RNNs
+
PAC Learning
Reminders

• Homework 5: Neural Networks
  – Out: Mon, Oct. 11
  – Due: Thu, Oct. 21 at 11:59pm

• More exam viewings today! (Wed, Oct. 13)
  – 12 – 1
  – 3 – 5
  – Split across BH 235B & BH 255A based on where you took your exam.
  – @1029 on Piazza
Q: In Lecture 12, when you showed us the binary Cross Entropy objective function, was there a minus sign missing?

A: Oops! Yes. Since we want to minimize cross entropy, there should have been a minus sign out front!

\[
\begin{align*}
\text{Forward} & \\
\text{Quadratic} & \quad J = \frac{1}{2}(y - y^*)^2 \\
\text{Cross Entropy} & \quad J = -(y^* \log(y) + (1 - y^*) \log(1 - y)) \\
\text{Backward} & \\
\frac{dJ}{dy} & = y - y^* \\
\frac{dJ}{dy} & = -(y^* \frac{1}{y} + (1 - y^*) \frac{1}{y - 1})
\end{align*}
\]
SGD for CNNs

Ex: Architecture: Given \( \hat{x}, y^{*} \)

\[
J = \ell(y, y^{*}) \\
y = \text{softmax}(z^{(5)}) \\
z^{(5)} = \text{linear}(z^{(4)}, \theta) \\
z^{(4)} = \text{relu}(z^{(3)}) \\
z^{(3)} = \text{conv}(z^{(2)}, \beta) \\
z^{(2)} = \text{max-pool}(z^{(1)}) \\
z^{(1)} = \text{conv}(\hat{x}, \alpha) \\
\]

Params: \( \theta = [\alpha, \beta, \theta] \)

SGD:

1. Init: \( \hat{\theta} \)
2. While not converged:
   
   Simple: \( i \in \{1, \ldots, N\} \)
   
   Forward: \( y = h_{\hat{\theta}}(z^{(i)}) \), \( J_i(\theta) = \ell(y, y^{*}) \)
   
   Backward: \( \nabla_{\theta} J_i(\theta) = \ldots \)
   
   Update: \( \hat{\theta} \leftarrow \hat{\theta} - \gamma \nabla_{\theta} J_i(\theta) \)
LAYERS OF A CNN
ReLU Layer

Input: $\hat{x} \in \mathbb{R}^k$  
Output: $\hat{y} \in \mathbb{R}^k$

Forward:

$\hat{y} = \sigma(\hat{x}) \quad \text{element-wise}$

$\sigma(a) = \max(0, a)$

$
\begin{align*}
\frac{dJ}{dx_i} &= \frac{dJ}{dy_i} \frac{dy_i}{dx_i} \\
\text{where } \frac{dy_i}{dx_i} &= \begin{cases} 1 & \text{if } x_i > 0 \\ 0 & \text{otherwise} \end{cases}
\end{align*}$
Softmax Layer

Input: $\hat{x} \in \mathbb{R}^k$  
Output: $\hat{y} \in \mathbb{R}^k$

Forward:

$$y_i = \frac{\exp(x_i)}{\sum_{k=1}^{K} \exp(x_k)}$$

Backward:

$$\frac{dS}{dx_j} = \sum_{i=1}^{K} \frac{dS}{dy_i} \frac{dy_i}{dx_j}$$

where

$$\frac{dy_i}{dx_j} = \begin{cases} y_i(1-y_i) & \text{if } i=j \\ -y_i y_j & \text{otherwise} \end{cases}$$
Fully-Connected Layer

- Suppose input is a 3D Tensor: $X = \left[ X_1, \ldots, X_{(C \times H \times W)} \right]$

- Stretch out into a long vector: $X = W^T X$

- Then standard linear layer:

$$Y = \alpha^T X + \alpha_0 \quad \text{where} \quad \alpha \in \mathbb{R}^{A \times B}, \quad |X| = A, \quad |Y| = B$$
Convolutional Layer

**Ex 1:** 1 input channel, 1 output channel

<table>
<thead>
<tr>
<th>Input</th>
<th>Conv</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$x_{12}$</td>
<td>$x_{13}$</td>
</tr>
<tr>
<td>$x_{21}$</td>
<td>$x_{22}$</td>
<td>$x_{23}$</td>
</tr>
<tr>
<td>$x_{31}$</td>
<td>$x_{32}$</td>
<td>$x_{33}$</td>
</tr>
</tbody>
</table>

$y_{11} = \alpha_{11} x_{11} + \alpha_{12} x_{12} + \alpha_{21} x_{21} + \alpha_{22} x_{22} + \alpha_{31} x_{31} + \alpha_{32} x_{32} + \alpha_{0}$

$y_{12} = \alpha_{11} x_{11} + \alpha_{12} x_{12} + \alpha_{21} x_{22} + \alpha_{22} x_{32} + \alpha_{0}$

$y_{21} = \alpha_{11} x_{11} + \alpha_{12} x_{12} + \alpha_{21} x_{21} + \alpha_{22} x_{31} + \alpha_{0}$

$y_{22} = \alpha_{11} x_{11} + \alpha_{12} x_{12} + \alpha_{21} x_{22} + \alpha_{22} x_{33} + \alpha_{0}$

