Practice Problems

ECE/CS 498 DS U/G

Lecture 28: Practice Problems

Ravi K. Iyer

Dept. of Electrical and Computer Engineering

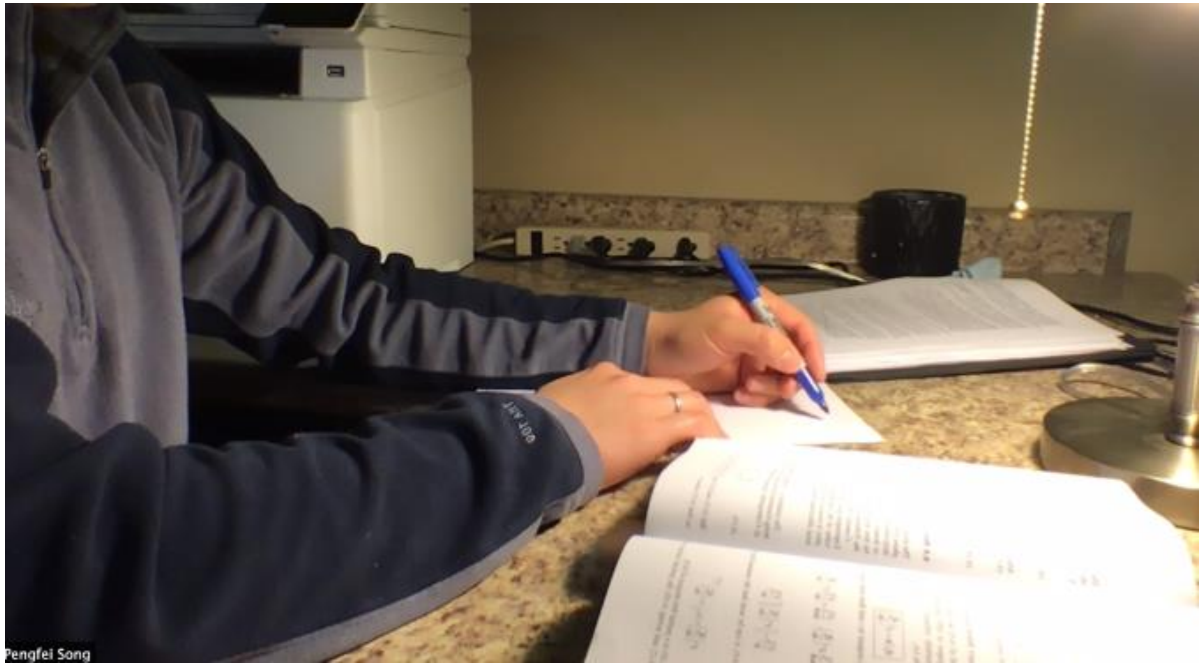
University of Illinois at Urbana Champaign

Announcements

- HW 5 **due tonight by 11:59 PM on Compass**
- Final Project:
  - **Final Presentations will be this Saturday 5/9 from 2-5 PM via Zoom**
    - All graduate students **must attend** the first three presentations
    - Afterwards, there will be two streams
    - Undergraduate students are strongly encouraged to attend
    - We will release presentation signups tonight
  - **Final Report is due Tuesday 5/12 at 11:59 PM via Compass**
    - Up to 8-page IEEE conference style report
    - Must be done in LaTex
    - We will release report template tonight
- TA Office Hours:
  - This week's TA office hours will occur as normal (MW 2-3 PM)
  - Next week, there will be a special extended TA office hours on **Thursday 5/14 from 4-6 PM**

Final Exam Update

- Due to issues with Proctorio integration, we will now be using Zoom for the final exam
- We will share the Zoom link soon
- You should sign into Zoom at least 5 minutes (that is, 7:55 AM) before the start of the exam (8:00 AM)
- **Make sure your webcam is on**. You need to show your hands and your workspace throughout the exam. For example,



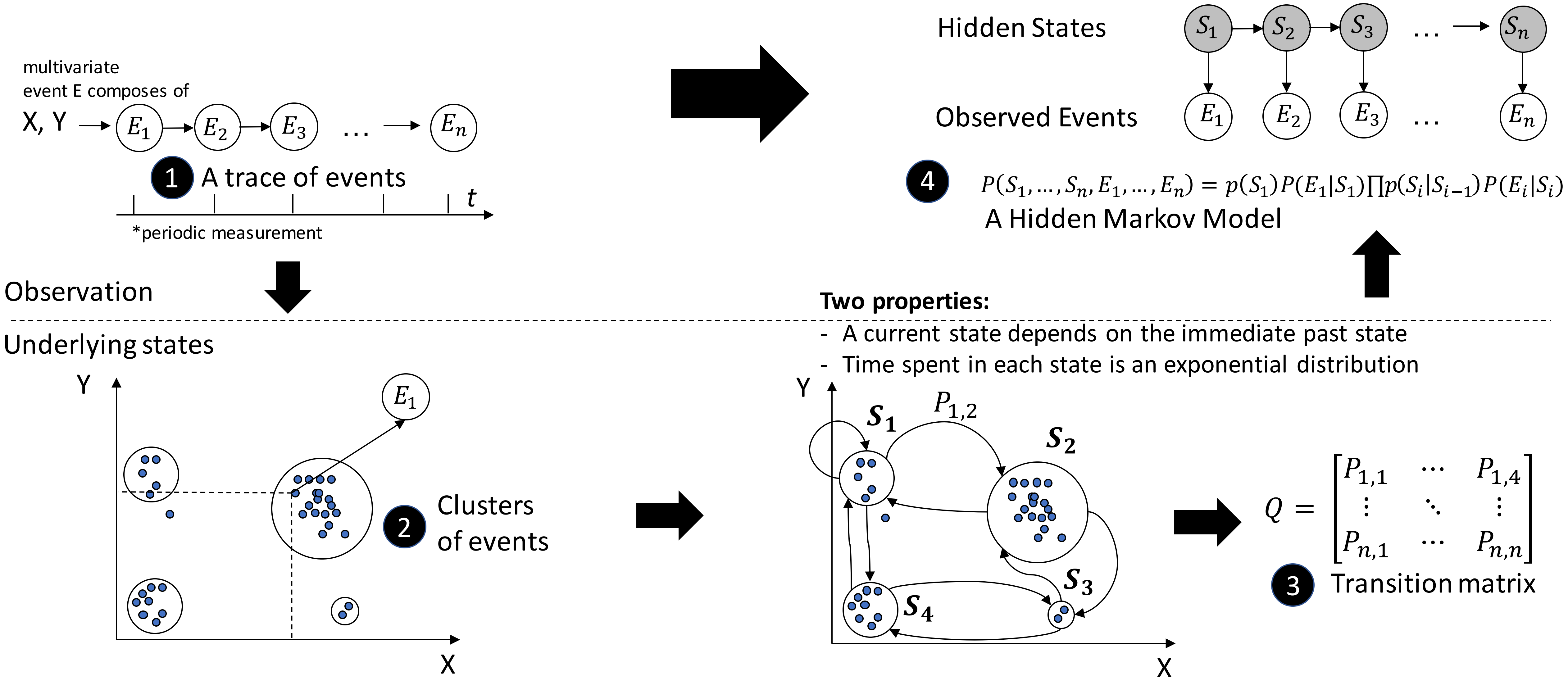
Final Exam Update

- **Make sure your microphone is on**
- Exam PDF will be released on Compass after you sign an honor agreement
- Exam schedule on **Friday 5/15**
  - 8:00 - 8:10 AM : Read through exam (no writing)
  - 8:10 – 10:40 AM : Complete exam and write out solutions. You will have 2.5 hours
  - 10:40 – 11:10 AM : Scan and submit your solution to Compass (no writing)
    - Late submissions will not be accepted
- If you have questions, raise your hand in Zoom and the instructors will privately message you
  - You will not be allowed to message other students

