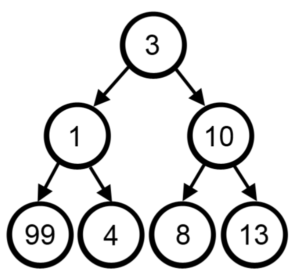
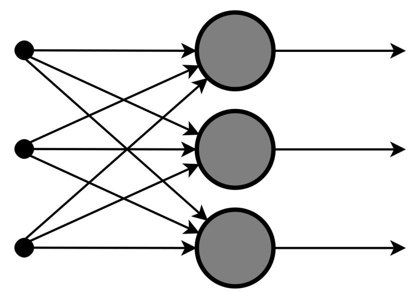
CSE 40171: Artificial Intelligence





Artificial Neural Networks with Anatomical Fidelity:
Recurrence in Artificial and Biological Networks

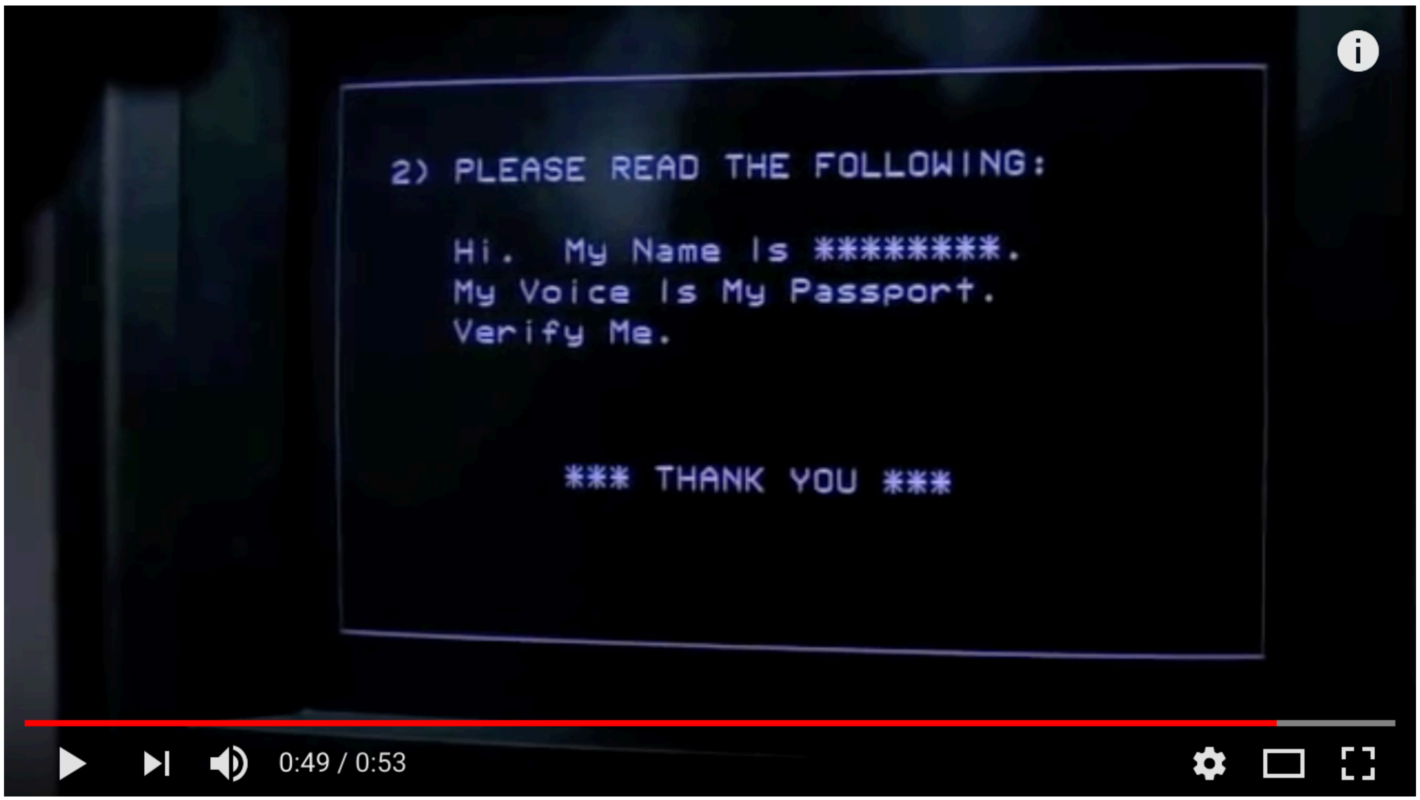
Homework #6 has been released
It is due at 11:59PM on 11/22

Project Updates are Due on 11/25 at 11:59PM

(See Course Website for Instructions)

Let’s turn our attention back to artificial neural networks for a moment…

How do we parse audio data?

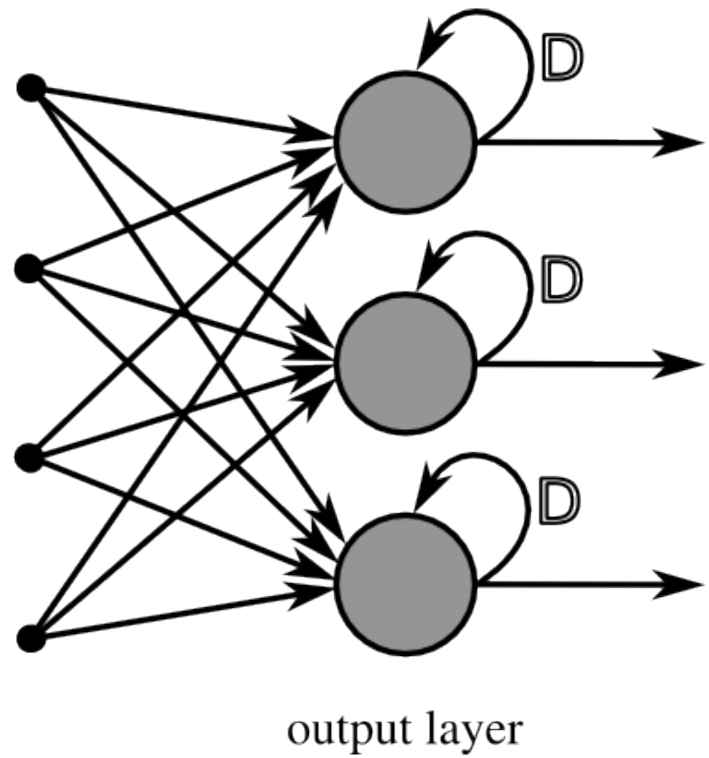


https://www.youtube.com/watch?v=-zVgWpVXb64

How about text?