**Ex 2:** 1 input channel, 2 output channels

<table>
<thead>
<tr>
<th>Input</th>
<th>Conv#1</th>
<th>Output#1</th>
<th>Conv#2</th>
<th>Output#2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$x_{12}$</td>
<td>$x_{13}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_{21}$</td>
<td>$x_{22}$</td>
<td>$x_{23}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_{31}$</td>
<td>$x_{32}$</td>
<td>$x_{33}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$y_{11}^{(1)} = \alpha_{11}^{(1)} x_{11} + \alpha_{12}^{(1)} x_{12} + \alpha_{21}^{(1)} x_{21} + \alpha_{22}^{(1)} x_{22} + \alpha_{0}$

$y_{12}^{(1)} = \alpha_{11}^{(1)} x_{11} + \alpha_{12}^{(1)} x_{12} + \alpha_{21}^{(1)} x_{22} + \alpha_{22}^{(1)} x_{32} + \alpha_{0}$

$y_{21}^{(1)} = \alpha_{11}^{(1)} x_{11} + \alpha_{12}^{(1)} x_{12} + \alpha_{21}^{(1)} x_{21} + \alpha_{22}^{(1)} x_{31} + \alpha_{0}$

$y_{22}^{(1)} = \alpha_{11}^{(1)} x_{11} + \alpha_{12}^{(1)} x_{12} + \alpha_{21}^{(1)} x_{22} + \alpha_{22}^{(1)} x_{33} + \alpha_{0}$

$y_{11}^{(2)} = \alpha_{11}^{(2)} x_{11} + \alpha_{12}^{(2)} x_{12} + \alpha_{21}^{(2)} x_{21} + \alpha_{22}^{(2)} x_{22} + \alpha_{0}$

$y_{12}^{(2)} = \alpha_{11}^{(2)} x_{11} + \alpha_{12}^{(2)} x_{12} + \alpha_{21}^{(2)} x_{22} + \alpha_{22}^{(2)} x_{32} + \alpha_{0}$

$y_{21}^{(2)} = \alpha_{11}^{(2)} x_{11} + \alpha_{12}^{(2)} x_{12} + \alpha_{21}^{(2)} x_{21} + \alpha_{22}^{(2)} x_{31} + \alpha_{0}$

$y_{22}^{(2)} = \alpha_{11}^{(2)} x_{11} + \alpha_{12}^{(2)} x_{12} + \alpha_{21}^{(2)} x_{22} + \alpha_{22}^{(2)} x_{33} + \alpha_{0}$
Convolutional Layer

Example: $C^I$ input channels, $C^O$ output channels

- The slice is the output from the convolution matrix.
- $H^O = \left(\frac{H^I + 2p - K}{s} + 1\right)$
- $W^O = \left(\frac{W^I + 2p - K}{s} + 1\right)$

where $p$ = # pixels of padding on input
$k$ = size of conv. matrix
$s$ = stride length

Forward:

$y_{ij}^{(k)} = x_0^{(k)} + \sum_{c=1}^{C^I} \sum_{r=1}^{K} \sum_{q=1}^{K} a_k^{(k)} X_{mn}^{(c)}$ where $m = s(i-1) + q$
$n = s(j-1) + r$

Backward:

$\frac{dJ}{dw_0^{(k)}} = \sum_i \sum_j \frac{dJ}{d y_{ij}^{(k)}} \frac{d y_{ij}^{(k)}}{d a_0^{(k)}}$

$\frac{dJ}{d X_q^{(c)}} = \sum_i \sum_j \frac{dJ}{d y_{ij}^{(k)}} \frac{d y_{ij}^{(k)}}{d a_q^{(c)}}$

$\frac{dJ}{d X_{mn}^{(c)}} = \sum_i \sum_j \frac{dJ}{d y_{ij}^{(k)}} \frac{d y_{ij}^{(k)}}{d a_q^{(c)}} \frac{d a_q^{(c)}}{d X_{mn}^{(c)}}$
Max-Pooling Layer

Example: 1 input channel, 1 output channel, stride of 1

<table>
<thead>
<tr>
<th>Input</th>
<th>Pool Size</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{11}$, $x_{12}$, $x_{13}$</td>
<td></td>
<td>$y_{11} = \max \left( x_{11}, x_{12}, x_{21}, x_{22} \right)$</td>
</tr>
<tr>
<td>$x_{21}$, $x_{22}$, $x_{23}$</td>
<td></td>
<td>$y_{12} = \max \left( x_{12}, x_{13}, x_{22}, x_{23} \right)$</td>
</tr>
<tr>
<td>$x_{31}$, $x_{32}$, $x_{33}$</td>
<td></td>
<td>$y_{21} = \max \left( x_{21}, x_{22}, x_{31}, x_{32} \right)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$y_{22} = \max \left( x_{22}, x_{23}, x_{32}, x_{33} \right)$</td>
</tr>
</tbody>
</table>
Max-Pooling Layer

Forward:
\[ Y_{ij}^{(k)} = \max_{q \in \{1, \ldots, k^2\}} X_{mi}^{(k)} \text{ where } m = s(i-1) + q, \quad n = s(j-1) + r \]