Final Exam Update

- **You will be allowed three 8.5x11 cheat sheets on your table, but nothing else**
  - It will be a **closed-book exam** – i.e. no external references (books, Google, electronics, other people, etc.) are allowed
  - During the exam, you should not use your computer except for (i) joining Zoom, (ii) looking through exam PDF, and (iii) scanning/submitting your solutions
  - You are only allowed to use your phone/tablet to scan your exam after you have completed it
- Here is a non-comprehensive list of topics for the exam:
  - **All course material is fair game**
  - **Material from after midterm exam will be emphasized (~75%)**
    - HMMs and Forward-Backward Algorithm
    - Factor Graphs and Belief Propagation
    - SVM
    - Neural Networks
    - Random Forest
    - Cross Validation, etc.
  - **Earlier course material will also be included (~25%)**
    - Basic Probability
    - Naïve Bayes and Bayesian Networks
    - Clustering (K-means, GMM, Hierarchical)
    - Linear/Logistic Regression
    - PCA, etc.

From a trace of events to a Hidden Markov Model



Hidden Markov Models

Model assumptions

An observation depends on its hidden state

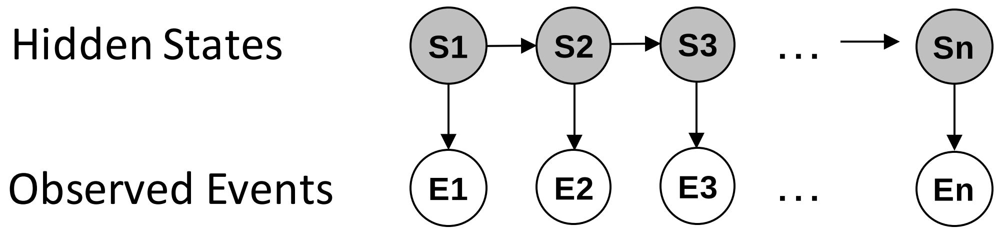
A state variable only depends on the immediate previous state (Markov assumption)

The future observations and the past observations are conditionally independent given the current hidden state

Advantages:

HMM can model sequential nature of input data (future depends on the past)

HMM has a linear-chain structure that clearly separates system state and observed events.



$$P(S_1, \ldots, S_n, E_1, \ldots, E_n) = p(S_1)P(E_1|S_1)\prod p(S_i|S_{i-1})P(E_i|S_i)$$

A Hidden Markov model on observed events and system states

Markov Model

- Consider a system which can occupy one of *N* discrete *states* or *categories*

$$x_t \in \{1, 2, \ldots, N\} \longrightarrow \text{state at time } t$$

- We are interested in *stochastic* systems, in which state evolution is random
- Any *joint* distribution can be factored into a series of *conditional* distributions:

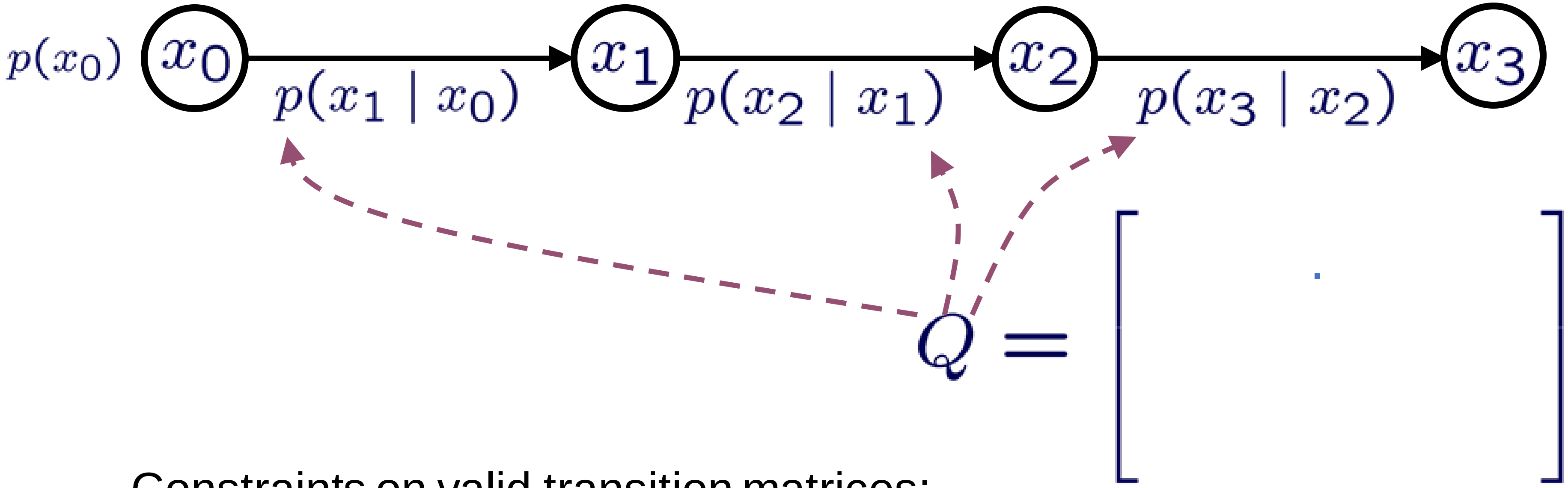
$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_0, \ldots, x_{t-1})$$