Arms, and the man I sing, who, forc'd by fate,
And haughty Juno's unrelenting hate,
Expell'd and exil'd, left the Trojan shore.
Long labors, both by sea and land, he bore,
And in the doubtful war, before he won
The Latian realm, and built the destin'd town;
His banish'd gods restor'd to rites divine,
And settled sure succession in his line,
From whence the race of Alban fathers come,
And the long glories of majestic Rome.
O Muse! the causes and the crimes relate;
What goddess was provok'd, and whence her hate;
For what offense the Queen of Heav'n began
To persecute so brave, so just a man;
Involv'd his anxious life in endless cares,
Expos'd to wants, and hurried into wars!

Recurrent Networks



Architecture of a Traditional RNN

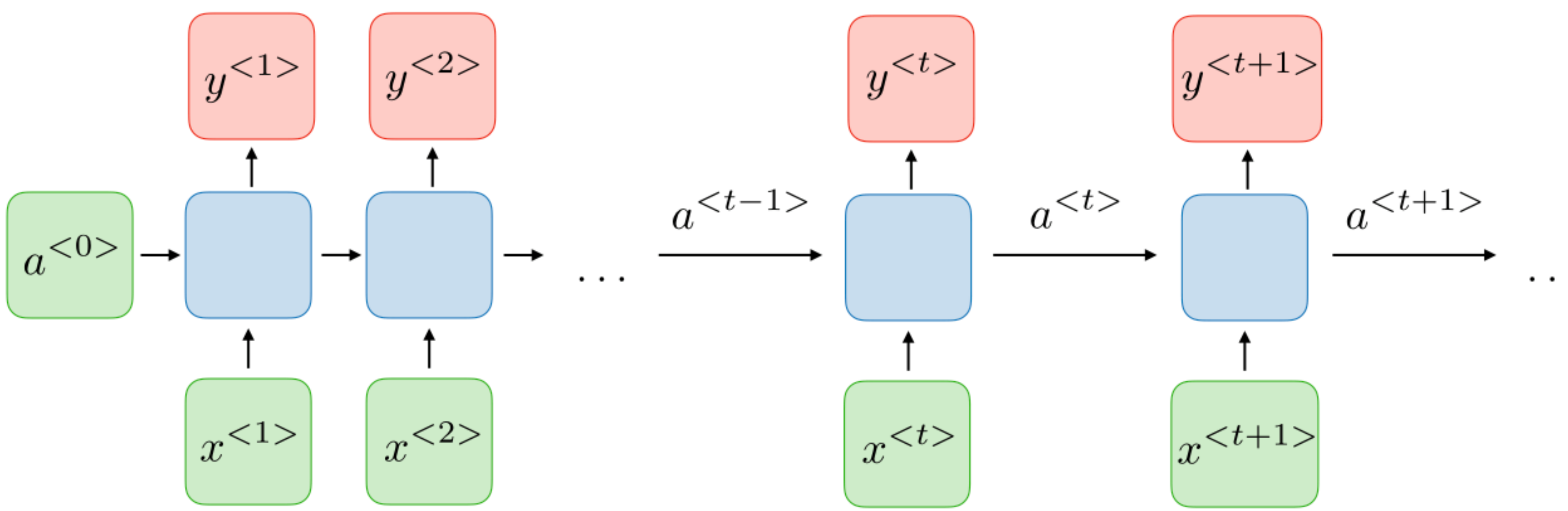


Image Credit: Afshine Amidi and Shervine Amidi Stanford CS 230

At each timestep t:

Activation $a^{<t>}$:

$$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$$

activation function **coefficient**

Output $y^{<t>}$:

$$y^{<t>} = g_2(W_{ya}a^{<t>} + b_y)$$

At each timestep t:

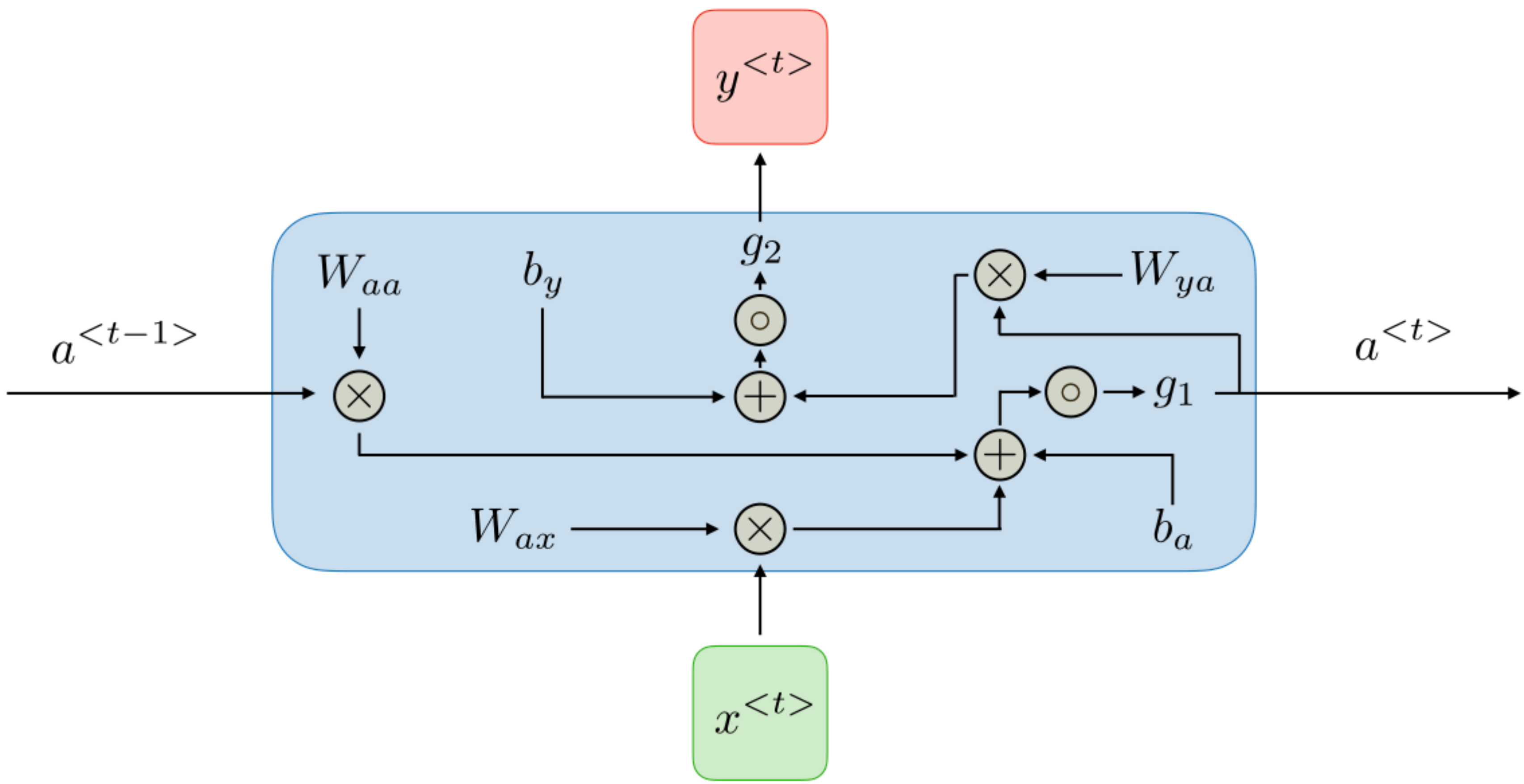
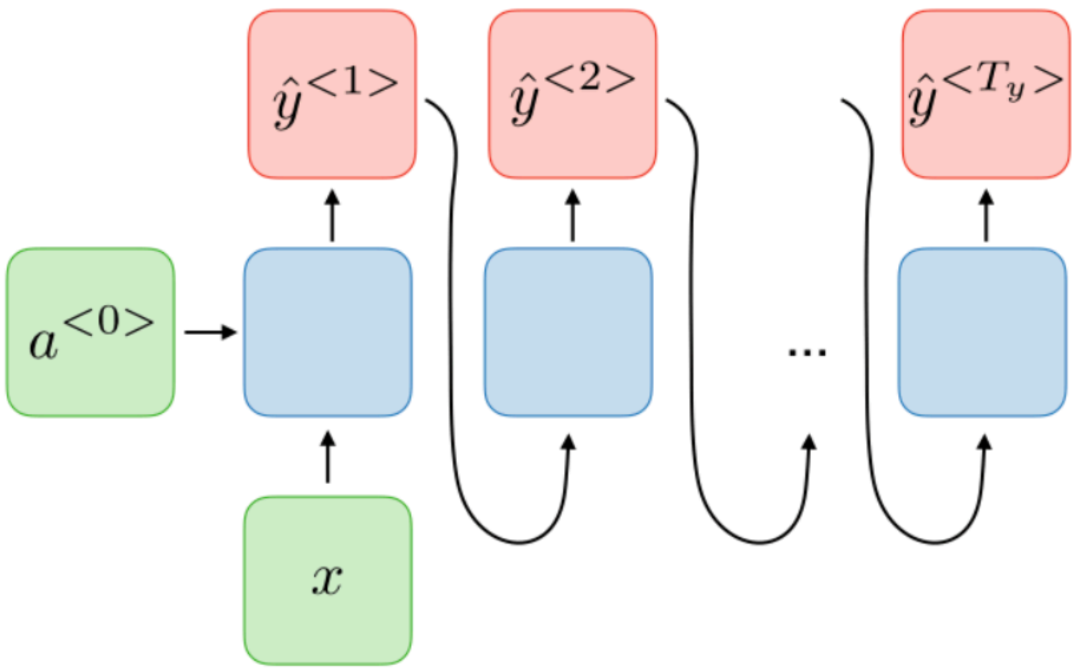