Backward:
\[ \frac{dJ}{dx_{mn}} = \sum_{i} \sum_{j} \frac{dJ}{dy_{ij}^{(k)}} \cdot \frac{dy_{ij}^{(k)}}{dx_{mn}} \]
Convolutional Neural Network (CNN)

- Typical layers include:
  - Convolutional layer
  - Max-pooling layer
  - Fully-connected (Linear) layer
  - ReLU layer (or some other nonlinear activation function)
  - Softmax
- These can be arranged into arbitrarily deep topologies

**Architecture #1: LeNet-5**

Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.
Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network’s input is 150,528-dimensional, and the number of neurons in the network’s remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

Reducing Overfitting

Our neural network architecture has 60 million parameters. Although the 1000 classes of ILSVRC make each training example impose 10 bits of constraint on the mapping from image to label, this turns out to be insufficient to learn so many parameters without considerable overfitting. Below, we describe the two primary ways in which we combat overfitting.

4.1 Data Augmentation

The easiest and most common method to reduce overfitting on image data is to artificially enlarge the dataset using label-preserving transformations (e.g., [25, 4, 5]). We employ two distinct forms of data augmentation, both of which allow transformed images to be produced from the original images with very little computation, so the transformed images do not need to be stored on disk. In our implementation, the transformed images are generated in Python code on the CPU while the GPU is training on the previous batch of images. So these data augmentation schemes are, in effect, computationally free.

The first form of data augmentation consists of generating image translations and horizontal reflections. We do this by extracting random $224 \times 224$ patches (and their horizontal reflections) from the $256 \times 256$ images and training our network on these extracted patches. This increases the size of our training set by a factor of 2048, though the resulting training examples are, of course, highly interdependent. Without this scheme, our network suffers from substantial overfitting, which would have forced us to use much smaller networks. At test time, the network makes a prediction by extracting five $224 \times 224$ patches (the four corner patches and the center patch) as well as their horizontal reflections (hence ten patches in all), and averaging the predictions made by the network’s softmax layer on the ten patches.

The second form of data augmentation consists of altering the intensities of the RGB channels in training images. Specifically, we perform PCA on the set of RGB pixel values throughout the ImageNet training set. To each training image, we add multiples of the found principal components.
CNNs for Image Recognition

Revolution of Depth

ILSVRC'15 ResNet
ILSVRC'14 GoogleNet
ILSVRC'14 VGG
ILSVRC'13
ILSVRC'12 AlexNet
ILSVRC'11
ILSVRC'10

ImageNet Classification top-5 error (%)


Slide from Kaiming He
3D Visualization of CNN

http://scs.ryerson.ca/~aharley/vis/conv/
Convolution of a Color Image

- Color images consist of 3 floats per pixel for RGB (red, green blue) color values
- Convolution must also be 3-dimensional

Figure from Fei-Fei Li & Andrej Karpathy & Justin Johnson (CS231N)
Animation of 3D Convolution

http://cs231n.github.io/convolutional-networks/

Figure from Fei-Fei Li & Andrej Karpathy & Justin Johnson (CS231N)
MNIST Digit Recognition with CNNs (in your browser)

https://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html
CNN Summary

CNNs

– Are used for all aspects of computer vision, and have won numerous pattern recognition competitions
– Able learn interpretable features at different levels of abstraction
– Typically, consist of convolution layers, pooling layers, nonlinearities, and fully connected layers

Other Resources:

– Readings on course website
– Andrej Karpathy, CS231n Notes
  http://cs231n.github.io/convolutional-networks/
BACKGROUND:
N-GRAM LANGUAGE MODELS
n-Gram Language Model

- **Goal**: Generate realistic looking sentences in a human language
- **Key Idea**: condition on the last n-1 words to sample the n\textsuperscript{th} word

\[
p(· | \text{START}) \quad p(· | \text{START, The}) \quad p(· | \text{The, bat}) \quad p(· | \text{bat, made}) \quad p(· | \text{made, noise}) \quad p(· | \text{noise, at})
\]
**n-Gram Language Model**

**Question**: How can we **define** a probability distribution over a sequence of length T?

The bat made night noise at w_1 w_2 w_3 w_4 w_5 w_6

**n-Gram Model (n=2)**

\[ p(w_1, w_2, \ldots, w_T) = \prod_{t=1}^{T} p(w_t | w_{t-1}) \]

\[ p(w_1, w_2, w_3, \ldots, w_6) = \]

- \( p(w_1) \)
- \( p(w_2 | w_1) \)
- \( p(w_3 | w_2) \)
- \( p(w_4 | w_3) \)
- \( p(w_5 | w_4) \)
- \( p(w_6 | w_5) \)
n-Gram Language Model

**Question:** How can we *define* a probability distribution over a sequence of length $T$?

\[ p(w_1, w_2, \ldots, w_T) = \prod_{t=1}^{T} p(w_t \mid w_{t-1}, w_{t-2}) \]

For $n$-Gram Model ($n=3$):

\[ p(w_1, w_2, w_3, \ldots, w_6) = \]

\[ p(w_1) \]
\[ p(w_2 \mid w_1) \]
\[ p(w_3 \mid w_2, w_1) \]
\[ p(w_4 \mid w_3, w_2) \]
\[ p(w_5 \mid w_4, w_3) \]
\[ p(w_6 \mid w_5, w_4) \]
n-Gram Language Model