- For a *Markov* process, the next state depends only on the current state:

$$p(x_{t+1} \mid x_0, \ldots, x_t) = p(x_{t+1} \mid x_t)$$

Markov Chains: Graphical Models

$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1})$$



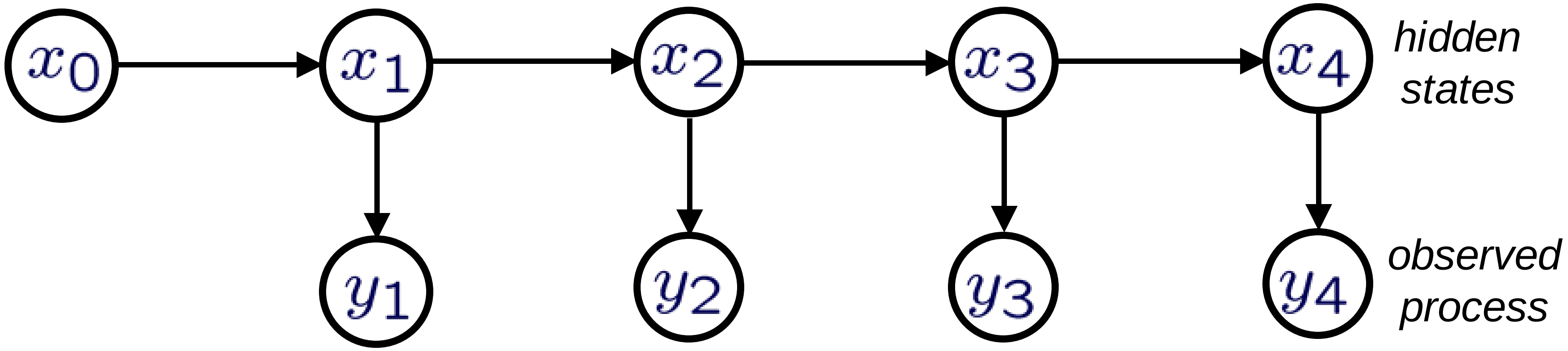
Constraints on valid transition matrices:

$$q_{ij} \geq 0 \,, \quad \sum_{i=1}^{N} q_{ij} = 1 \quad \text{for all } j$$

$$q_{ij} \triangleq p(x_{t+1} = i \mid x_t = j)$$

Hidden Markov Models (Packet Stall Example Cont'd)

- Stall exists due to congestion
- Not directly measurable at runtime (hidden)
- Motivates *hidden Markov models (HMM):*



$$p(x_0, x_1, \dots, x_T, y_1, y_2, \dots, y_T) = p(x_0)\prod_{t=1}^{T} p(x_t|x_{t-1})p(y_t|x_t)$$

Given x_t, previous observations do not impact future observations

$$p(y_t, y_{t+1}, \dots, y_T | x_t, y_{t-1}, y_{t-2}, \dots, y_1) = p(y_t, y_{t+1}, \dots, y_T | x_t)$$

Hidden Markov Models

Model

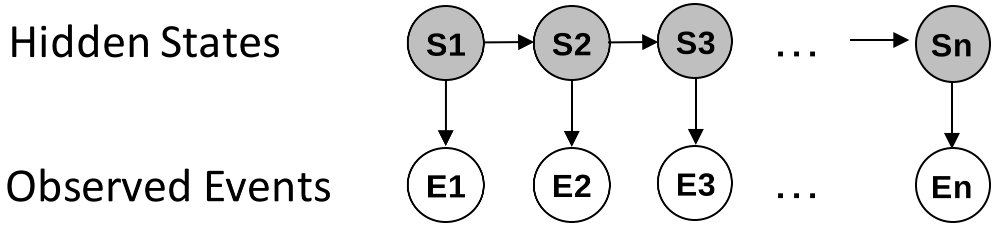
- Set of hidden states $\boldsymbol{S} = \{\boldsymbol{\sigma_1}, \dots, \boldsymbol{\sigma_N}\}$
- Set of observable events $\boldsymbol{E} = \{\boldsymbol{\epsilon_1}, \dots, \boldsymbol{\epsilon_M}\}$
- Transition probability matrix $\boldsymbol{A}$
- Observation matrix $\boldsymbol{B}$
- Initial distribution of hidden states $\boldsymbol{\pi}$

Model assumptions

- An observation depends on its hidden state
- A state variable only depends on the immediate previous state (Markov assumption)
- The future observations and the past observations are conditionally independent given the current hidden state

Advantages:

- HMM can model sequential nature of input data (future depends on the past)
- HMM has a linear-chain structure that clearly separates system state and observed events.



A Hidden Markov model on observed events and system states

$$P(S_1, \dots, S_n, E_1, \dots, E_n) = P(S_1)P(E_1|S_1)\prod_{i=2}^{n} P(S_i|S_{i-1})P(E_i|S_i)$$

General Inference question

Given the sequence of n observations $E_1, E_2, \dots, E_n$, and the model (A, B, π), how do we choose a corresponding state sequence $S_1, S_2, \dots, S_n$ which is optimal in some meaningful sense (i.e., best explains the observations)?

A simpler question: Given the sequence of n observations $E_1, E_2, \dots, E_n$, and the model (A, B, π), what is the most probable state S_t at $t \in \{1, \dots, n\}$?

$$\underset{j \in \{1,\dots,N\}}{\operatorname{argmax}} P(S_t = \sigma_j | E_1, E_2, \dots, E_n)$$

$$S = \{\sigma_1, \dots, \sigma_N\}$$

Breaking down the inference question

$$P(S_t|E_1, E_2, \dots, E_n) = \frac{P(E_{t+1}, \dots, E_n\ |S_t)\, P(S_t|E_1, \dots, E_t)}{P(E_{t+1}, \dots, E_n|E_1, \dots, E_t)}$$

$P(S_t|E_1, \dots, E_t)$:
Probability of hidden state at time t given observation up to time t (**Forwards algorithm**)

$P(E_{t+1}, \dots, E_n\ |S_t)$:
Probability of the future observed sequence given the hidden state at time t (**Backwards algorithm**)

$P(E_{t+1}, \dots, E_n|E_1, \dots, E_t)$:
Does not depend on the hidden state (will not affect the maximization because it is just a scaling factor)

Forwards algorithm: General Expression

Define: $\alpha_t(j) = P(S_t = \sigma_j | E_1, E_2, \ldots, E_t)$ and $Z_t = P(E_t | E_1, \ldots E_{t-1})$

In general,

$$\alpha_t(j) = \frac{1}{Z_t} P(E_t | S_t = \sigma_j) \sum_{i=1}^{N} P(S_t = \sigma_j \mid S_{t-1} = \sigma_i)\, \alpha_{t-1}(i) \qquad Z_t = \sum_{j=1}^{N} b_t \odot (A^T \alpha_{t-1})$$

Transition probability a_{ij}

Above equation can be written as a matrix for all j,

$$\begin{bmatrix} \alpha_t(1) \\ \vdots \\ \alpha_t(j) \\ \vdots \\ \alpha_t(N) \end{bmatrix} \propto \begin{bmatrix} P(E_t|S_t = \sigma_1) \\ \vdots \\ P(E_t|S_t = \sigma_j) \\ \vdots \\ P(E_t|S_t = \sigma_N) \end{bmatrix} \odot \begin{bmatrix} a_{11} & \cdots & \cdots & \cdots & a_{N1} \\ \vdots & \ddots & \cdots & \cdots & \vdots \\ a_{1j} & \cdots & a_{ij} & \cdots & a_{Nj} \\ \vdots & \cdots & \cdots & \ddots & \cdots \\ a_{1N} & \cdots & \cdots & \cdots & a_{NN} \end{bmatrix} \begin{bmatrix} \alpha_{t-1}(1) \\ \vdots \\ \alpha_{t-1}(i) \\ \vdots \\ \alpha_{t-1}(N) \end{bmatrix}$$

