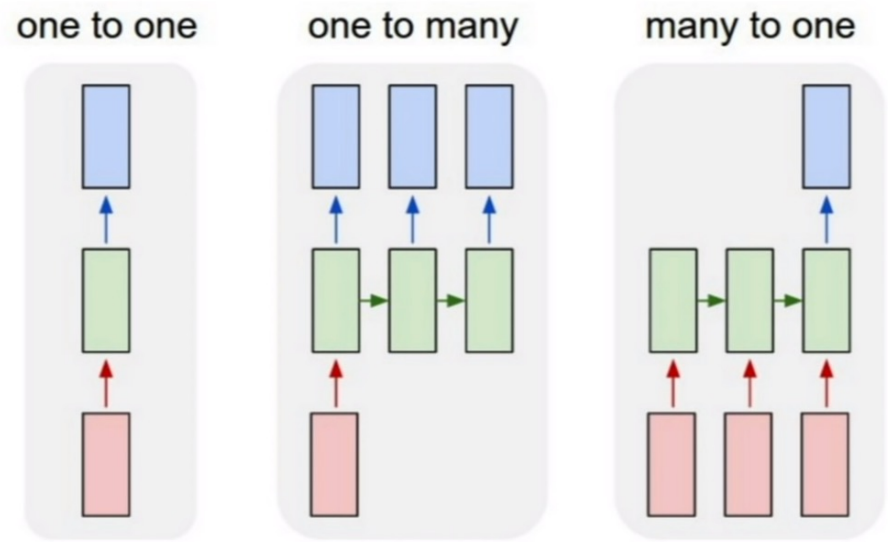

A person riding a motorcycle on a dirt road.

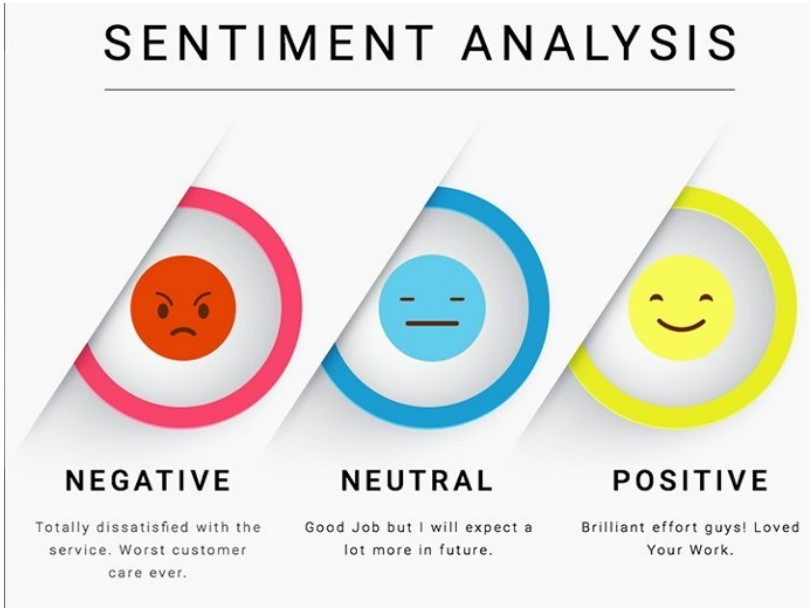

Two dogs play in the grass.


A group of young people playing a game of frisbee.


Two hockey players are fighting over the puck.

Recurrent Neural Networks: Process Sequences

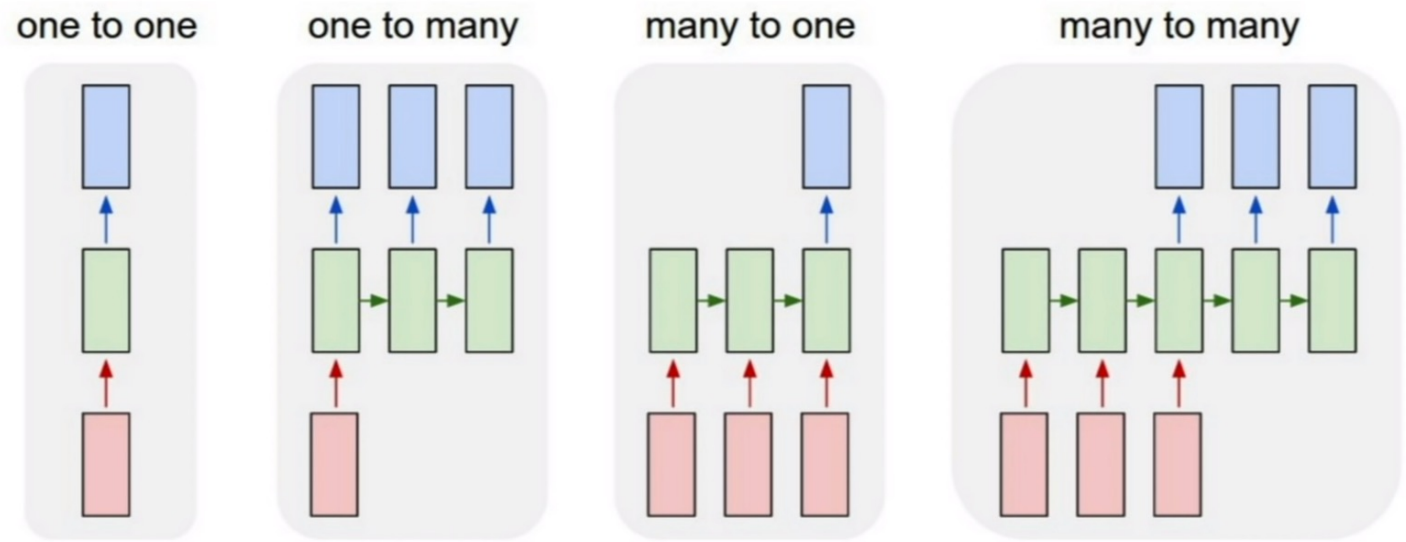




e.g. Sentiment Classification
Sequence of words -> sentiment

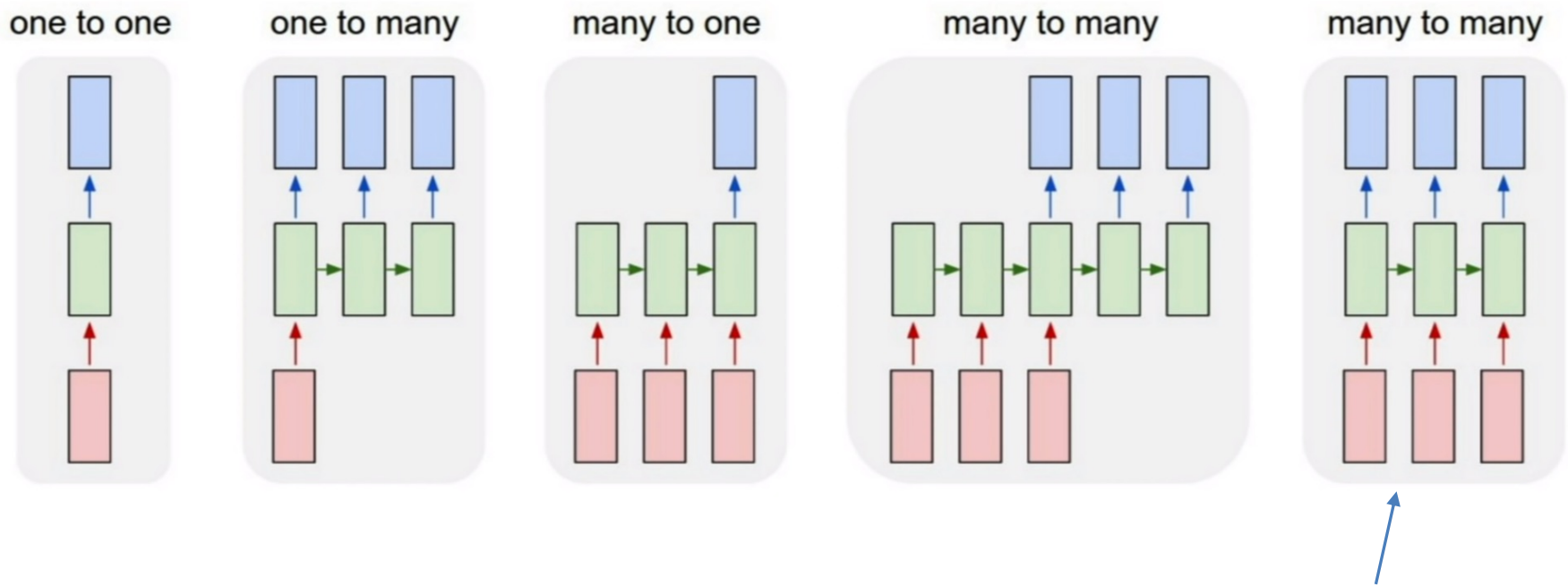
Video action recognition
(frames) -> action label

Recurrent Neural Networks: Process Sequences



e.g. Machine translation
Sequence of words ->
Sequence of words

Recurrent Neural Networks: Process Sequences



e.g. Video classification on frames

Sequences in the Wild

Audio

What are Sequences?

English: I love pickles and onions on my sandwich.