**Question:** How can we define a probability distribution over a sequence of length T?

\[
p(w_1, w_2, \ldots, w_T) = \prod_{t=1}^{T} p(w_t \mid w_{t-1}, w_{t-2})
\]

**n-Gram Model (n=3)**

\[
p(w_1, w_2, w_3, \ldots, w_6) = p(w_1) \cdot p(w_2 \mid w_1) \cdot p(w_3 \mid w_2, w_1) \cdot p(w_4 \mid w_3, w_2) \cdot p(w_5 \mid w_4, w_3) \cdot p(w_6 \mid w_5, w_4)
\]

**Note:** This is called a **model** because we made some **assumptions** about how many previous words to condition on (i.e. only n-1 words)
Learning an n-Gram Model

**Question**: How do we **learn** the probabilities for the n-Gram Model?

$$p(w_t \mid w_{t-2} = \text{The, } \ w_{t-1} = \text{bat})$$

<table>
<thead>
<tr>
<th>$w_t$</th>
<th>$p(\cdot \mid \cdot, \cdot)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ate</td>
<td>0.015</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>flies</td>
<td>0.046</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>zebra</td>
<td>0.000</td>
</tr>
</tbody>
</table>

$$p(w_t \mid w_{t-2} = \text{made, } \ w_{t-1} = \text{noise})$$

<table>
<thead>
<tr>
<th>$w_t$</th>
<th>$p(\cdot \mid \cdot, \cdot)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>at</td>
<td>0.020</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>pollution</td>
<td>0.030</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>zebra</td>
<td>0.000</td>
</tr>
</tbody>
</table>

$$p(w_t \mid w_{t-2} = \text{cows, } \ w_{t-1} = \text{eat})$$

<table>
<thead>
<tr>
<th>$w_t$</th>
<th>$p(\cdot \mid \cdot, \cdot)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>corn</td>
<td>0.420</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>grass</td>
<td>0.510</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>zebra</td>
<td>0.000</td>
</tr>
</tbody>
</table>
Learning an n-Gram Model

**Question:** How do we **learn** the probabilities for the n-Gram Model?

**Answer:** From data! Just **count** n-gram frequencies

\[
p(w_t | w_{t-2} = \text{cows}, w_{t-1} = \text{eat})
\]

| \(w_t\)  | \(p(\cdot | \cdot, \cdot)\) |
|---------|---------------------------|
| corn    | 4/11                      |
| grass   | 3/11                      |
| hay     | 2/11                      |
| if      | 1/11                      |
| which   | 1/11                      |

... the **cows eat grass**...
... our **cows eat hay** daily...
... factory-farm **cows eat corn**...
... on an organic farm, **cows eat hay** and...
... do your **cows eat grass** or corn?...
... what do **cows eat if** they have...
... **cows eat corn** when there is no...
... which **cows eat which** foods depends...
... if **cows eat grass**...
... when **cows eat corn** their stomachs...
... should we let **cows eat corn**?...
Sampling from a Language Model

**Question**: How do we sample from a Language Model?

**Answer**:
1. Treat each probability distribution like a (50k-sided) weighted die
2. Pick the die corresponding to \( p(w_t \mid w_{t-2}, w_{t-1}) \)
3. Roll that die and generate whichever word \( w_t \) lands face up
4. Repeat

The bat made night noise at
Sampling from a Language Model

**Question**: How do we sample from a Language Model?

**Answer**:
1. Treat each probability distribution like a (50k-sided) weighted die
2. Pick the die corresponding to \( p(w_t \mid w_{t-2}, w_{t-1}) \)
3. Roll that die and generate whichever word \( w_t \) lands face up
4. Repeat

<table>
<thead>
<tr>
<th>Training Data (Shakespeare)</th>
<th>5-Gram Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>I tell you, friends, most charitable care ave the patricians of you. For your wants, Your suffering in this dearth, you may as well Strike at the heaven with your staves as lift them Against the Roman state, whose course will on The way it takes, cracking ten thousand curbs Of more strong link asunder than can ever Appear in your impediment. For the dearth, The gods, not the patricians, make it, and Your knees to them, not arms, must help.</td>
<td>Approacheth, denay. dungy Thither! Julius think: grant,—0 Yead linens, sheep's Ancient, Agreed: Petrarch plaguay Resolved pear! observingly honourest adulteries wherever scabbard guess; affirmation—his monsieur; died. jealousy, chequins me. Daphne building. weakness: sun-rise, cannot stays carry't, unpurposed. prophet-like drink; back-return 'gainst surmise Bridget ships? wane; interim? She's striving wet;</td>
</tr>
</tbody>
</table>
RECURRENT NEURAL NETWORK (RNN) LANGUAGE MODELS
Recurrent Neural Networks (RNNs)

inputs: \( x = (x_1, x_2, \ldots, x_T), x_i \in \mathcal{R}^I \)
hidden units: \( h = (h_1, h_2, \ldots, h_T), h_i \in \mathcal{R}^J \)
outputs: \( y = (y_1, y_2, \ldots, y_T), y_i \in \mathcal{R}^K \)
nonlinearity: \( \mathcal{H} \)