$\odot$ Represents elementwise product (Hadamard product)

$$\alpha_t \propto b_t \odot (A^T \alpha_{t-1})$$

b_t is the column of the observation matrix B corresponding to E_t

Forwards Algorithm: Paleontological Temperature

For observations T, L, D, T, L

$P(S_2|E_1 = T, E_2 = L)$ is,

$$\begin{bmatrix}\alpha_2(H)\\ \alpha_2(C)\end{bmatrix} \propto \begin{bmatrix}0.5\\ 0.1\end{bmatrix} \odot \left(\begin{bmatrix}0.7 & 0.4\\ 0.3 & 0.6\end{bmatrix}\begin{bmatrix}\alpha_1(H)\\ \alpha_1(C)\end{bmatrix}\right)$$

$$\begin{array}{ccc} & \text{H} & \text{C} \\ \begin{array}{c}\text{H}\\ \text{C}\end{array} & \multicolumn{2}{c}{\begin{bmatrix}0.7 & 0.3\\ 0.4 & 0.6\end{bmatrix}}\end{array}$$

Transition probability matrix

$$\begin{array}{cccc} & \text{T} & \text{D} & \text{L} \\ \begin{array}{c}\text{H}\\ \text{C}\end{array} & \multicolumn{3}{c}{\begin{bmatrix}0.1 & \boxed{0.4} & \boxed{0.5}\\ 0.7 & \boxed{0.2} & \boxed{0.1}\end{bmatrix}}\end{array}$$

Observation matrix

Similarly, $P(S_3|E_1 = T, E_2 = L,\ E_3 = D)$ is,

$$\begin{bmatrix}\alpha_3(H)\\ \alpha_3(C)\end{bmatrix} \propto \begin{bmatrix}0.4\\ 0.2\end{bmatrix} \odot \left(\begin{bmatrix}0.7 & 0.4\\ 0.3 & 0.6\end{bmatrix}\begin{bmatrix}\alpha_2(H)\\ \alpha_2(C)\end{bmatrix}\right)$$

Forwards Algorithm

1. Input: (A, B, π) and observed sequence $E_1, \dots, E_n$
2. $[\alpha_1, Z_1]$ = normalize($b_1 \odot \pi$)
3. **for** $t = 2{:}n$ **do**
   $[\alpha_t, Z_t]$ = normalize($b_t \odot (A^T \alpha_{t-1})$)
4. return $\alpha_1, \dots, \alpha_n$ and $\log\big(P(E_1, \dots, E_n)\big) = \sum_t \log(Z_t)$

Note:

Subroutine: [v, Z] = normalize(u): $Z = \sum_j u_j$; $v_j = u_j / Z$;

Breaking down the inference question

$$P(S_t|E_1, E_2, \ldots, E_n) = \frac{P(E_{t+1}, \ldots, E_n\ |S_t)\ P(S_t|E_1, \ldots, E_t)}{P(E_{t+1}, \ldots, E_n|E_1, \ldots, E_t)}$$

$P(S_t|E_1, \ldots, E_t)$:
Probability of hidden state at time t given observation up to time t (**Forwards algorithm**)

$P(E_{t+1}, \ldots, E_n\ |S_t)$:
Probability of the future observed sequence given the hidden state at time t (**Backwards algorithm**)

$P(E_{t+1}, \ldots, E_n|E_1, \ldots, E_t)$:
Does not depend on the hidden state (will not affect the maximization because it is just a scaling factor)

Backwards Algorithm

1. Input: (A, B, π) and observed sequence $E_1, \dots, E_n$
2. $\beta_n = 1$; // initialize $\beta_n(j)$ to 1 for all states σ_j
3. **for** $t = n - 1 : 1$ **do**
   $\beta_{t-1} = A(b_t \quad \beta_t)$
4. return $\beta_1, \dots, \beta_n$

Breaking down the inference question

$$P(S_t|E_1, E_2, \dots, E_n) = \frac{P(E_{t+1}, \dots, E_n\ |S_t)\ P(S_t|E_1, \dots, E_t)}{P(E_{t+1}, \dots, E_n|E_1, \dots, E_t)}$$

$P(S_t|E_1, \dots, E_t)$:
Probability of hidden state at time t given observation up to time t (**Forwards algorithm**)

$P(E_{t+1}, \dots, E_n\ |S_t)$:
Probability of the future observed sequence given the hidden state at time t (**Backwards algorithm**)

$P(E_{t+1}, \dots, E_n|E_1, \dots, E_t)$:
Does not depend on the hidden state (will not affect the maximization because it is just a scaling factor)

Inference: Most likely state

- Forwards-backwards algorithm gives $P(S_t = \sigma_j | E_1, \ldots, E_n)$ for all j
- Find the individually most likely state at time t given all observations

$$S_t^* = \operatorname*{argmax}_{j \in \{1,..,N\}} \gamma_t(j)$$

Optimality of inference

- In the inference problem we attempting to uncover the hidden part of HMM, i.e., find the “correct” state sequence
- It is impossible to find the “correct” state sequence (solution)
- Use optimality criterion to find the “best” possible solution
- Several reasonable criteria exist and is a strong function of the intended application
    - Most likely state given observations
        - Application in finding average statistics, expected number of correct states
        - Solved using Forwards-Backwards algorithm
    - Single best sequence that maximises probability of observed events
        - Application in continuous speech recognition
        - Solved using Viterbi algorithm

Hidden Markov Model – Online Battle Simulator Game

You are playing an online battle simulator named *WT*. To balance the gameplay, the game has a "balancer" that decides the difficulty of the game before the start of each round based on the previous round's difficulty. There are two difficulty levels *hard* (H) and *easy* (E), which are hidden from you. For the first round you play, the probability of the round being H is 0.25 and the probability of the round being E is 0.75. The transition probabilities between the states is given as A.

$$A = \begin{array}{c c} & \begin{array}{cc} H & E \end{array} \\ \begin{array}{c} H \\ E \end{array} & \begin{bmatrix} 0.8 & 0.2 \\ 0.5 & 0.5 \end{bmatrix} \end{array} \qquad \pi = \begin{array}{c} \begin{array}{cc} H & E \end{array} \\ \begin{bmatrix} 0.25 & 0.75 \end{bmatrix} \end{array}$$

Each game round can end in either a win (W), draw (D) or loss(L). The observation matrix is given below.

$$B = \begin{array}{c c} & \begin{array}{ccc} W & D & L \end{array} \\ \begin{array}{c} H \\ E \end{array} & \begin{bmatrix} 1/6 & 2/6 & 3/6 \\ 4/6 & 1/6 & 1/6 \end{bmatrix} \end{array}$$

Suppose you played for eight consecutive rounds and have the following result: **W, D, D, L, W, L, D, L**. You would like to infer the difficulty for each round you have played by using an HMM.