Image Credit: Afshine Amidi and Shervine Amidi Stanford CS 230

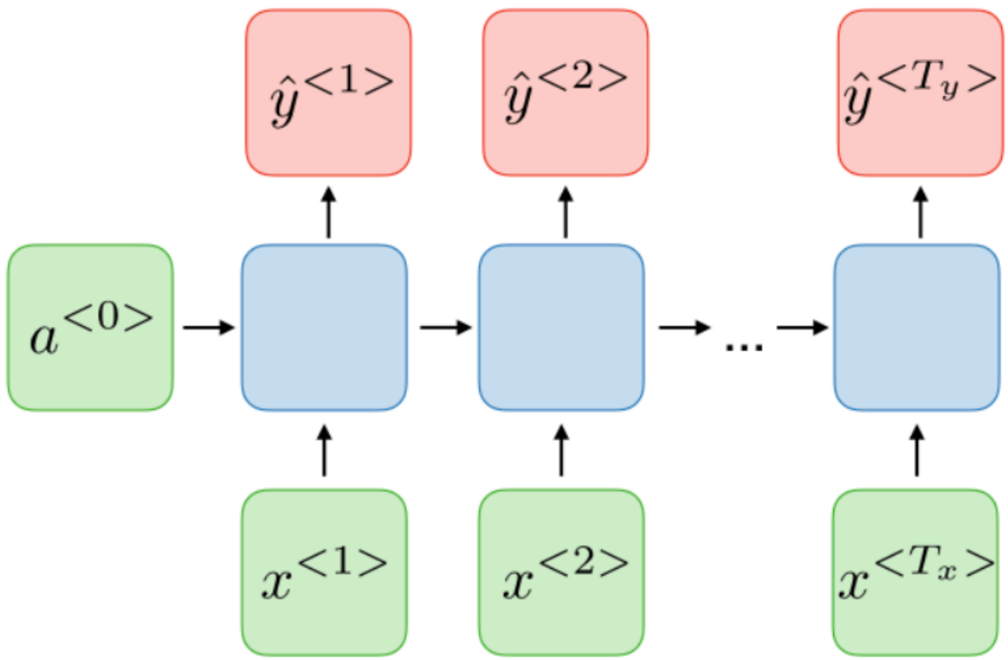
Example Application: Music Generation



One-to-Many, $T_x = 1$, $T_y > 1$

Image Credit: Afshine Amidi and Shervine Amidi Stanford CS 230

Example Application: Named Entity Recognition



Many-to-Many, $T_x = T_y$

Image Credit: Afshine Amidi and Shervine Amidi Stanford CS 230

Training a Recurrent Net

Loss Function:

$$\mathcal{L}(\hat{y}, y) = \sum_{t=1}^{T_y} \mathcal{L}(\hat{y}^{<t>}, y^{<t>})$$

Prediction Ground-truth Time Step

Backprop:

$$\frac{\partial \mathcal{L}^{(T)}}{\partial W} = \sum_{t=1}^{T} \left. \frac{\partial \mathcal{L}^{(T)}}{\partial W} \right|_{(t)}$$

Weight Matrix

Advantages of RNNs

- Ability to process time series data of any length
- Model size does not increase with input size
- Takes into account history when processing
- Weights are shared across time

Disadvantages of RNNs

- Computation is slow
- Difficulty accessing information from many timesteps in the past
- Current state cannot consider any input from the future

Vanishing / Exploding Gradient

For many years, RNNs were interesting theoretically, but not practical to train

Reason: it is difficult to capture long term dependencies because of a multiplicative gradient that can be exponentially increasing / decreasing with respect to the number of layers in the network

RNN Strategy 1: Gradient Clipping

Cap the maximum value of the gradient:

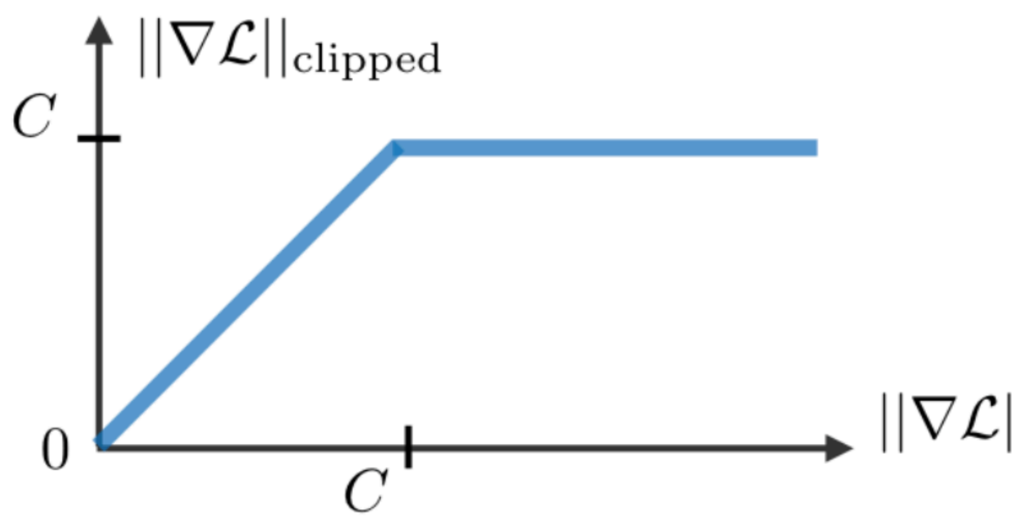


Image Credit: Afshine Amidi and Shervine Amidi Stanford CS 230

RNN Strategy 2: Types of Gates

Specific gates that have a well-defined purpose can address the vanishing gradient problem.

Sigmoid Function

General Form: $$\Gamma = \sigma(Wx^{<t>} + Ua^{<t-1>} + b)$$

Gate-Specific Coefficients

Type of gate	Role
Update gate Γ_u	How much past should matter now?
Relevance gate Γ_r	Drop previous information?
Forget gate Γ_f	Erase a cell or not?
Output gate Γ_o	How much to reveal of a cell?