Stock Prediction: <date1, value1>, <date2, value2> ... <dateN, valueN>

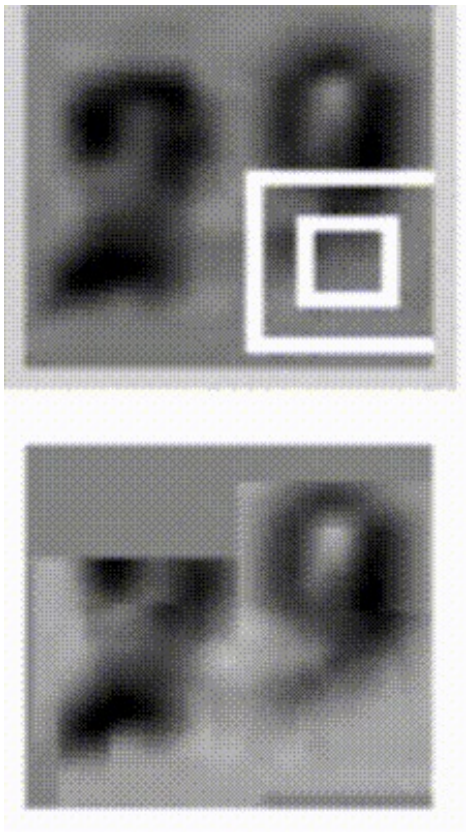
- In general, It is a temporally ordered set of data points.
- RNNs find themselves useful because of the following mappings:

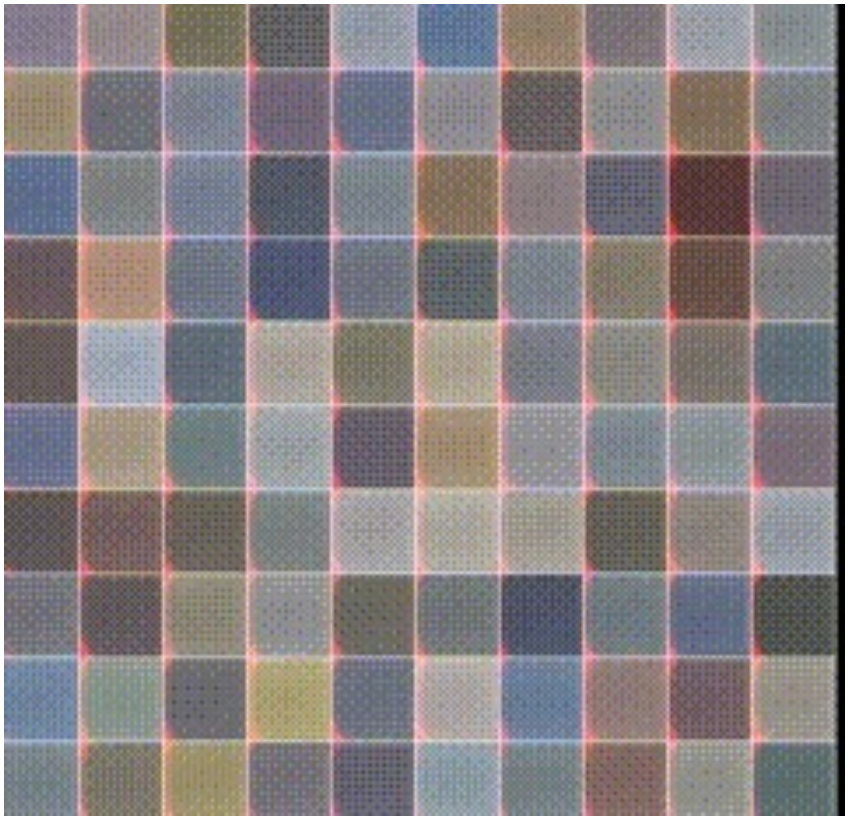
$$f: seq \rightarrow \mathbb{R}^D$$

$$f: \mathbb{R}^D \rightarrow seq$$

$$f: seq \rightarrow seq$$

Recurrent Neural Network



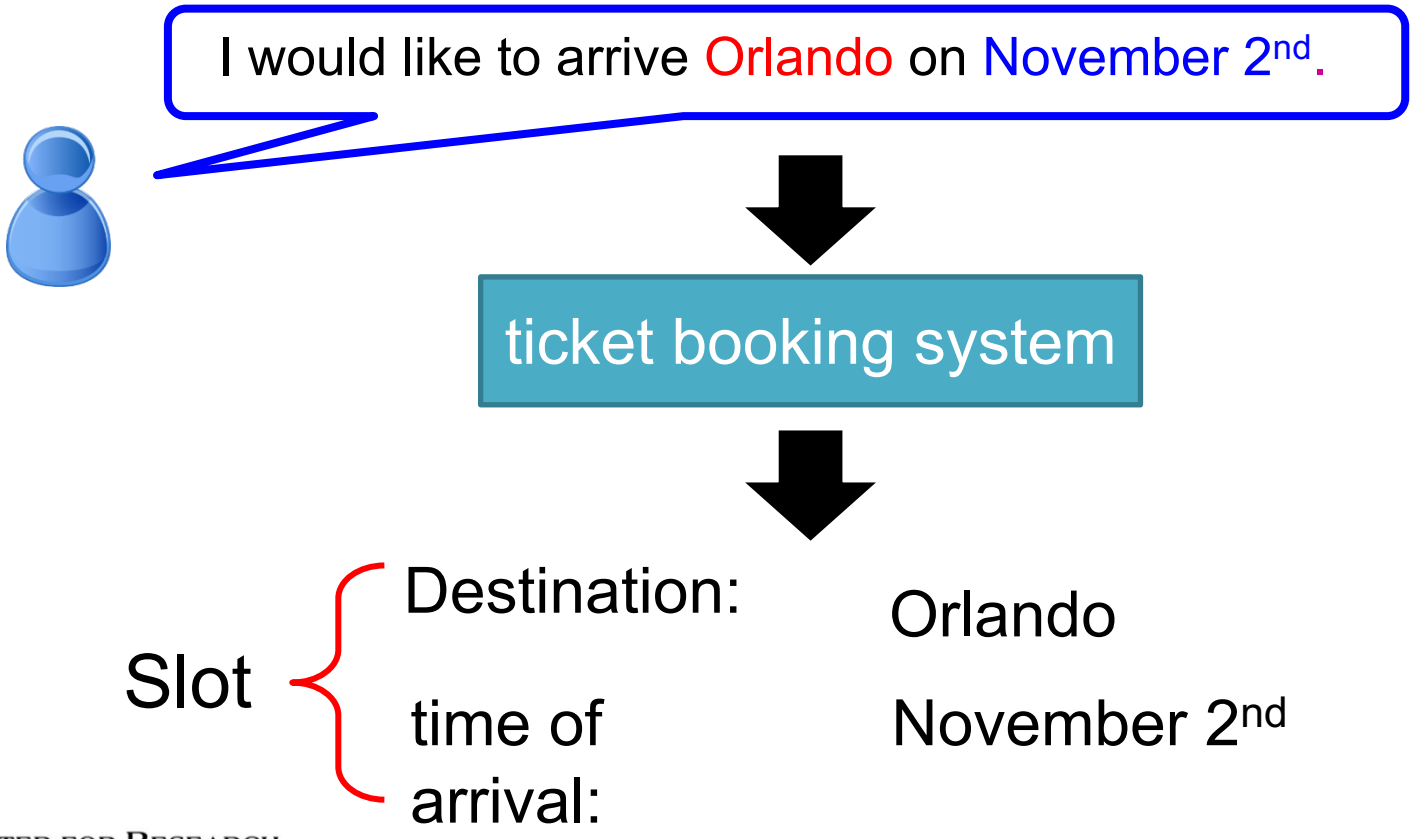


Left: RNN learns to read house numbers.
Right: RNN learns to paint house numbers.

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Example Application

- Slot Filling

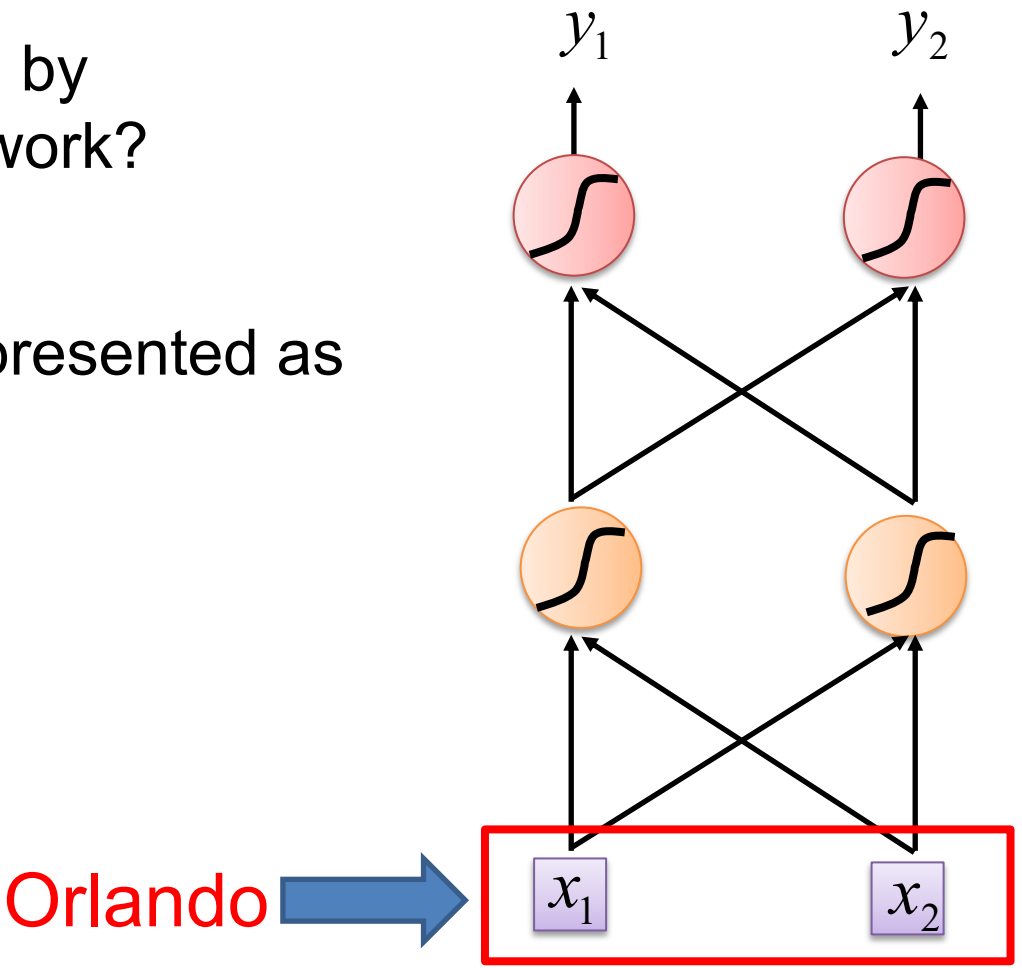


Example Application

Solving slot filling by Feedforward network?