Definition of the RNN:
\[
\begin{align*}
    h_t &= \mathcal{H} (W_{xh} x_t + W_{hh} h_{t-1} + b_h) \\
    y_t &= W_{hy} h_t + b_y
\end{align*}
\]
The Chain Rule of Probability

**Question:** How can we define a probability distribution over a sequence of length $T$?

Chain rule of probability: \[ p(w_1, w_2, \ldots, w_T) = \prod_{t=1}^{T} p(w_t \mid w_{t-1}, \ldots, w_1) \]

Note: This is called the chain rule because it is always true for every probability distribution.
**RNN Language Model**

**RNN Language Model:** \[ p(w_1, w_2, \ldots, w_T) = \prod_{t=1}^{T} p(w_t \mid f_\theta(w_{t-1}, \ldots, w_1)) \]

\[ p(w_1, w_2, w_3, \ldots, w_6) = \]

- \( p(w_1) \)
- \( p(w_2 \mid f_\theta(w_1)) \)
- \( p(w_3 \mid f_\theta(w_2, w_1)) \)
- \( p(w_4 \mid f_\theta(w_3, w_2, w_1)) \)
- \( p(w_5 \mid f_\theta(w_4, w_3, w_2, w_1)) \)
- \( p(w_6 \mid f_\theta(w_5, w_4, w_3, w_2, w_1)) \)

**Key Idea:**

1. convert all previous words to a **fixed length vector**
2. define distribution \( p(w_t \mid f_\theta(w_{t-1}, \ldots, w_1)) \) that conditions on the vector
RNN Language Model

Key Idea:
(1) convert all previous words to a **fixed length vector**
(2) define distribution $p(w_t | f_\theta(w_{t-1}, \ldots, w_1))$ that conditions on the vector $h_t = f_\theta(w_{t-1}, \ldots, w_1)$
**Key Idea:**

(1) convert all previous words to a **fixed length vector**

(2) define distribution \( p(w_t | f_\theta(w_{t-1}, \ldots, w_1)) \) that conditions on the vector \( h_t = f_\theta(w_{t-1}, \ldots, w_1) \)
RNN Language Model

**Key Idea:**

1. convert all previous words to a **fixed length vector**
2. define distribution \( p(w_t | h_t) \) that conditions on the vector \( h_t = f_\theta(w_{t-1}, ..., w_1) \)
RNN Language Model

**Key Idea:**
(1) convert all previous words to a **fixed length vector**
(2) define distribution \( p(w_t | f_\theta(w_{t-1}, \ldots, w_1)) \) that conditions on the vector \( h_t = f_\theta(w_{t-1}, \ldots, w_1) \)
**Key Idea:**
(1) convert all previous words to a **fixed length vector**
(2) define distribution $p(w_t \mid f_\theta(w_{t-1}, \ldots, w_1))$ that conditions on
the vector $h_t = f_\theta(w_{t-1}, \ldots, w_1)$
Key Idea:
(1) convert all previous words to a **fixed length vector**
(2) define distribution $p(w_t \mid f_\theta(w_{t-1}, \ldots, w_1))$ that conditions on the vector $h_t = f_\theta(w_{t-1}, \ldots, w_1)$
Key Idea:
(1) convert all previous words to a **fixed length vector**
(2) define distribution $p(w_t | f_\theta(w_{t-1}, \ldots, w_1))$ that conditions on the vector $h_t = f_\theta(w_{t-1}, \ldots, w_1)$
RNN Language Model

Key Idea:
(1) convert all previous words to a fixed length vector
(2) define distribution \( p(w_t | f_\theta(w_{t-1}, \ldots, w_1)) \) that conditions on the vector \( h_t = f_\theta(w_{t-1}, \ldots, w_1) \)
RNN Language Model

\[
p(w_1, w_2, w_3, \ldots, w_T) = p(w_1 | h_1) p(w_2 | h_2) \ldots p(w_T | h_T)
\]
Question: How do we sample from a Language Model?

Answer:
1. Treat each probability distribution like a (50k-sided) weighted die
2. Pick the die corresponding to $p(w_t | w_{t-2}, w_{t-1})$
3. Roll that die and generate whichever word $w_t$ lands face up
4. Repeat

The same approach to sampling we used for an n-Gram Language Model also works here for an RNN Language Model.
Sampling from an RNN-LM

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

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

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him is but young and tender; and, I should be loath to foil him, as I honour, if he come in: By love to you, I came hither to acquaint you with, that either you might stay him from his intendment or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.

Example from http://karpathy.github.io/2015/05/21/rnn-effectiveness/
Sampling from an RNN-LM

Shakespeare’s As You Like It

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

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

RNN-LM Sample

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him well. Your brother is but young and tender; and, for your love, I would be loath to foil him, as I must, for my own honour, if he come in: therefore, out of my love to you, I came hither to acquaint you withal, that either you might stay him from his intendment or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.
Sampling from an RNN-LM

RNN-LM Sample

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

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

Shakespeare’s As You Like It

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him well. Your brother is but young and tender; and, for your love, I would be loath to foil him, as I must, for my own honour, if he come in: therefore, out of my love to you, I came hither to acquaint you withal, that either you might stay him from his intendment or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.