Image Credit: Afshine Amidi and Shervine Amidi Stanford CS 230

Standard Arch. That Works: LSTM

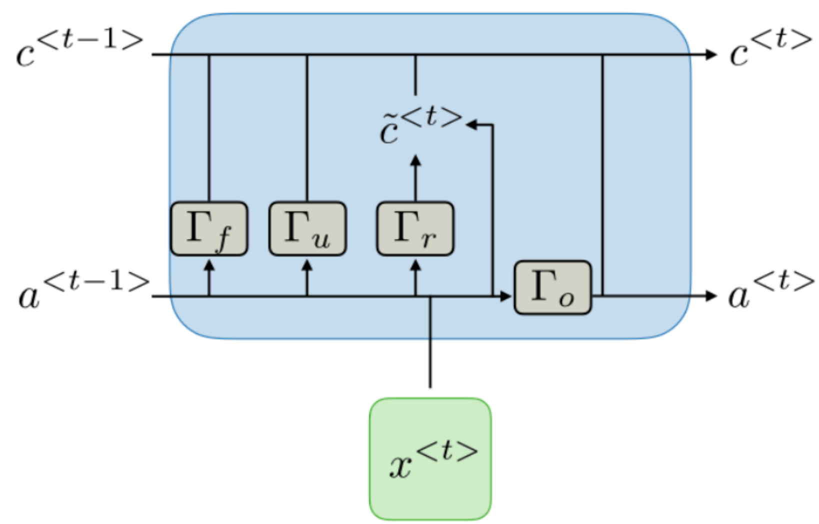
Characterization	**Long Short-Term Memory** (LSTM)
$\tilde{c}^{<t>}$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$
$c^{<t>}$	$\Gamma_u \star \tilde{c}^{<t>} + \Gamma_f \star c^{<t-1>}$
$a^{<t>}$	$\Gamma_o \star c^{<t>}$
Dependencies	

Image Credit: Afshine Amidi and Shervine Amidi Stanford CS 230

Recurrent Models of Brain Function

Two models of recurrence

Linear recurrent model:

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \mathbf{F}(\mathbf{h} + \mathbf{M} \cdot \mathbf{v})$$

Non-linear recurrent model:

$$\mathbf{F}(\mathbf{h} + \mathbf{M} \cdot \mathbf{r}) = [\mathbf{h} + \mathbf{M} \cdot \mathbf{r} - \gamma]_+$$

Adds rectification

Tuning Curves

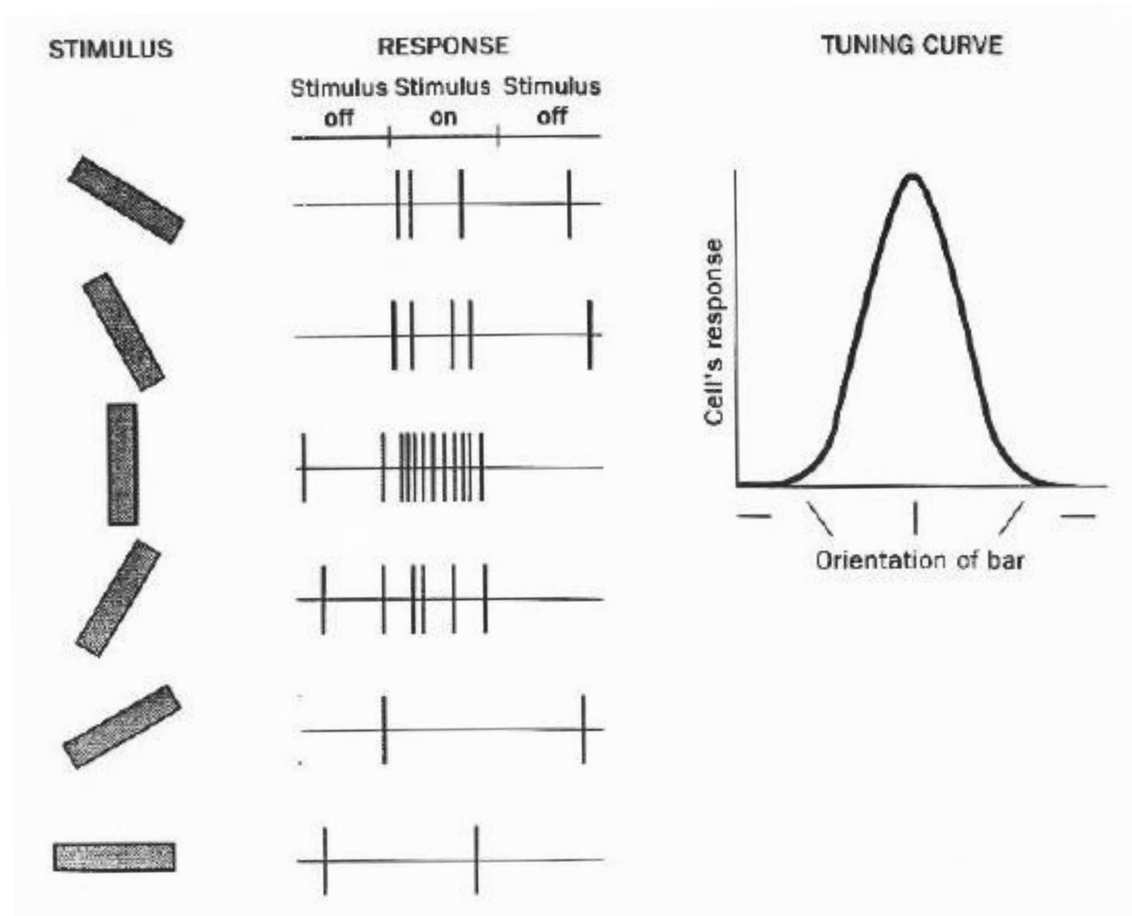


FIGURE 4.8 Response of a single cortical cell to bars presented at various orientations.