Input: a word

(Each word is represented as a vector)



Example Application

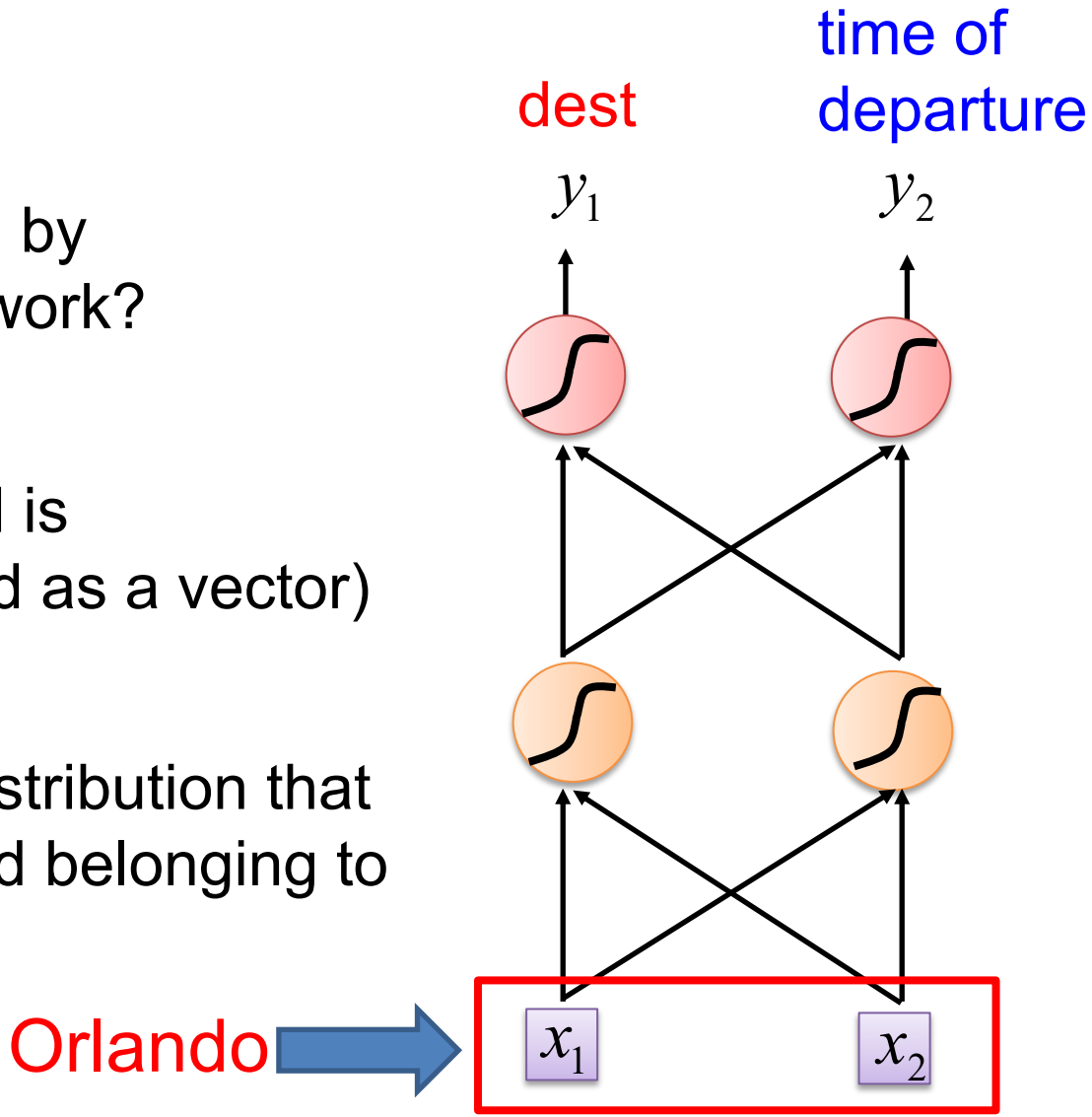
Solving slot filling by Feedforward network?

Input: a word

(Each word is represented as a vector)

Output:

Probability distribution that the input word belonging to the slots



Example Application

arrive Orlando on November 2nd

other dest other time time

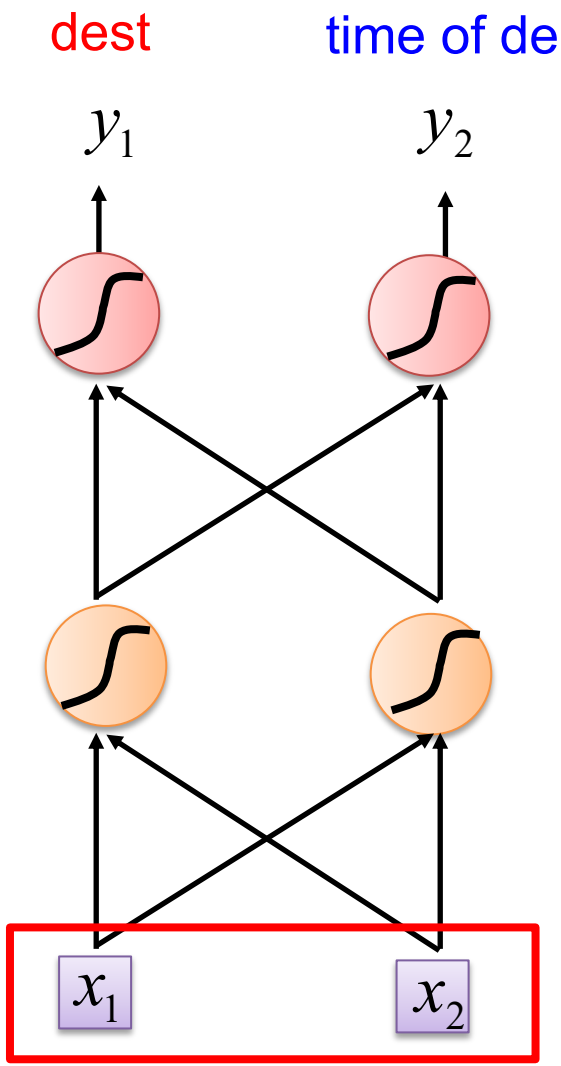
Problem?

leave Orlando on November 2nd

place of departure

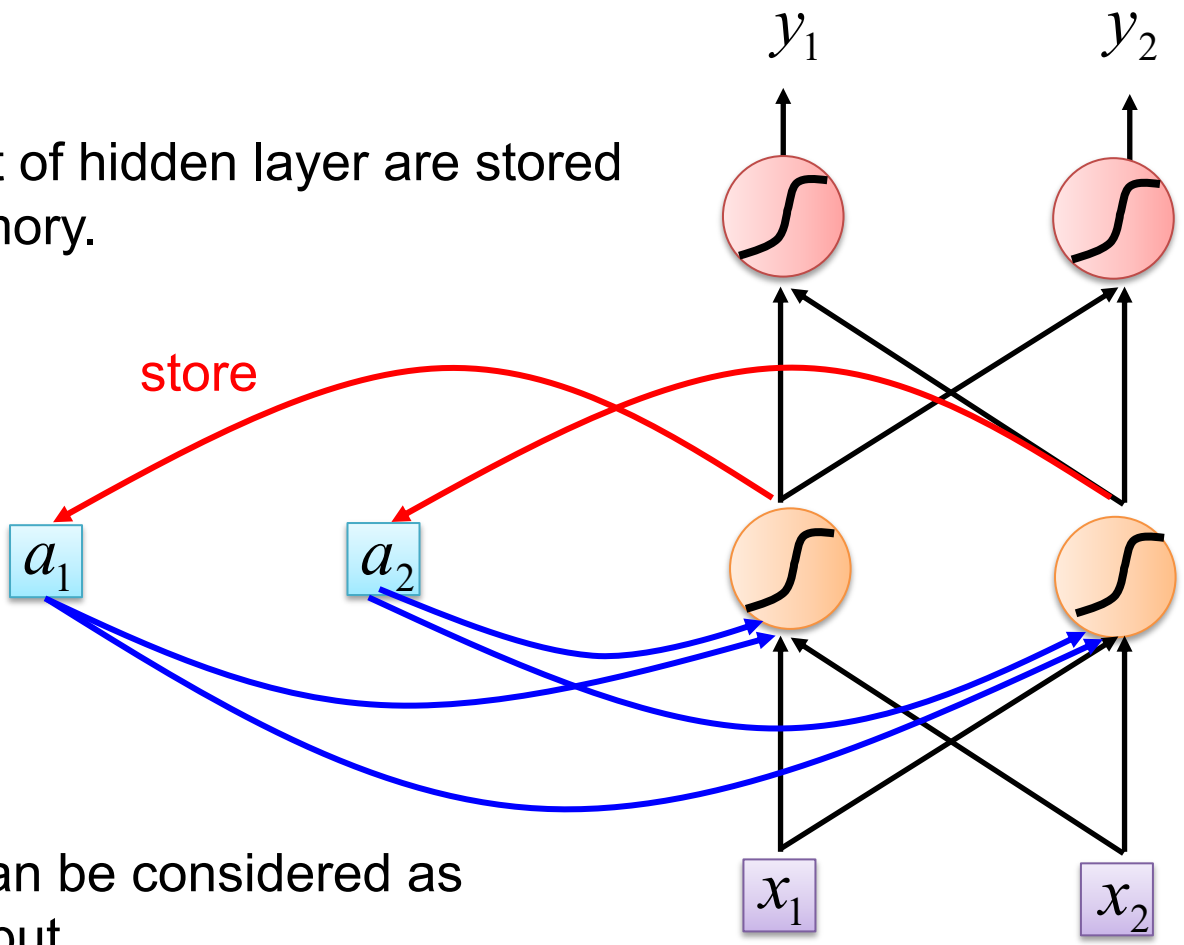
Neural network needs memory!

Orlando



Recurrent Neural Network (RNN)

The output of hidden layer are stored in the memory.