Example from http://karpathy.github.io/2015/05/21/rnn-effectiveness/
Sampling from an RNN-LM

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

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

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him is but young and tender; and, I should be loath to foil him, as I honour, if he come in: My love to you, I came hither to acquaint you with, that either you might stay him from his intent or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.

Example from http://karpathy.github.io/2015/05/21/rnn-effectiveness/
SEQUENCE TO SEQUENCE MODELS
Sequence to Sequence Model

Speech Recognition

Machine Translation
기계 번역은 특히 영어와 한국어와 같은 언어 쌍의 경우 매우 어렵습니다.

Summarization
Now suppose you want to generate a sequence conditioned on another input.

**Key Idea:**

1. Use an **encoder** model to generate a vector representation of the **input**.
2. Feed the output of the encoder to a **decoder** which will generate the **output**.

**Applications:**
- Translation: Spanish → English
- Summarization: article → summary
- Speech recognition: speech signal → transcription

---

**Diagram:**

Encoder:
- $e_1, e_2, e_3, e_4$
- Vamos, al, cafe, ahora

Decoder:
- $d_1, d_2, d_3$
- START, Let’s, go

$p(w_3|h_3)$
LEARNING THEORY
PAC(-MAN) Learning

For some hypothesis $h \in \mathcal{H}$:

1. True Error
   
   $R(h)$

2. Training Error
   
   $\hat{R}(h)$

Question 2:
What is the expected number of PAC-MAN levels Matt will complete before a Game-Over?

A. 1-10
B. 11-20
C. 21-30
Questions for today (and next lecture)

1. Given a classifier with zero training error, what can we say about true error (aka. generalization error)?
   (Sample Complexity, Realizable Case)

2. Given a classifier with low training error, what can we say about true error (aka. generalization error)?
   (Sample Complexity, Agnostic Case)

3. Is there a theoretical justification for regularization to avoid overfitting?
   (Structural Risk Minimization)
PAC/SLT Model for Supervised ML

\[ x(i) \sim p^*(\cdot) \]

\[ D_{\text{train}} \]

\[ y(1) \]

\[ y(2) \]

\[ y(3) \]

Learning Algorithm

\[ h(x) \]
PAC/SLT Model for Supervised ML

• **Problem Setting**
  – Set of possible inputs, \( x \in \mathcal{X} \) (all possible patients)
  – Set of possible outputs, \( y \in \mathcal{Y} \) (all possible diagnoses)
  – Distribution over instances, \( p^*(\cdot) \)
  – Exists an unknown target function, \( c^* : \mathcal{X} \rightarrow \mathcal{Y} \)
    (the doctor’s brain)
  – Set, \( \mathcal{H} \), of candidate hypothesis functions, \( h : \mathcal{X} \rightarrow \mathcal{Y} \)
    (all possible decision trees)

• **Learner is given** \( N \) training examples
  \( D = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(N)}, y^{(N)})\} \)
  where \( x^{(i)} \sim p^*(\cdot) \) and \( y^{(i)} = c^*(x^{(i)}) \)
  (history of patients and their diagnoses)

• **Learner produces** a hypothesis function, \( \hat{y} = h(x) \), that
  best approximates unknown target function \( y = c^*(x) \) on
  the training data
PAC/SLT Model for Supervised ML

• Problem Setting
  – Set of possible inputs, \( x \in \mathcal{X} \) (all possible patients)
  – Set of possible outputs, \( y \in \mathcal{Y} \) (all possible diagnoses)
  – Distribution over instances, \( p^*(\cdot) \)
  – Exists an unknown target function, \( \text{c}^* : \mathcal{X} \rightarrow \mathcal{Y} \) (the doctor’s brain)
  – Set, \( \mathcal{H} \), of candidate hypothesis functions, \( h : \mathcal{X} \rightarrow \mathcal{Y} \) (all possible decision trees)

• Learner is given \( N \) training examples:
  \[ D = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(N)}, y^{(N)})\} \]
  where \( x^{(i)} \sim p^*(\cdot) \) and \( y^{(i)} = c^*(x^{(i)}) \)
  (history of patients and their diagnoses)

• Learner produces a hypothesis function, \( \hat{y} \), that best approximates unknown target function \( y = c^*(x) \) on the training data

Two important settings we’ll consider:

1. **Classification**: the possible outputs are **discrete**
2. **Regression**: the possible outputs are **real-valued**
PAC/SLT Model for Supervised ML

\[ x^{(i)} \sim p^*(\cdot) \]

\[ x^{(1)} \]

\[ x^{(2)} \]

\[ x^{(3)} \]

\[ x^{(4)} \]

\[ x^{(5)} \]

\[ D_{\text{train}} \]

\[ y^{(1)} \]

\[ y^{(2)} \]

\[ y^{(3)} \]

\[ y^{(4)} \]

\[ y^{(5)} \]

\[ c^*(x) \]

\[ h(x) \]

Learning Algorithm

\[ \hat{y}^{(4)} \]

\[ \hat{y}^{(5)} \]

Predictions

Test Error Rate

\[ \sim p^*(\cdot) \]
Two Types of Error

1. True Error (aka. expected risk)

\[ R(h) = P_{x \sim p^*}(c^*(x) \neq h(x)) \]