HMM Solution (1)

Forward algorithm

1. Input: (A, B, π) and observed sequence $E_1, \ldots, E_n$
2. $[\alpha_1, Z_1]$ = normalize($b_1 \odot \pi$)
3. **for** $t = 2{:}\,n$ **do**
   $[\alpha_t, Z_t]$ = normalize($b_t \odot (A^T \alpha_{t-1})$)
4. return $\alpha_1, \ldots, \alpha_n$ and $\log\big(P(E_1, \ldots, E_n)\big) = \sum_t \log(Z_t)$

5. Subroutine: [v, Z] = normalize(u): $Z = \sum_j u_j$; $v_j = u_j / Z$;

NOTE: $\odot$ represents elementwise product (Hadamard product)

Backward algorithm

1. Input: (A, B, π) and observed sequence $E_1, \ldots, E_n$
2. $\beta_n = 1$; // initialize $\beta_n(j)$ to 1 for all states σ_j
3. **for** $t = n - 1{:}\,1$ **do**
   $\beta_{t-1} = A(b_t \odot \beta_t)$
4. return $\beta_1, \ldots, \beta_n$

HMM Solution (2)

What is the most likely state at time step t=2 given the evidence?

$$S_2^* = argmax_{j\in\{H,E\}}\, \gamma_2(j) = argmax_{j\in\{H,E\}}\, \alpha_2(j) * \beta_2(j)$$

$$\alpha_1 \propto b_1 \odot \pi = \begin{bmatrix}1/6\\4/6\end{bmatrix} \odot \begin{bmatrix}1/4\\3/4\end{bmatrix} = \begin{bmatrix}1/24\\1/2\end{bmatrix}$$

Normalize(α_1) = $\begin{bmatrix}1/13\\12/13\end{bmatrix}$

$$\alpha_2 \propto b_2 \odot (A^T \alpha_1) = \begin{bmatrix}2/6\\1/6\end{bmatrix} \odot \left(\begin{bmatrix}0.8 & 0.5\\0.2 & 0.5\end{bmatrix}\begin{bmatrix}1/13\\12/13\end{bmatrix}\right) = \begin{bmatrix}34/195\\31/390\end{bmatrix}$$

Normalize(α_2) = $\begin{bmatrix}68/99\\31/99\end{bmatrix}$

Computing β_2, γ_2 is left as an exercise.

$$\begin{array}{ccc} & H & E \\ \pi = & 0.25 & 0.75 \end{array}$$

$$\pi = \begin{bmatrix}0.25 & 0.75\end{bmatrix}$$

$$A = \begin{array}{c|cc} & H & E \\ H & 0.8 & 0.2 \\ E & 0.5 & 0.5 \end{array}$$

$$B = \begin{array}{c|ccc} & W & D & L \\ H & 1/6 & 2/6 & 3/6 \\ E & 4/6 & 1/6 & 1/6 \end{array}$$

Observations

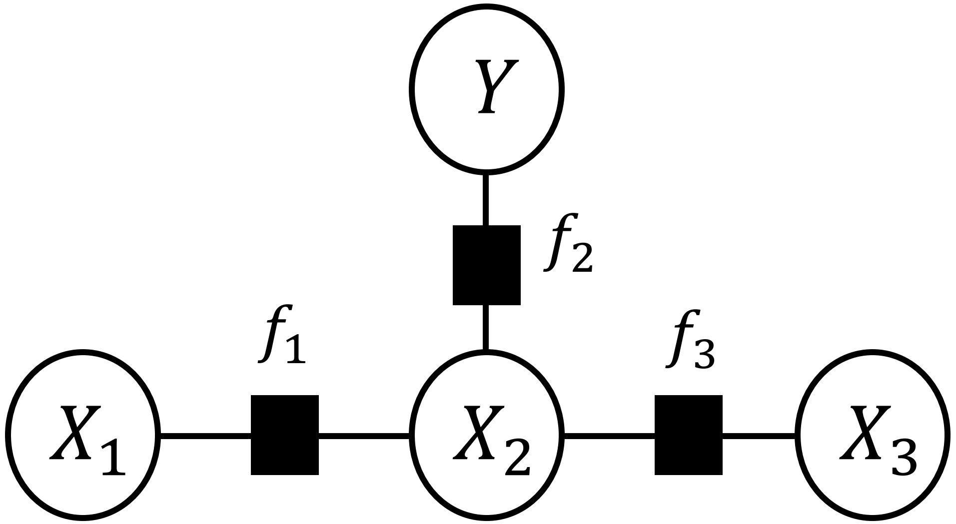
W	**D**	**D**	**L**	**W**	**L**	**D**	**L**

Factor Graphs Belief Propagation

Suppose X_1 is observed to be 0.

Using belief propagation with the factor graph to the right, calculate

- $P(X_2|X_1 = 0)$
- $P(Y|X_1 = 0)$
- $P(X_3|X_1 = 0)$



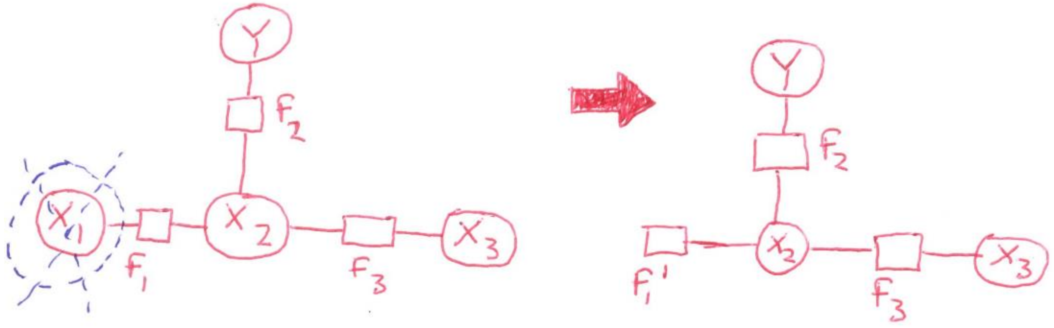
X_1	X_2	$f_1(X_1, X_2)$
0	0	0.9
0	1	1
1	0	1
1	1	0.9

X_2	Y	$f_2(X_2, Y)$
0	0	0.1
0	1	1
1	0	1
1	1	0.1

X_2	X_3	$f_3(X_2, X_3)$
0	0	1
0	1	0.1
1	0	0.1
1	1	1

Since X_1 has been observed to be 0, we can make the following adjustments:

- Graph structure:



- Factor table for f_1

X_1	X_2	$f_1(X_1, X_2)$
0	0	0.9
0	1	1
~~1~~	~~0~~	~~1~~
~~1~~	~~1~~	~~0.9~~

$\Rightarrow$

X_2	$f_1'(X_2)$
0	0.9
1	1

- Factor tables for f_2 and f_3 remain the same.