Hubel and Wiesel 1968

Linear Amplification

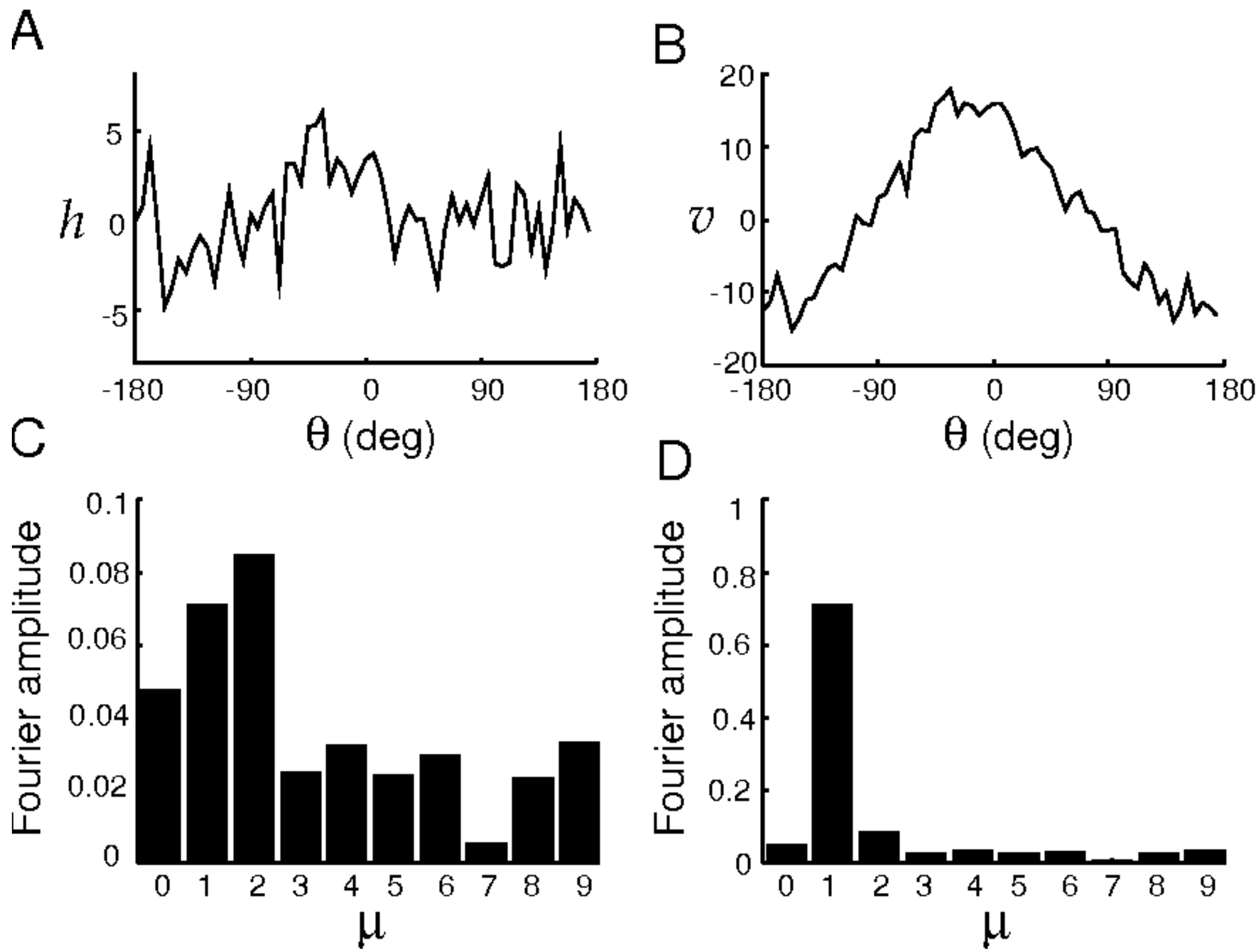
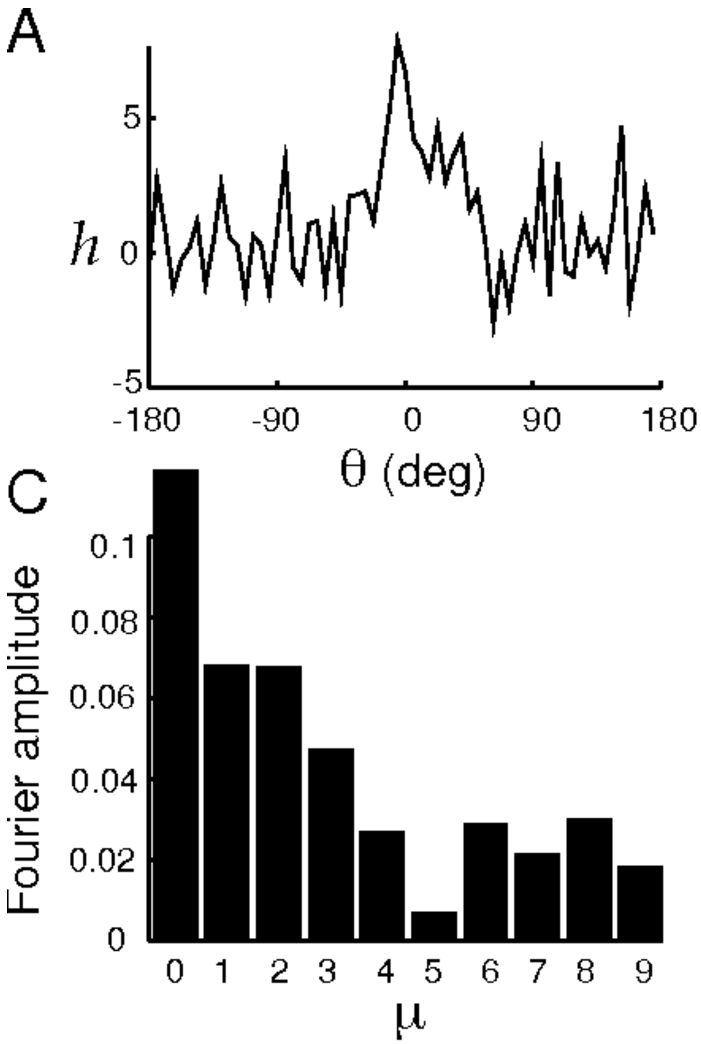