2. Train Error (aka. empirical risk)

\[ \hat{R}(h) = P_{x \sim S}(c^*(x) \neq h(x)) \]

\[ = \frac{1}{N} \sum_{i=1}^{N} 1(c^*(x^{(i)}) \neq h(x^{(i)})) \]

\[ = \frac{1}{N} \sum_{i=1}^{N} 1(y^{(i)} \neq h(x^{(i)})) \]

where \( S = \{x^{(1)}, \ldots, x^{(N)}\}_{i=1}^{N} \) is the training data set, and \( x \sim S \) denotes that \( x \) is sampled from the empirical distribution.
PAC / SLT Model

1. Generate instances from unknown distribution $p^*$
   \[ x^{(i)} \sim p^*(x), \forall i \]  
\[ (1) \]

2. Oracle labels each instance with unknown function $c^*$
   \[ y^{(i)} = c^*(x^{(i)}), \forall i \]  
\[ (2) \]

3. Learning algorithm chooses hypothesis $h \in \mathcal{H}$ with low(est) training error, $\hat{R}(h)$
   \[ \hat{h} = \arg\min_h \hat{R}(h) \]  
\[ (3) \]

4. Goal: Choose an $h$ with low generalization error $R(h)$
Three Hypotheses of Interest

The true function $c^*$ is the one we are trying to learn and that labeled the training data:

$$y^{(i)} = c^*(x^{(i)}), \forall i$$  \hspace{1cm} (1)

The expected risk minimizer has lowest true error:

$$h^* = \underset{h \in H}{\mathrm{argmin}} R(h)$$

The empirical risk minimizer has lowest training error:

$$\hat{h} = \underset{h \in H}{\mathrm{argmin}} \hat{R}(h)$$  \hspace{1cm} (3)

Question: True or False: $h^*$ and $c^*$ are always equal.
Question 1

A

B

C
PAC LEARNING
Probably Approximately Correct (PAC) Learning

Whiteboard:

– PAC Criterion
– Meaning of “Probably Approximately Correct”
– Def: PAC Learner
– Sample Complexity
– Consistent Learner
– Realizable vs. Agnostic Cases
– Finite vs. Infinite Hypothesis Spaces
SAMPLE COMPLEXITY RESULTS
**Sample Complexity Results**

**Definition 0.1.** The *sample complexity* of a learning algorithm is the number of examples required to achieve arbitrarily small error (with respect to the optimal hypothesis) with high probability (i.e. close to 1).

**Four Cases we care about…**

We’ll start with the finite case…

| Finite $|\mathcal{H}|$ | Realizable |
|-------------------------|------------|
|                         |            |

| Infinite $|\mathcal{H}|$ | Agnostic   |
|-----------------|-------------|
|                 |             |
Probably Approximately Correct (PAC) Learning

Whiteboard:
  – Theorem 1: Realizable Case, Finite $|H|$ 
  – Proof of Theorem 1
Sample Complexity Results

**Definition 0.1.** The **sample complexity** of a learning algorithm is the number of examples required to achieve arbitrarily small error (with respect to the optimal hypothesis) with high probability (i.e. close to 1).

**Four Cases we care about...**

<table>
<thead>
<tr>
<th></th>
<th>Realizable</th>
<th>Agnostic</th>
</tr>
</thead>
<tbody>
<tr>
<td>**Finite</td>
<td>(</td>
<td>\mathcal{H}</td>
</tr>
<tr>
<td>**Infinite</td>
<td>(</td>
<td>\mathcal{H}</td>
</tr>
</tbody>
</table>
Example: Conjunctions

Question:
Suppose $H = \text{class of conjunctions over } x \text{ in } \{0,1\}^M$

Example hypotheses:
- $h(x) = x_1 (1-x_3) x_5$
- $h(x) = x_1 (1-x_2) x_4 (1-x_5)$

If $M = 10$, $\varepsilon = 0.1$, $\delta = 0.01$, how many examples suffice according to Theorem 1?

Answer:
A. $10*(2*\ln(10)+\ln(100)) \approx 92$
B. $10*(3*\ln(10)+\ln(100)) \approx 116$
C. $10*(10*\ln(2)+\ln(100)) \approx 116$
D. $10*(10*\ln(3)+\ln(100)) \approx 156$
E. $100*(2*\ln(10)+\ln(10)) \approx 691$
F. $100*(3*\ln(10)+\ln(10)) \approx 922$
G. $100*(10*\ln(2)+\ln(10)) \approx 924$
H. $100*(10*\ln(3)+\ln(10)) \approx 1329$

**Thm. 1**

$N \geq \frac{1}{\varepsilon} \left[ \log(|\mathcal{H}|) + \log\left(\frac{1}{\delta}\right) \right]$ labeled examples are sufficient so that with probability $(1 - \delta)$ all $h \in \mathcal{H}$ with $\hat{R}(h) = 0$ have $R(h) \leq \varepsilon$. 
Question 2

A
B
C
D
E
F
G
H

@ When poll is active, respond at pollev.com/10301601polls
Sample Complexity Results

**Definition 0.1.** The sample complexity of a learning algorithm is the number of examples required to achieve arbitrarily small error (with respect to the optimal hypothesis) with high probability (i.e. close to 1).