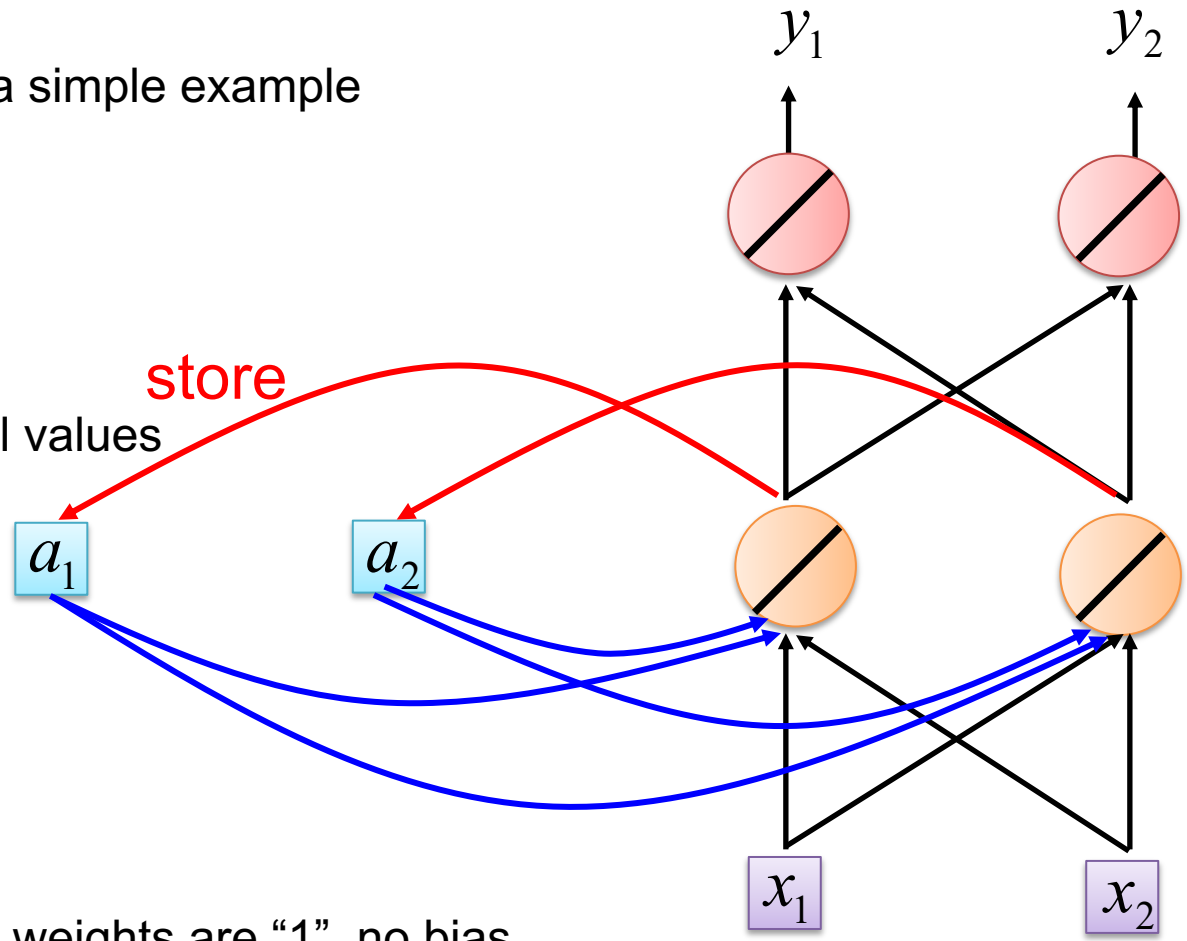


Memory can be considered as another input.