Let's find $P(X_2 | X_1 = 0)$

$$\mu_{f_1' \to X_2}(X_2) = f_1'(X_2) = \begin{array}{c}\begin{bmatrix} 0.9 \\ 1 \end{bmatrix}\end{array}\begin{array}{l}(X_2 = 0) \\ (X_2 = 1)\end{array}$$

$$\mu_{f_2 \to X_2}(X_2) = \sum_{Y} f_2(X_2, Y)\, \mu_{Y \to f_2}(Y)$$

$$= \begin{bmatrix} 0.1 \\ 1 \end{bmatrix}\begin{array}{l}(X_2=0, Y=0) \\ (X_2=1, Y=0)\end{array} \times 1$$

$$+ \begin{bmatrix} 1 \\ 0.1 \end{bmatrix}\begin{array}{l}(X_2=0, Y=1) \\ (X_2=1, Y=1)\end{array} \times 1 \quad = \begin{bmatrix} 1.1 \\ 1.1 \end{bmatrix}\begin{array}{l}(X_2=0) \\ (X_2=1)\end{array}$$

$$\mu_{f_3 \to X_2}(X_2) = \sum_{X_3} f_3(X_2, X_3)\, \mu_{X_3 \to f_3}(X_3)$$

$$= \begin{bmatrix} 1 \\ 0.1 \end{bmatrix}\begin{array}{l}(X_2=0, X_3=0) \\ (X_2=1, X_3=0)\end{array} \times 1 + \begin{bmatrix} 0.1 \\ 1 \end{bmatrix}\begin{array}{l}(X_2=0, X_3=1) \\ (X_2=1, X_3=1)\end{array} \times 1$$

$$= \begin{bmatrix} 1.1 \\ 1.1 \end{bmatrix}\begin{array}{l}(X_2=0) \\ (X_2=1)\end{array}$$

$$P(X_2 | X_1 = 0) = \frac{1}{Z}\left(\mu_{f_1' \to X_2} \times \mu_{f_2 \to X_2} \times \mu_{f_3 \to X_2}\right)$$

$$= \frac{1}{Z}\left(\begin{bmatrix} 0.9 \\ 1 \end{bmatrix} \odot \begin{bmatrix} 1.1 \\ 1.1 \end{bmatrix} \odot \begin{bmatrix} 1.1 \\ 1.1 \end{bmatrix}\right)$$

$$= \frac{1}{Z} \begin{bmatrix} 1.089 \\ 1.21 \end{bmatrix}$$

$$= \frac{1}{1.089 + 1.21} \begin{bmatrix} 1.089 \\ 1.21 \end{bmatrix}$$

$$\boxed{= \begin{bmatrix} 0.47 \\ 0.53 \end{bmatrix} \begin{matrix} (P(X_2=0 \mid X_1=0)) \\ (P(X_2=1 \mid X_1=0)) \end{matrix}}$$

Next, let's Find $P(Y \mid X_1=0)$

$$\mu_{f_1 \to X_2}(X_2) = \begin{bmatrix} 0.9 \\ 1 \end{bmatrix} \begin{matrix} (X_2=0) \\ (X_2=1) \end{matrix}$$

$$\mu_{f_3 \to X_2}(X_2) = \begin{bmatrix} 1.1 \\ 1.1 \end{bmatrix} \begin{matrix} (X_2=0) \\ (X_2=1) \end{matrix}$$ ← This was calculated earlier

$$\mu_{X_2 \to f_2}(X_2) = \mu_{f_1 \to X_2}(X_2) \times \mu_{f_3 \to X_2}(X_2)$$

$$= \begin{bmatrix} 0.9 \\ 1 \end{bmatrix} \odot \begin{bmatrix} 1.1 \\ 1.1 \end{bmatrix} = \begin{bmatrix} 0.99 \\ 1.1 \end{bmatrix} \begin{matrix} (X_2=0) \\ (X_2=1) \end{matrix}$$

$$\mu_{f_2 \to Y}(Y) = \sum_{X_2} f_2(X_2, Y)\, \mu_{X_2 \to f_2}(X_2)$$

$$= \begin{bmatrix} 0.1 \\ 1 \end{bmatrix} \begin{matrix} (X_2=0, Y=0) \\ (X_2=0, Y=1) \end{matrix} \times 0.99$$

$$+ \begin{bmatrix} 1 \\ 0.1 \end{bmatrix} \begin{matrix} (X_2=1, Y=0) \\ (X_2=1, Y=1) \end{matrix} \times 1.1$$

$$= \begin{bmatrix} 1.199 \\ 1.1 \end{bmatrix} \begin{matrix} (Y=0) \\ (Y=1) \end{matrix}$$

$$P(Y \mid X_1=0) = \frac{1}{Z} \mu_{f_2 \to Y}(Y)$$

$$= \frac{1}{Z} \begin{bmatrix} 1.199 \\ 1.1 \end{bmatrix}$$

$$= \frac{1}{1.199 + 1.1} \begin{bmatrix} 1.199 \\ 1.1 \end{bmatrix}$$

$$\boxed{= \begin{bmatrix} 0.52 \\ 0.48 \end{bmatrix} \begin{matrix} (P(Y=0 \mid X_1=0)) \\ (P(Y=1 \mid X_1=0)) \end{matrix}}$$

Finally, let's calculate $P(x_3 \mid x_1 = 0)$

$$\mu_{f_1' \to x_2}(x_2) = \begin{bmatrix} 0.9 \\ 1 \end{bmatrix} \begin{array}{l} (x_2=0) \\ (x_2=1) \end{array}$$

$$\mu_{f_2 \to x_2}(x_2) = \begin{bmatrix} 1.1 \\ 1.1 \end{bmatrix} \begin{array}{l} (x_2=0) \\ (x_2=1) \end{array}$$ ← This was already calculated previously

$$\mu_{x_2 \to f_3}(x_2) = \mu_{f_1' \to x_2} \times \mu_{f_2 \to x_2}$$

$$= \begin{bmatrix} 0.9 \\ 1 \end{bmatrix} \odot \begin{bmatrix} 1.1 \\ 1.1 \end{bmatrix} = \begin{bmatrix} 0.99 \\ 1.1 \end{bmatrix} \begin{array}{l} (x_2=0) \\ (x_2=1) \end{array}$$

$$\mu_{f_3 \to x_3}(x_3) = \sum_{x_2} f_3(x_2, x_3) \times \mu_{x_2 \to f_3}(x_2)$$

$$= \begin{bmatrix} 1 \\ 0.1 \end{bmatrix} \begin{array}{l} (x_2=0, x_3=0) \\ (x_2=0, x_3=1) \end{array} \times 0.99$$

$$+ \begin{bmatrix} 0.1 \\ 1 \end{bmatrix} \begin{array}{l} (x_2=1, x_3=0) \\ (x_2=1, x_3=1) \end{array} \times 1.1$$

$$= \begin{bmatrix} 1.1 \\ 1.199 \end{bmatrix} \begin{array}{l} (x_3=0) \\ (x_3=1) \end{array}$$

Thus, $P(x_3 \mid x_1 = 0) = \frac{1}{Z} \mu_{f_3 \to x_3}(x_3)$

$$= \frac{1}{Z} \begin{bmatrix} 1.1 \\ 1.199 \end{bmatrix}$$

$$= \frac{1}{1.1 + 1.199} \begin{bmatrix} 1.1 \\ 1.199 \end{bmatrix}$$

$$\boxed{= \begin{bmatrix} 0.48 \\ 0.52 \end{bmatrix} \begin{array}{l} (P(x_3=0 \mid x_1=0)) \\ (P(x_3=1 \mid x_1=0)) \end{array}}$$

Gradient Descent

Given N training data points $\{(\boldsymbol{x^k}, y^k)\}$ for $k = \{1, \ldots, N\}$, $\boldsymbol{x}^k \in R^d$, and ground truth y^k, we seek a linear regressor $f(\boldsymbol{x}) = \boldsymbol{w^T x}$ optimizing the loss function $L(z) = \left(y - f(x)\right)^4$

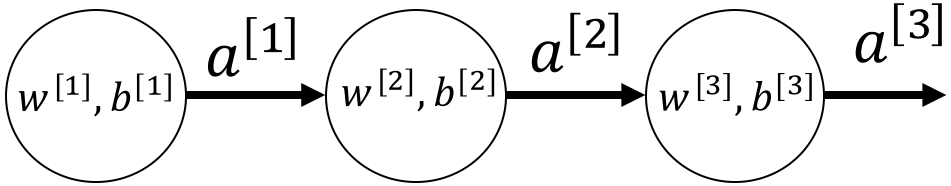
1. Find the gradient and gradient descent update equation to find $\boldsymbol{w}$.