**Four Cases we care about...**

| Finite $|\mathcal{H}|$ | Realizable | Agnostic |
|-----------------|------------|----------|
| $\text{Thm. 1}$ | $N \geq \frac{1}{\epsilon} \left[ \log(|\mathcal{H}|) + \log(\frac{1}{\delta}) \right]$ labeled examples are sufficient so that with probability $(1 - \delta)$ all $h \in \mathcal{H}$ with $\hat{R}(h) = 0$ have $R(h) \leq \epsilon.$ |
| $\text{Thm. 2}$ | $N \geq \frac{1}{2\epsilon^2} \left[ \log(|\mathcal{H}|) + \log(\frac{2}{\delta}) \right]$ labeled examples are sufficient so that with probability $(1 - \delta)$ for all $h \in \mathcal{H}$ we have that $|R(h) - \hat{R}(h)| \leq \epsilon.$ |

| Infinite $|\mathcal{H}|$ |  |  |
1. Bound is **inversely linear in epsilon** (e.g. halving the error requires double the examples)
2. Bound is **only logarithmic in |H|** (e.g. quadrupling the hypothesis space only requires double the examples)

---

**Realizable**

**Finite |H|**

**Thm. 1** \( N \geq \frac{1}{\epsilon} \left[ \log(|H|) + \log\left(\frac{1}{\delta}\right) \right] \) labeled examples are sufficient so that with probability \((1 - \delta)\) all \( h \in H \) with \( \hat{R}(h) = 0 \) have \( R(h) \leq \epsilon. \)

**Infinite |H|**

---

**Agnostic**

**Thm. 2** \( N \geq \frac{1}{2\epsilon^2} \left[ \log(|H|) + \log\left(\frac{2}{\delta}\right) \right] \) labeled examples are sufficient so that with probability \((1 - \delta)\) for all \( h \in H \) we have that \( |R(h) - \hat{R}(h)| \leq \epsilon. \)
Sample Complexity Results

**Definition 0.1.** The *sample complexity* of a learning algorithm is the number of examples required to achieve arbitrarily small error (with respect to the optimal hypothesis) with high probability (i.e. close to 1).

**Four Cases we care about...**

**Realizable**

| Finite $|\mathcal{H}|$ | Infinite $|\mathcal{H}|$ |
|-----------------------|-------------------------|

**Thm. 1** \( N \geq \frac{1}{\epsilon} \left[ \log(|\mathcal{H}|) + \log\left(\frac{1}{\delta}\right) \right] \) labeled examples are sufficient so that with probability \((1 - \delta)\) all \( h \in \mathcal{H} \) with \( R(h) \leq \epsilon \).

**Agnostic**

| Finite $|\mathcal{H}|$ | Infinite $|\mathcal{H}|$ |
|-----------------------|-------------------------|

**Thm. 2** \( N \geq \frac{1}{2\epsilon^2} \left[ \log(|\mathcal{H}|) + \log\left(\frac{2}{\delta}\right) \right] \) labeled examples are sufficient so that for all \( h \in \mathcal{H} \) we have \( R(h) \leq \epsilon \).

We need a new definition of "complexity" for a Hypothesis space for these results (see VC Dimension).
**Sample Complexity Results**

**Definition 0.1.** The **sample complexity** of a learning algorithm is the number of examples required to achieve arbitrarily small error (with respect to the optimal hypothesis) with high probability (i.e. close to 1).

**Four Cases we care about...**

| Finite $|\mathcal{H}|$ | Realizable | Agnostic |
|-----------------------|------------|----------|
| **Thm. 1** $N \geq \frac{1}{\epsilon} \left[ \log(|\mathcal{H}|) + \log\left(\frac{1}{\delta}\right) \right]$ labeled examples are sufficient so that with probability $(1 - \delta)$ all $h \in \mathcal{H}$ with $\hat{R}(h) = 0$ have $\hat{R}(h) \leq \epsilon$. |

| Infinite $|\mathcal{H}|$ | Realizable | Agnostic |
|-----------------------|------------|----------|
| **Thm. 3** $N = O\left(\frac{1}{\epsilon} \left[ \text{VC}(\mathcal{H}) \log\left(\frac{1}{\epsilon}\right) + \log\left(\frac{1}{\delta}\right) \right] \right)$ labeled examples are sufficient so that with probability $(1 - \delta)$ all $h \in \mathcal{H}$ with $\hat{R}(h) = 0$ have $\hat{R}(h) \leq \epsilon$. |
| **Thm. 2** $N \geq \frac{1}{2\epsilon^2} \left[ \log(|\mathcal{H}|) + \log\left(\frac{2}{\delta}\right) \right]$ labeled examples are sufficient so that with probability $(1 - \delta)$ for all $h \in \mathcal{H}$ we have that $|R(h) - \hat{R}(h)| \leq \epsilon$. |
| **Thm. 4** $N = O\left(\frac{1}{\epsilon^2} \left[ \text{VC}(\mathcal{H}) + \log\left(\frac{1}{\delta}\right) \right] \right)$ labeled examples are sufficient so that with probability $(1 - \delta)$ for all $h \in \mathcal{H}$ we have that $|R(h) - \hat{R}(h)| \leq \epsilon$. |