RNN

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}2\\2\end{bmatrix}$ ……

Run a simple example



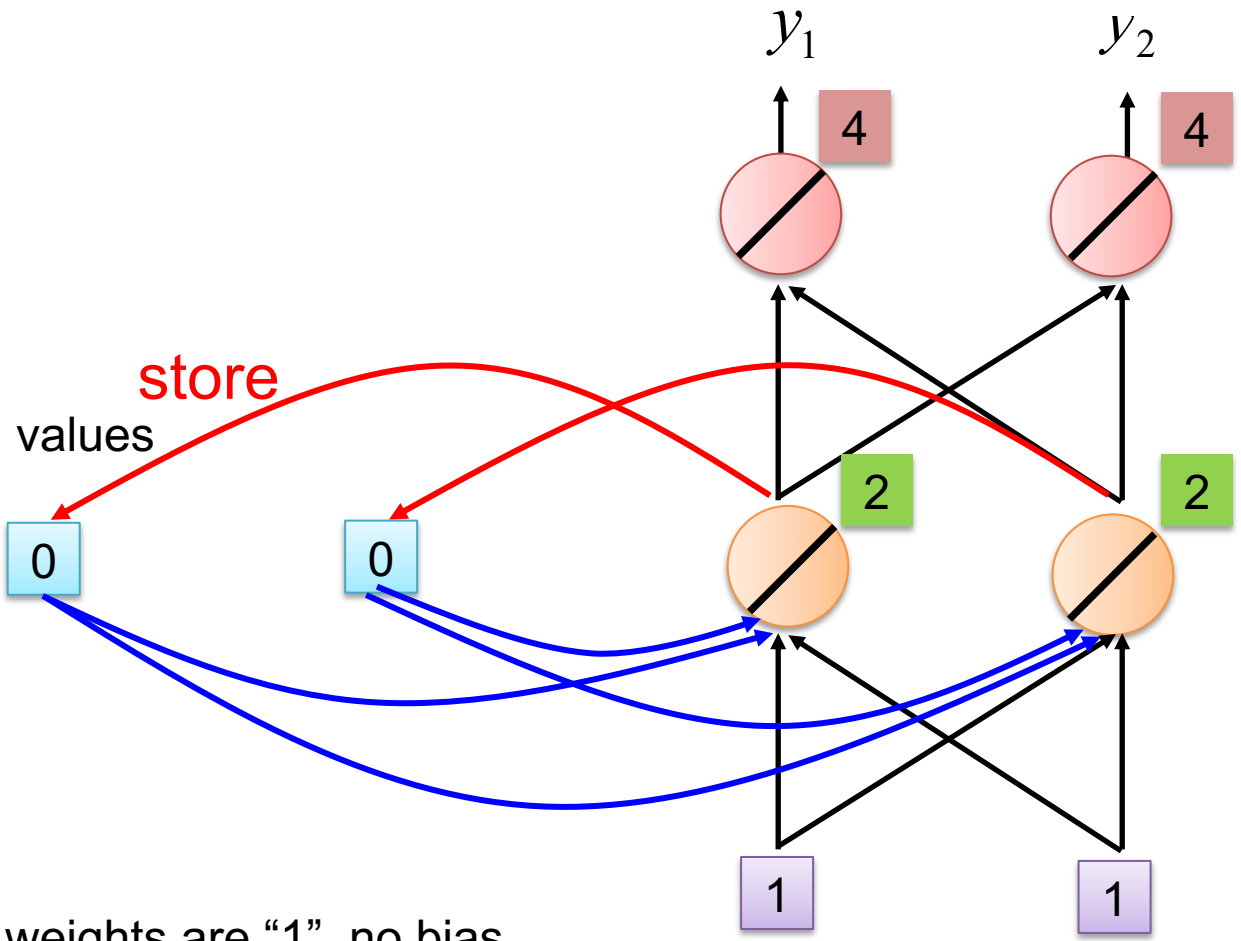
Given initial values to memory

All the weights are “1”, no bias
All activation functions are linear

RNN

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}$

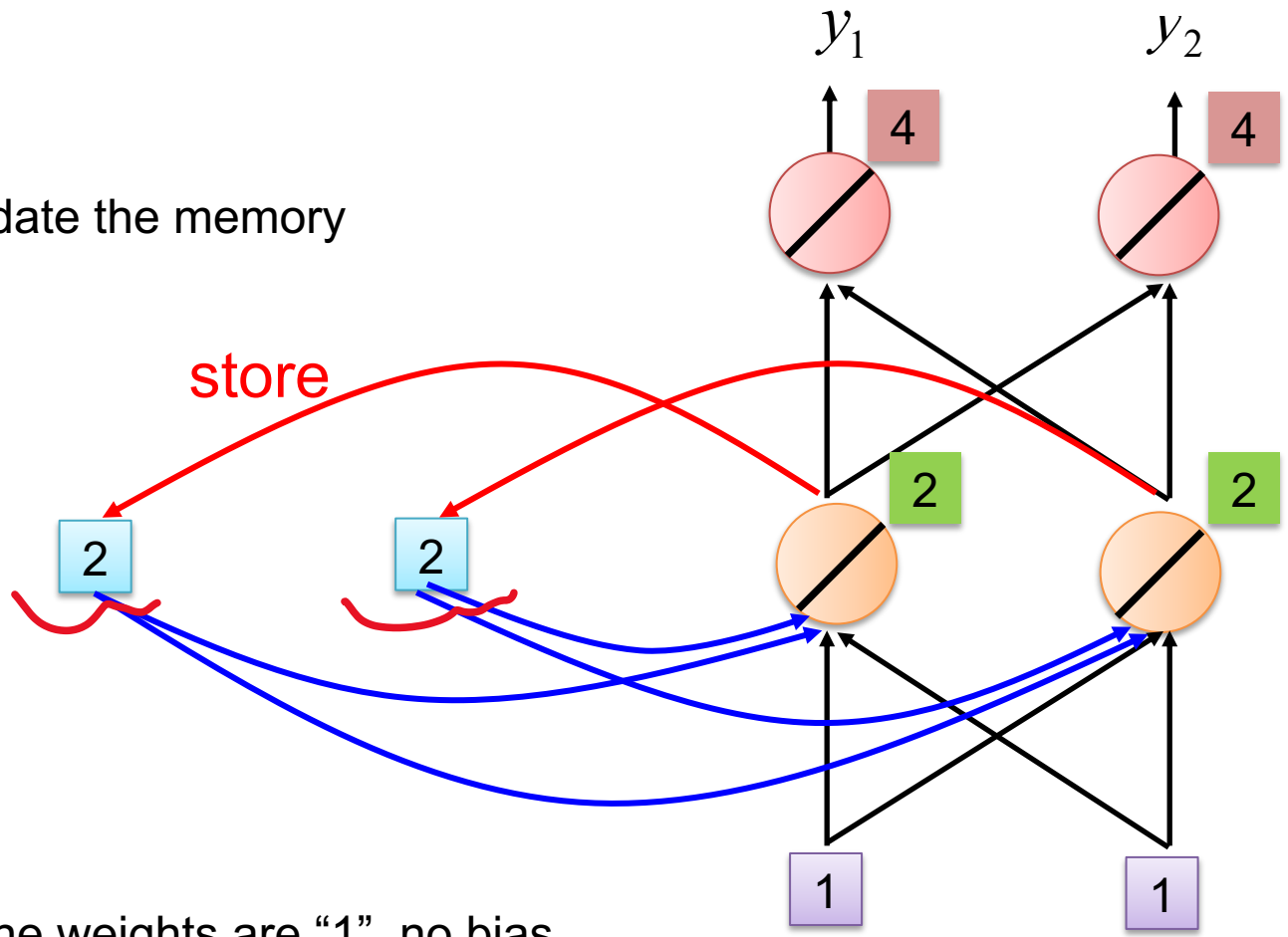


All the weights are “1”, no bias
All activation functions are linear

RNN

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}$



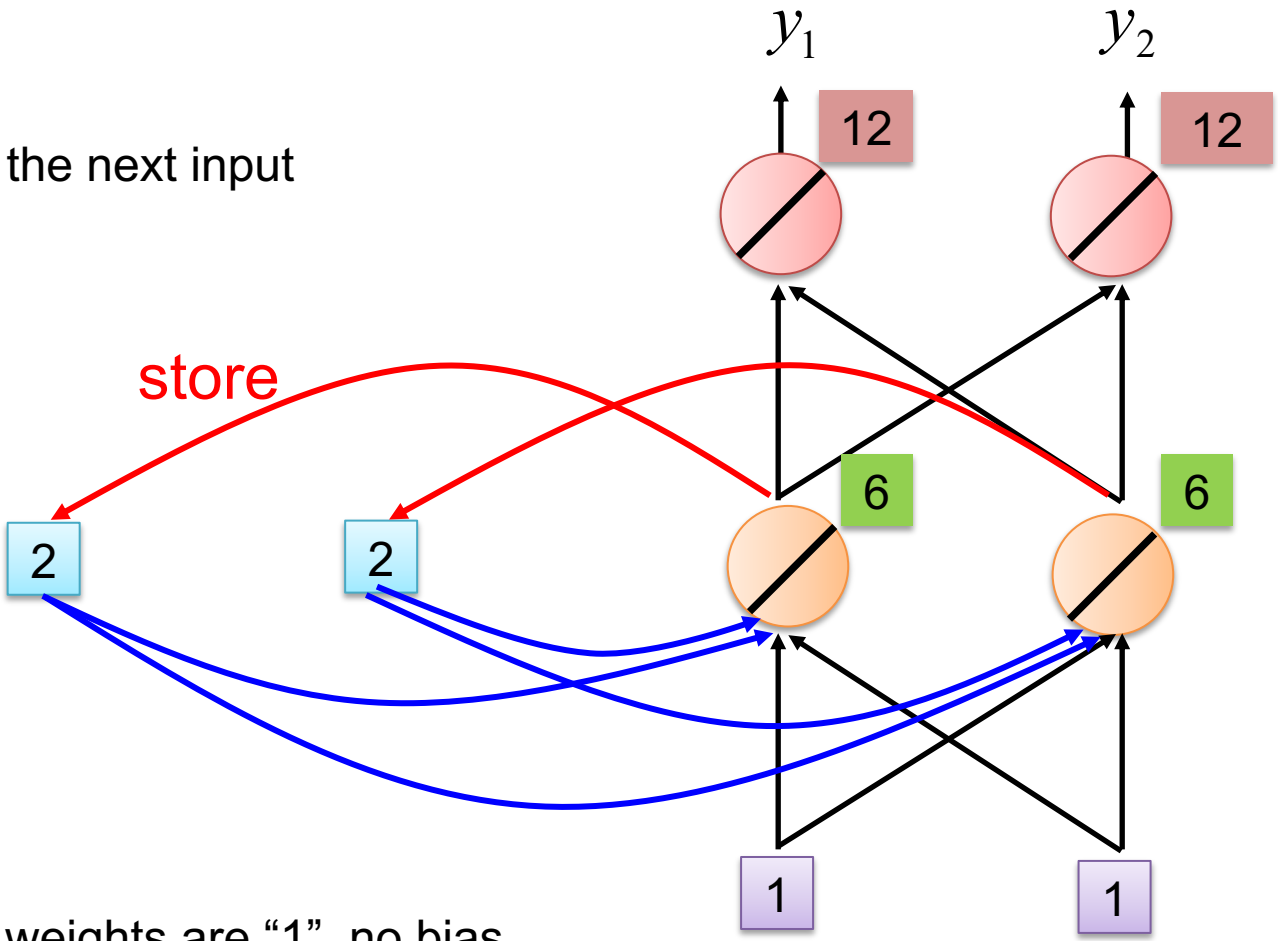
All the weights are “1”, no bias
All activation functions are linear

RNN

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}$ $\begin{bmatrix}12\\12\end{bmatrix}$

Take the next input

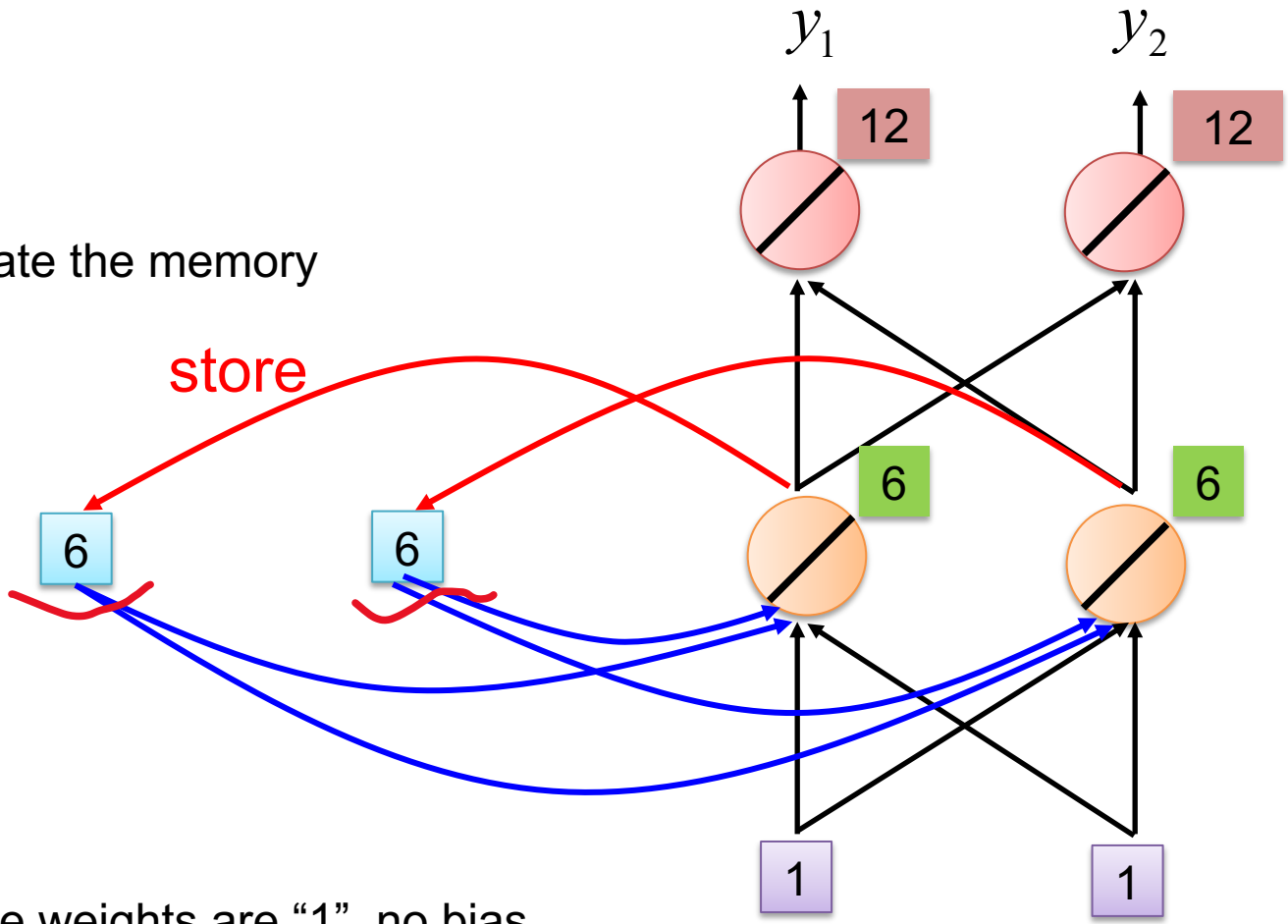


All the weights are “1”, no bias
All activation functions are linear

RNN

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}$ $\begin{bmatrix}12\\12\end{bmatrix}$



Update the memory

store

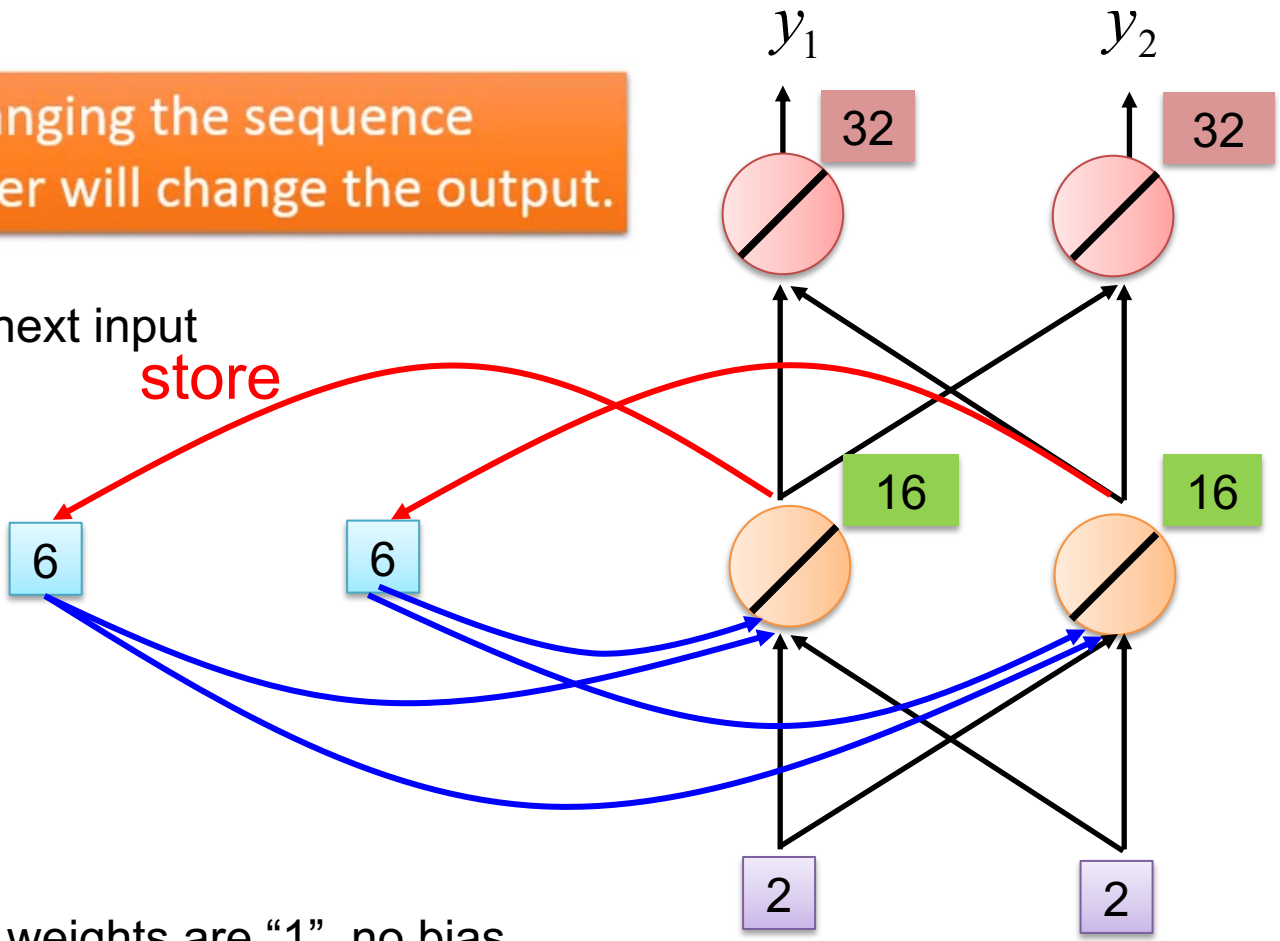
All the weights are “1”, no bias
All activation functions are linear

RNN

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}\begin{bmatrix}12\\12\end{bmatrix}\begin{bmatrix}32\\32\end{bmatrix}$

Changing the sequence order will change the output.