Image Credit: Dayan and Abbot 2001

Non-Linear Amplification



no negative output

Image Credit: Dayan and Abbot 2001

Recurrent Model of Simple Cells in Primary Visual Cortex

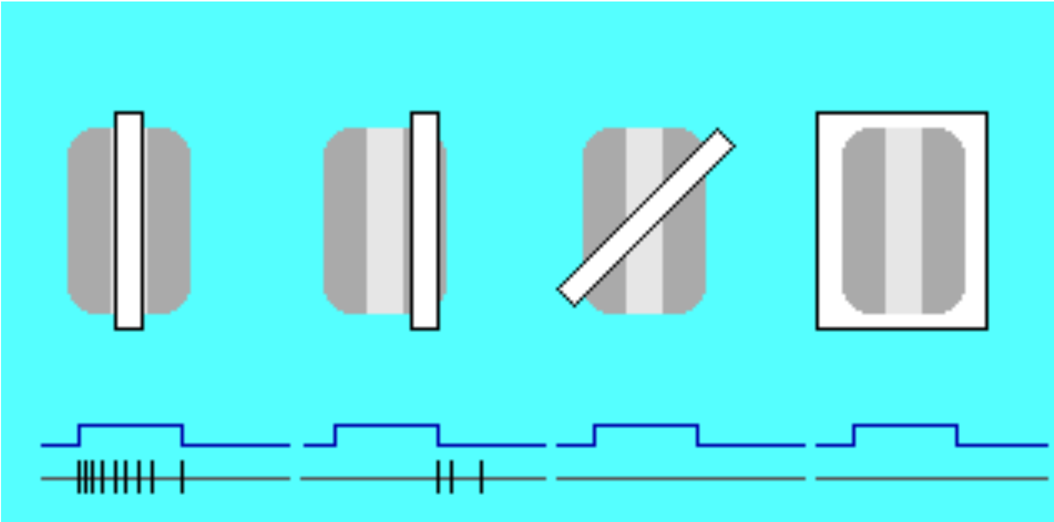


Image Credit: http://www.lifesci.sussex.ac.uk/home/George_Mather/Linked%20Pages/Physiol/Cortex.html

Recurrent Model of Simple Cells in Primary Visual Cortex

Ben-Yishai, Bar-Or and Sompolinsky 2005

firing rate of neuron

clamped orientation angles

$$\tau_r \frac{dv(\theta)}{dt} = -v(\theta) + \left[h(\theta) + \int_{-\pi/2}^{\pi/2} \frac{d\theta'}{\pi} (-\lambda_0 + \lambda_1 \cos(2(\theta - \theta'))) v(\theta') \right]_+$$