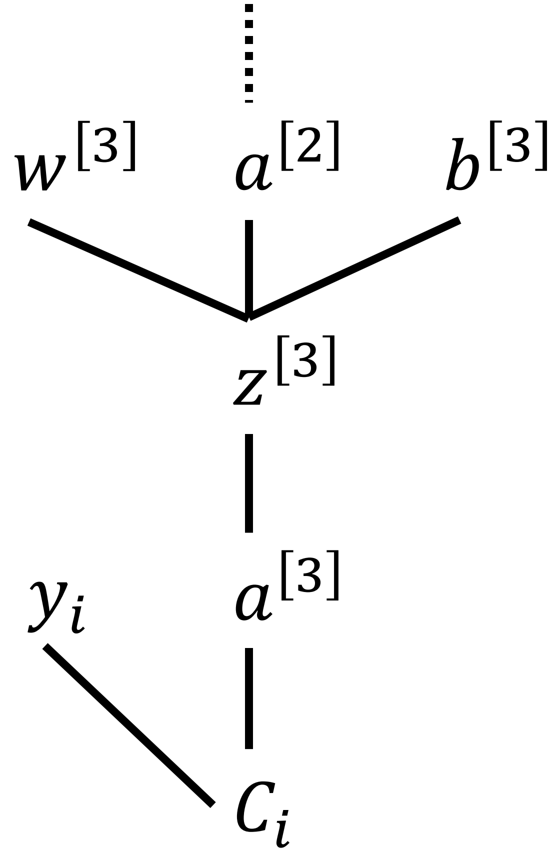
2. Suppose you also want to include a penalty term $\lambda||\boldsymbol{w}||^2$ to the overall loss function. Derive the gradient for gradient descent to update $\boldsymbol{w}$.

Back Propagation: Chain Rule

Apply *chain rule of derivative* to update parameters:

$$\boldsymbol{w^{[1]}, b^{[1]}, w^{[2]}, b^{[2]}}$$



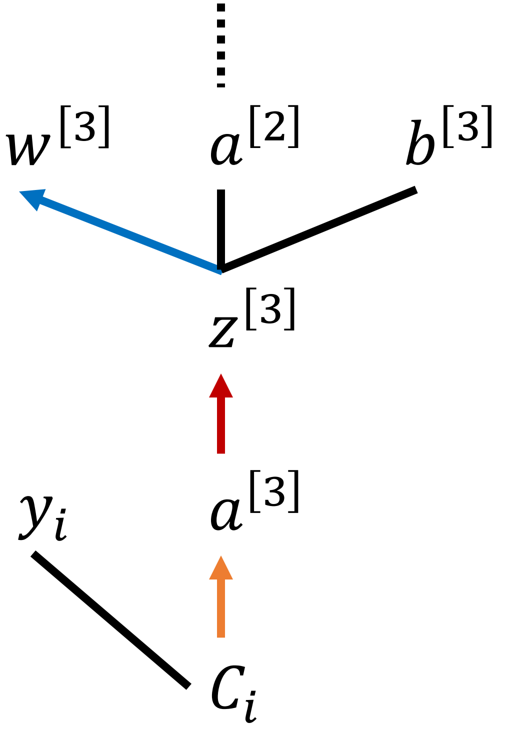


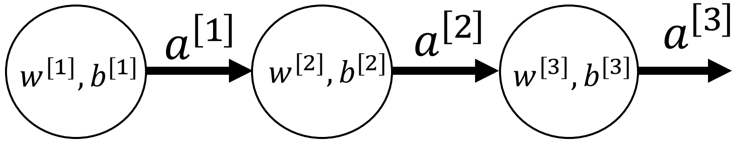
- Our goal is to figure out how to update the weights to minimize the cost in the next iteration of gradient descent
- To begin, we wish to calculate $\frac{\partial C_i}{\partial w^{[3]}}$
- Recall that

$$\begin{aligned} C_i &= \left(a^{[3]} - y_i\right)^2 \\ &= \left(\sigma\left(z^{[3]}\right) - y_i\right)^2 \\ &= \left(\sigma\left(w^{[3]}a^{[2]} + b^{[3]}\right) - y_i\right)^2 \end{aligned}$$

- We can visualize this relationship with the dependency tree to the left

Back Propagation: Chain Rule





- Now, let's compute $\frac{\partial C_i}{\partial w^{[3]}}$

$$\frac{\partial C_i}{\partial w^{[3]}} = \frac{\partial C_i}{\partial a^{[3]}} \frac{\partial a^{[3]}}{\partial z^{[3]}} \frac{\partial z^{[3]}}{\partial w^{[3]}}$$

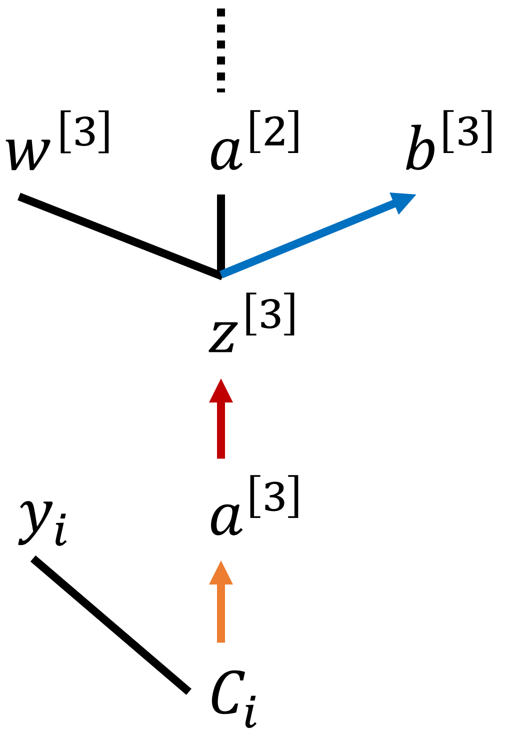
$$\frac{\partial C_i}{\partial a^{[3]}} = \frac{\partial \left(a^{[3]} - y_i\right)^2}{\partial a^{[3]}} = 2\left(a^{[3]} - y_i\right)$$

$$\frac{\partial a^{[3]}}{\partial z^{[3]}} = \frac{\partial \sigma\left(z^{[3]}\right)}{\partial z^{[3]}} = \sigma\left(z^{[3]}\right)\left(1 - \sigma\left(z^{[3]}\right)\right)$$

$$\frac{\partial z^{[3]}}{\partial w^{[3]}} = \frac{\partial \left(w^{[3]} a^{[2]} + b^{[3]}\right)}{\partial w^{[3]}} = a^{[2]}$$

$$\Rightarrow \frac{\partial C_i}{\partial w^{[3]}} = 2\left(a^{[3]} - y_i\right)\sigma\left(z^{[3]}\right)\left(1 - \sigma\left(z^{[3]}\right)\right)a^{[2]}$$