Take the next input

All the weights are “1”, no bias
All activation functions are linear

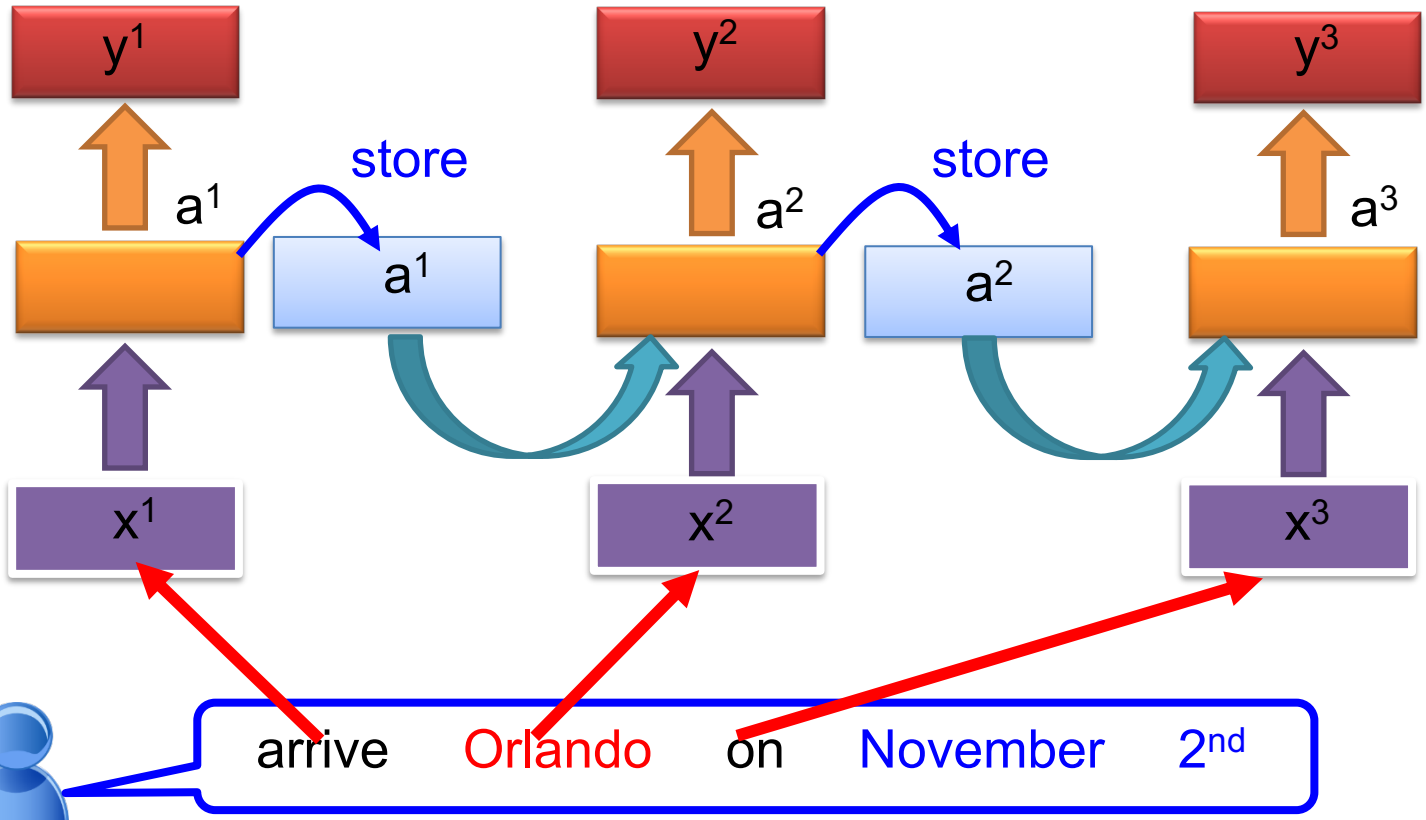
RNN

The same network is used again and again.