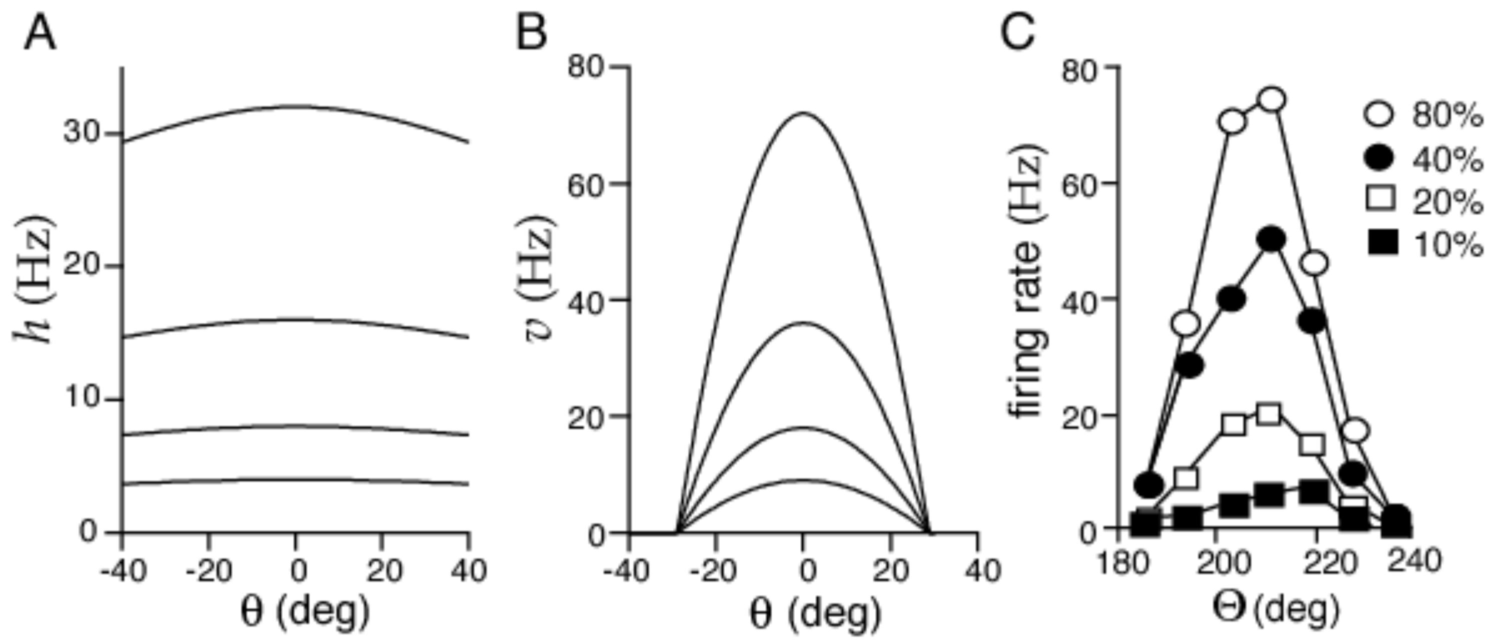
orientation

image contrast

$$h(\theta) = Ac(1 - \epsilon + \epsilon \cos(2\theta))$$

overall amplitude

Recurrent Model of Simple Cells in Primary Visual Cortex



Feed-Forward Model

Recurrent Model

Experimental Data

Image Credit: Dayan and Abbot 2001

A Recurrent Model of Complex Cells in Primary Visual Cortex

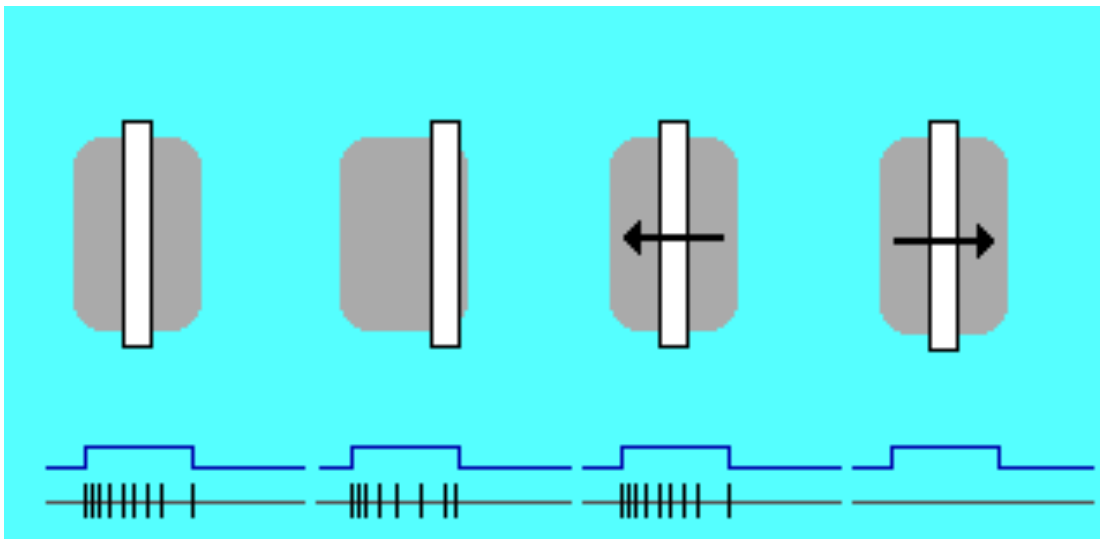


Image Credit: http://www.lifesci.sussex.ac.uk/home/George_Mather/Linked%20Pages/Physiol/Cortex.html

A Recurrent Model of Complex Cells in Primary Visual Cortex

Chance, Nelson and Abbott 1999

Weight Function:

angle

$$M(\phi - \phi') = \lambda_1/(2\pi\rho_\phi)$$

control recurrence

Firing Rates Determined By:

$$\tau_r \frac{dv(\phi)}{dt} = -v(\phi) + \left[h(\phi) + \frac{\lambda_1}{2\pi}\int_{-\pi}^{\pi} d\phi' v(\phi')\right]_+$$

feed-forward input

A Recurrent Model of Complex Cells in Primary Visual Cortex

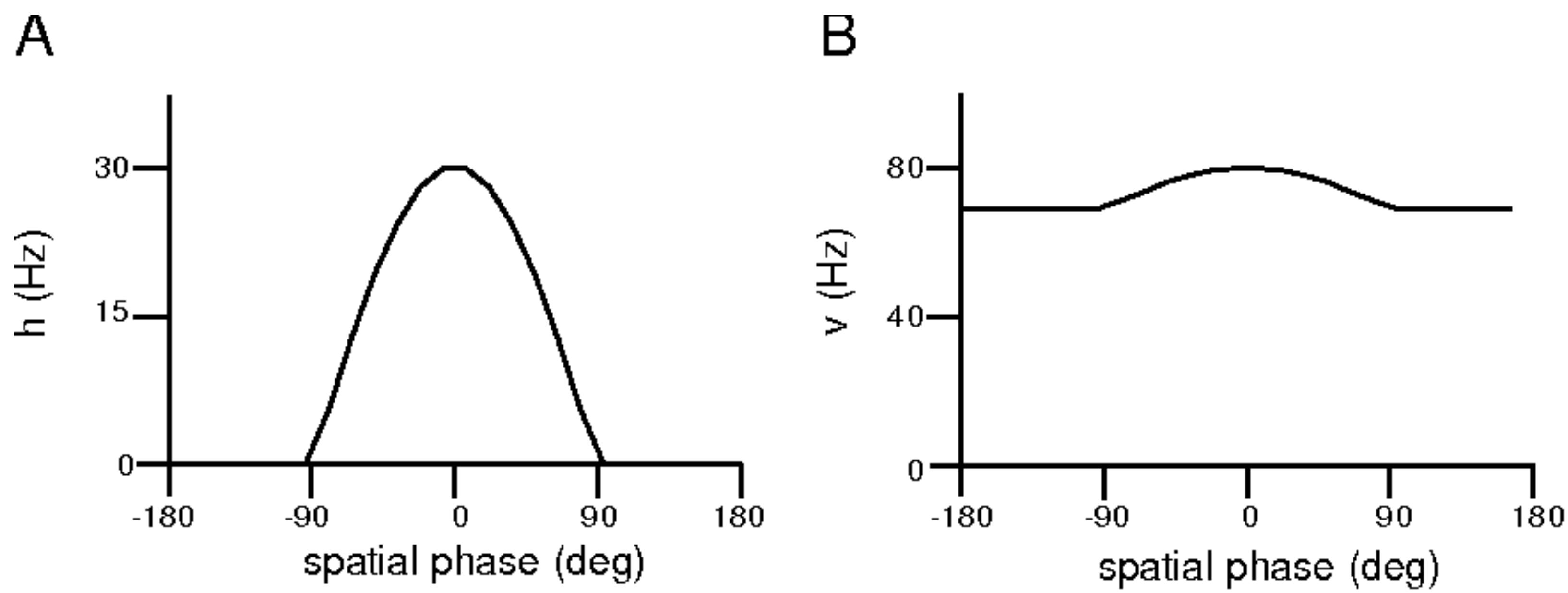


Image Credit: Dayan and Abbot 2001