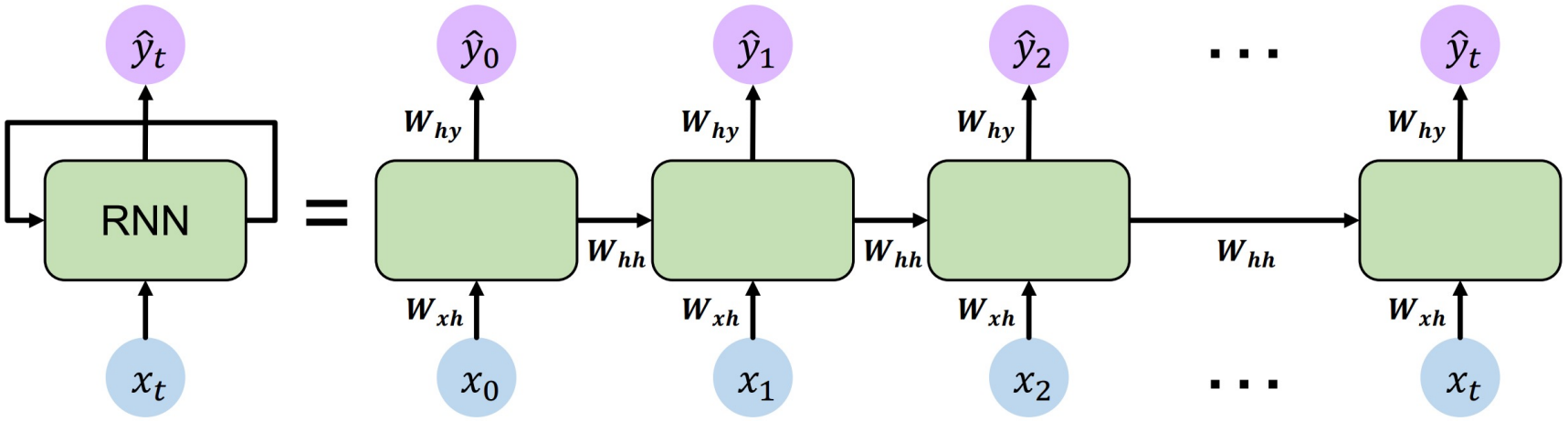
Probability of "arrive" in each slot

Probability of "Orlando" in each slot

Probability of "on" in each slot

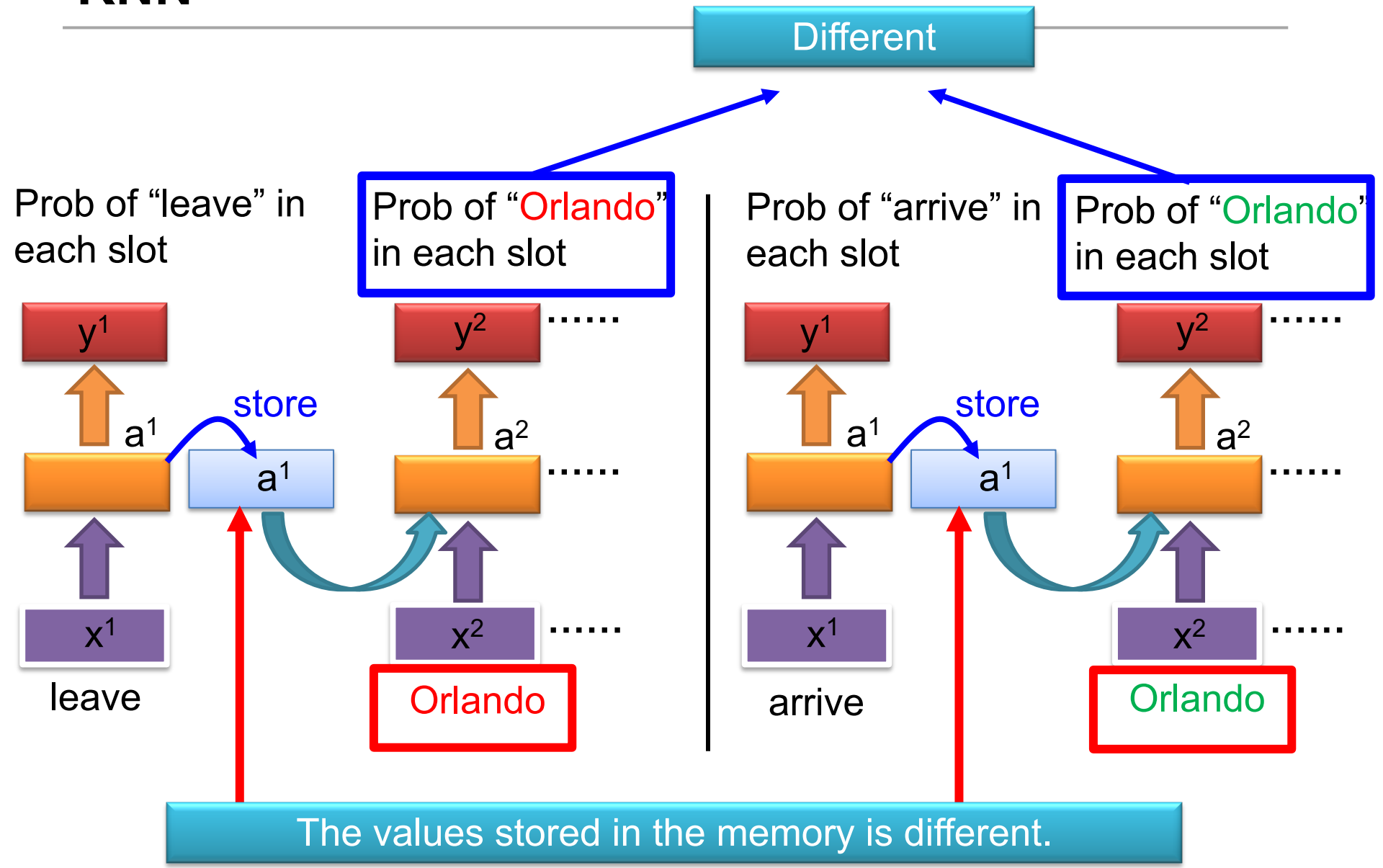


RNNs: computational graph across time

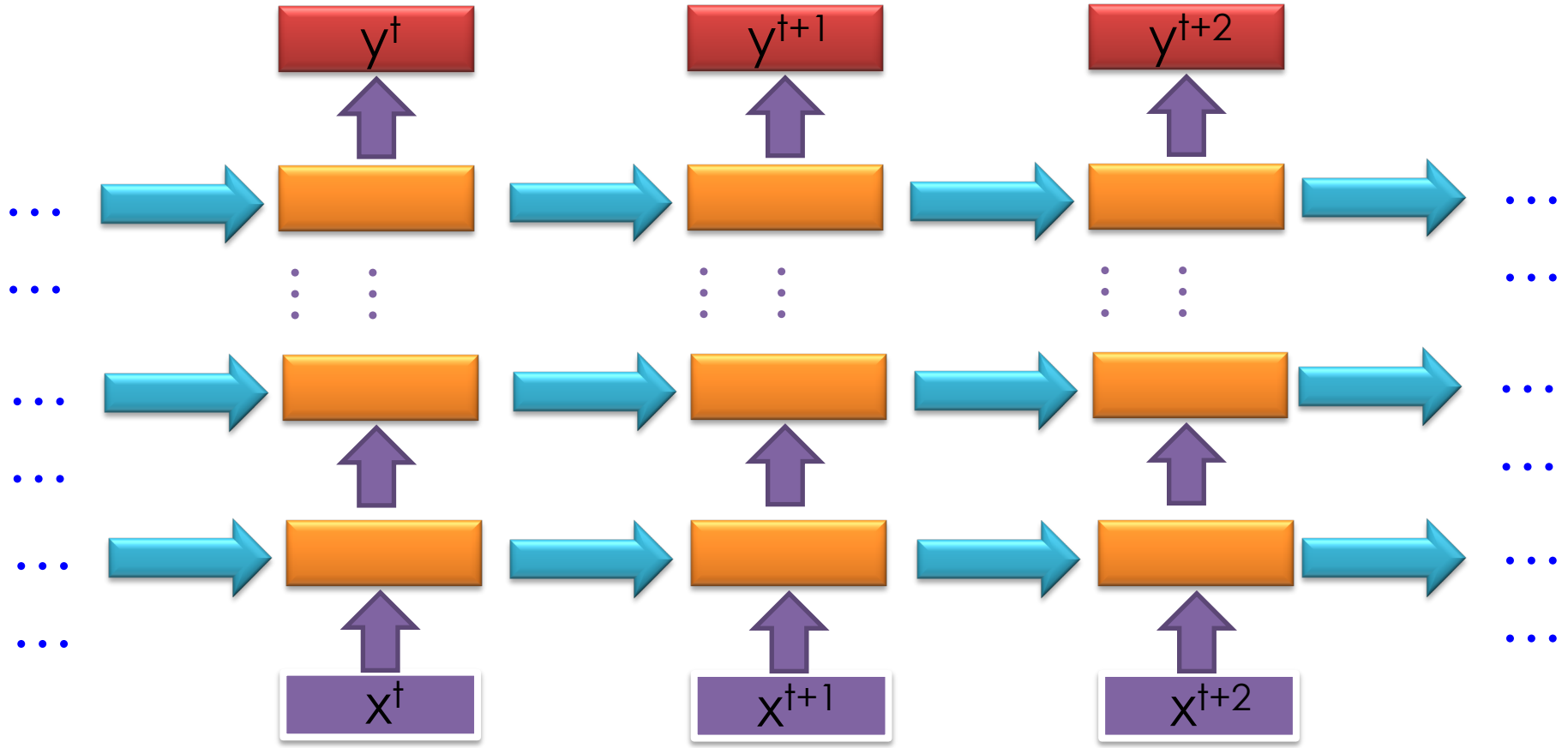


Unrolling

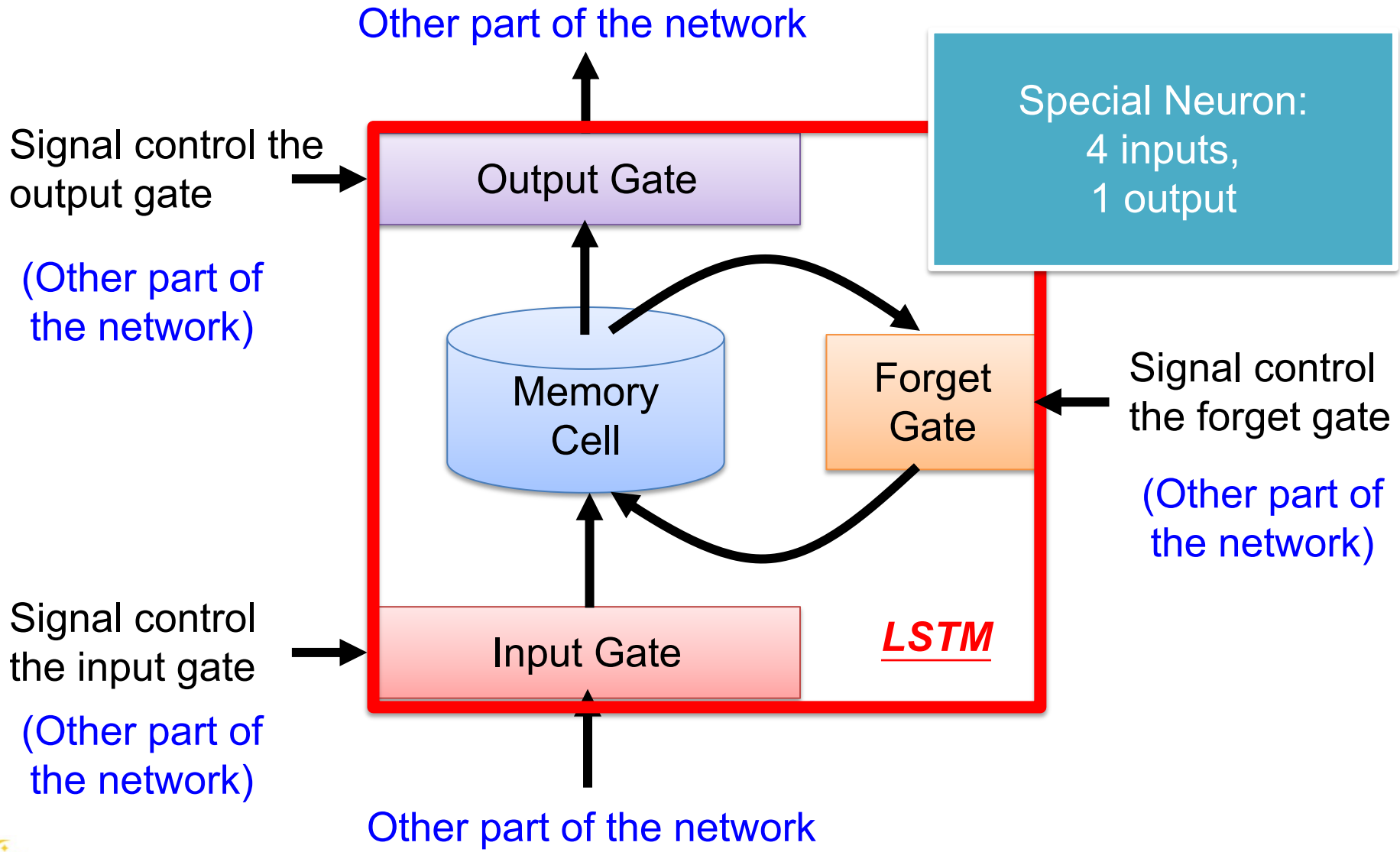
RNN


Different
Prob of "leave" in each slot
Prob of "Orlando" in each slot
Prob of "arrive" in each slot
Prob of "Orlando" in each slot
y1
y2
a1
a2
store
x1
x2
leave
Orlando
arrive
Orlando
The values stored in the memory is different.

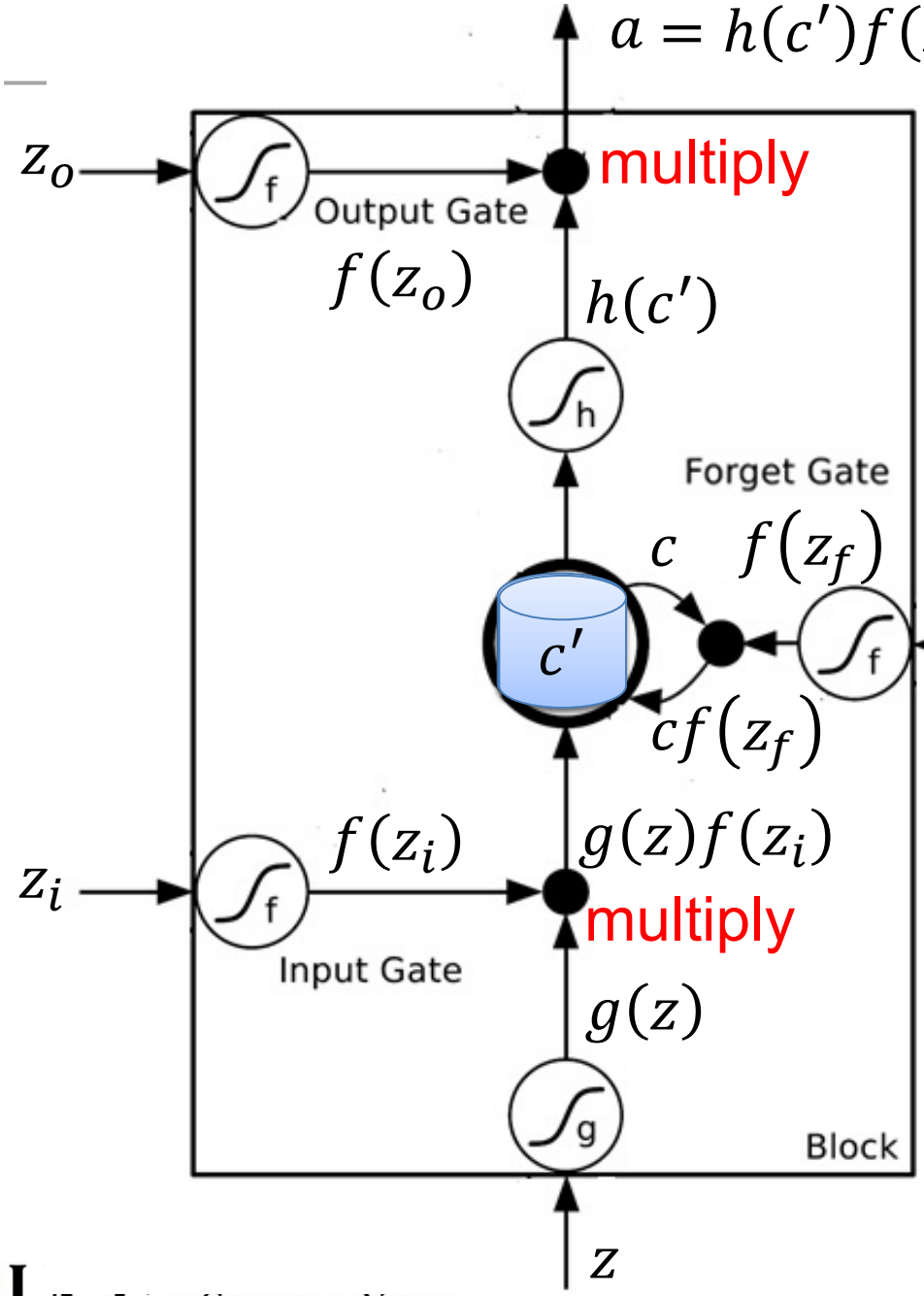
Of course, it can be deep …



Long Short-term Memory (LSTM)



Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.



Activation function f is usually a sigmoid function

Between 0 and 1

Mimic open and close gate

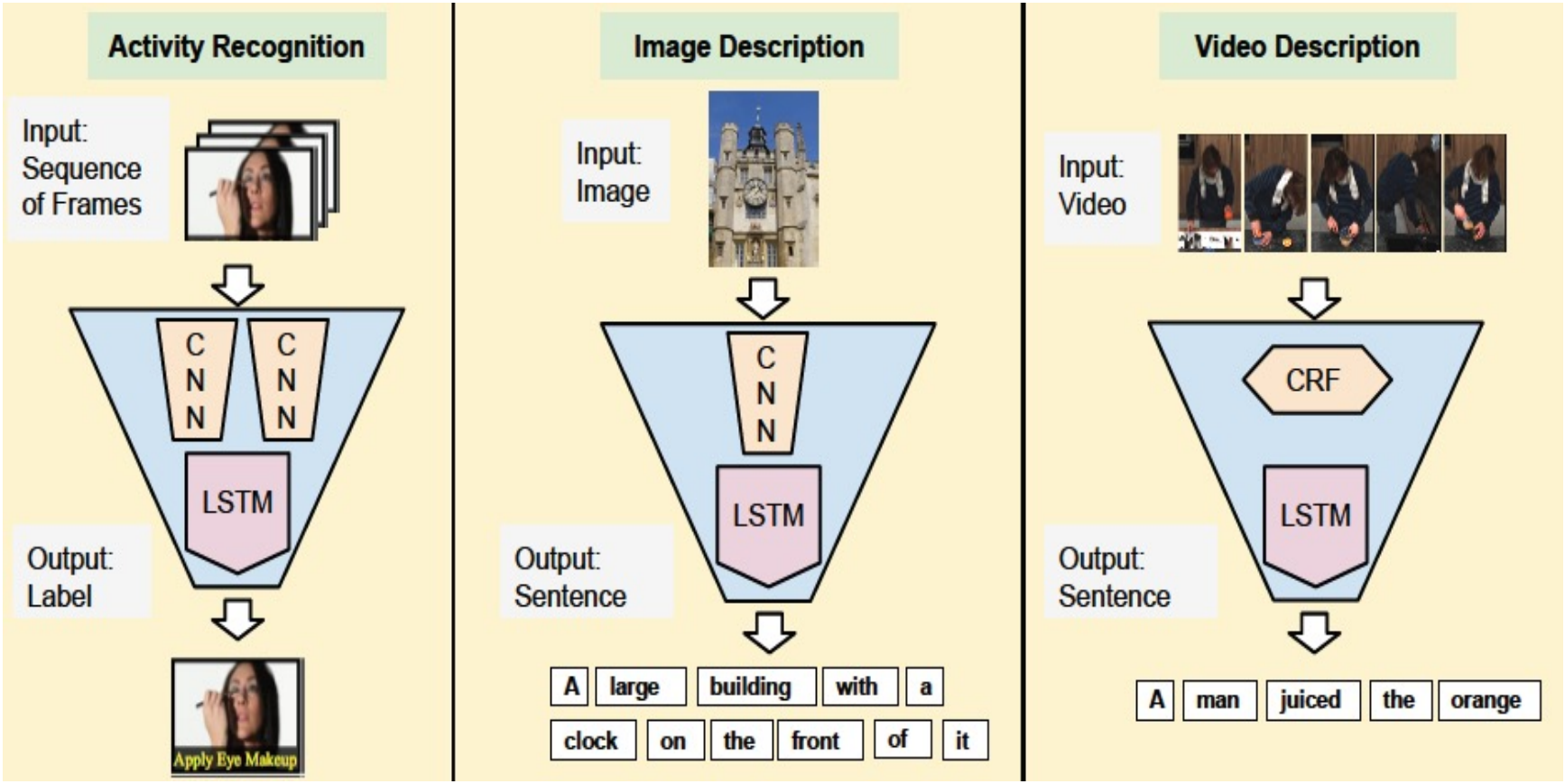
$$c' = g(z)f(z_i) + cf(z_f)$$

Limitations of RNNs

- Computations for over positions cannot be parallelized
- Long-range interactions are bottlenecked by a fixed size memory

Example applications

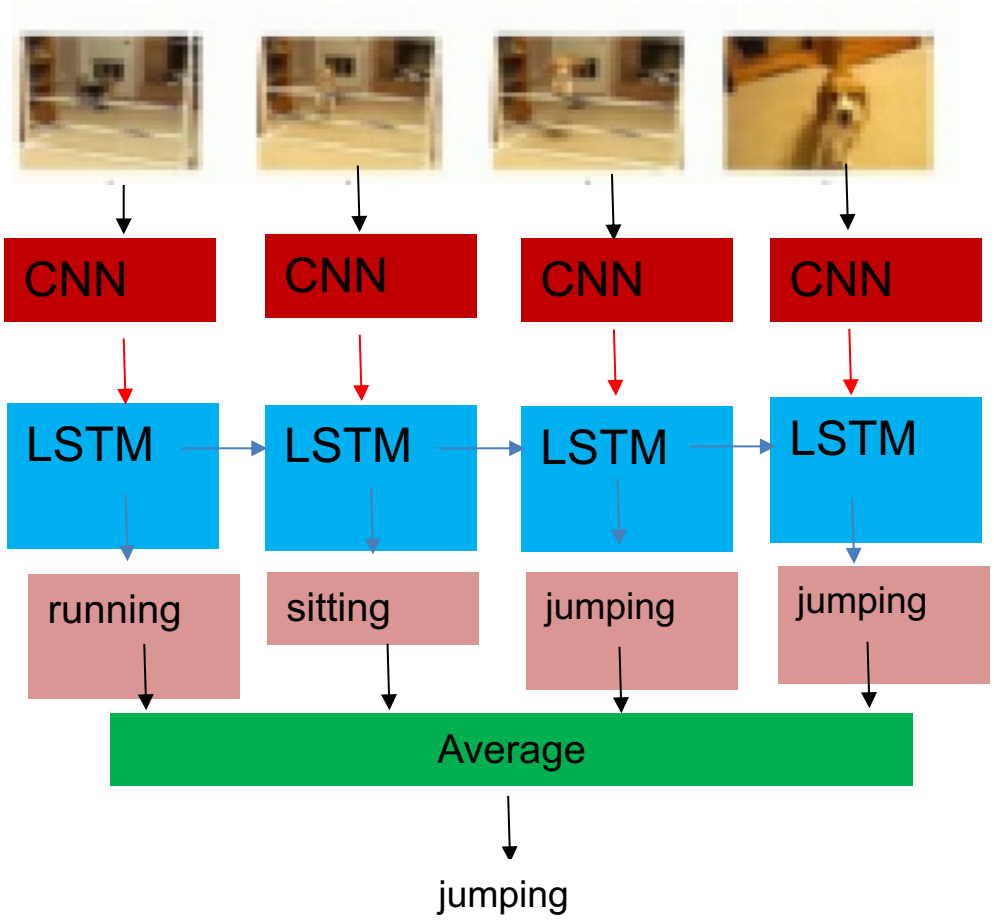
Sequential inputs /outputs



An example for image captioning: https://blog.clairvoyantsoft.com/image-caption-generator-535b8e9a66ac

Image credit: main paper

Activity recognition



Demos

- MNIST with LSTM
  - https://www.kaggle.com/muhammedfathi/mnist-with-rnn-and-lstm
- https://www.tensorflow.org/guide/keras/rnn
- https://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/
- More examples here:
  - https://keras.io/examples/

Thank you!

Question?

References and Slide Credits

- Many slides are adapted from the existing teaching or tutorial slides by Hung-yi Lee, Andrew Ng, Alexander Amini, Lex Fridman, Stanford course - CS231n: Convolutional Neural Networks for Visual Recognition, and many others
- Special thanks to Dr. Hung-yi Lee for making his machine learning course slides and materials available
- Alexander Amini, MIT 6.S191 Introduction to Deep Learning: http://introtodeeplearning.com/
  - Youtube videos: https://www.youtube.com/watch?v=5tvmMX8r_OM&list=PLtBw6njQRU-rwp5__7C0oIVt26ZgjG9NI&index=1
- Lex Fridman, MIT Deep Learning and Artificial Intelligence Lectures: https://deeplearning.mit.edu/ https://www.youtube.com/watch?v=O5xeyoRL95U