Back Propagation: Chain Rule



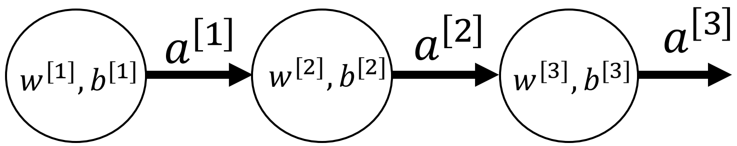
- Similarly, we can compute $\frac{\partial C_i}{\partial b^{[3]}}$

$$\frac{\partial C_i}{\partial b^{[3]}} = \frac{\partial C_i}{\partial a^{[3]}} \frac{\partial a^{[3]}}{\partial z^{[3]}} \frac{\partial z^{[3]}}{\partial b^{[3]}}$$

$$\frac{\partial C_i}{\partial a^{[3]}} = \frac{\partial \left(a^{[3]} - y_i\right)^2}{\partial a^{[3]}} = 2\left(a^{[3]} - y_i\right)$$

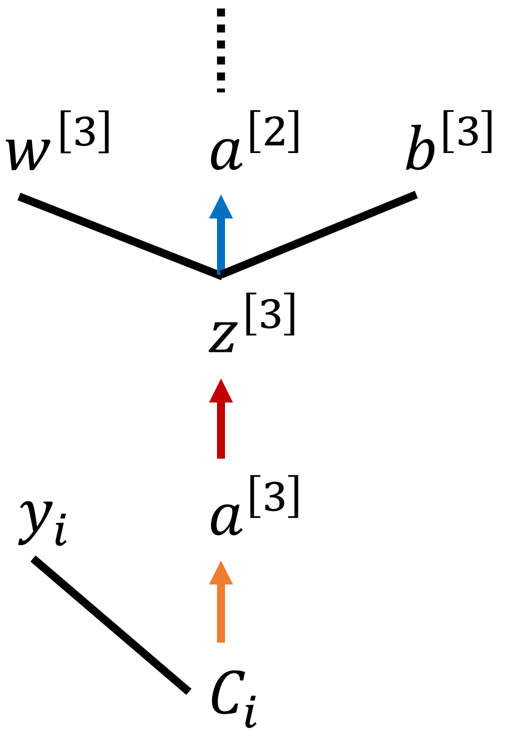
$$\frac{\partial a^{[3]}}{\partial z^{[3]}} = \frac{\partial \sigma\left(z^{[3]}\right)}{\partial z^{[3]}} = \sigma\left(z^{[3]}\right)\left(1 - \sigma\left(z^{[3]}\right)\right)$$

$$\frac{\partial z^{[3]}}{\partial b^{[3]}} = \frac{\partial \left(w^{[3]} a^{[2]} + b^{[3]}\right)}{\partial b^{[3]}} = 1$$



$$\Rightarrow \frac{\partial C_i}{\partial b^{[3]}} = 2\left(a^{[3]} - y_i\right)\sigma\left(z^{[3]}\right)\left(1 - \sigma\left(z^{[3]}\right)\right)$$

Back Propagation: Chain Rule



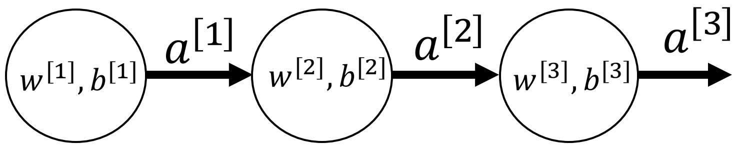
- Finally, we compute $\frac{\partial C_i}{\partial a^{[2]}}$

$$\frac{\partial C_i}{\partial a^{[2]}} = \frac{\partial C_i}{\partial a^{[3]}} \frac{\partial a^{[3]}}{\partial z^{[3]}} \frac{\partial z^{[3]}}{\partial a^{[2]}}$$

$$\frac{\partial C_i}{\partial a^{[3]}} = \frac{\partial \left(a^{[3]} - y_i\right)^2}{\partial a^{[3]}} = 2\left(a^{[3]} - y_i\right)$$

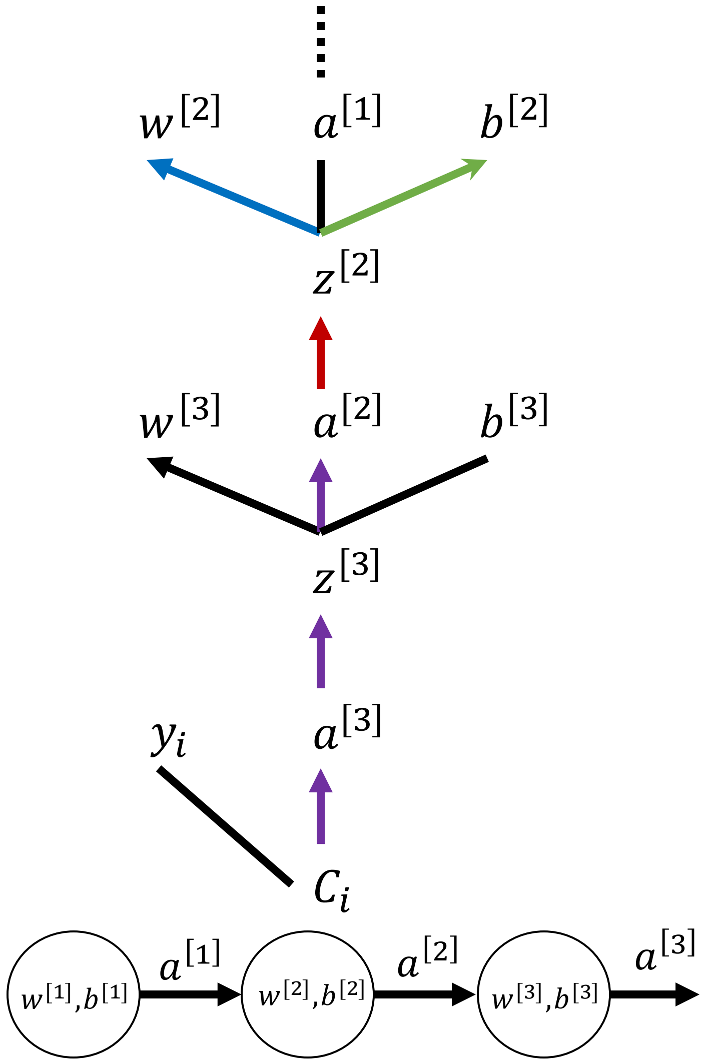
$$\frac{\partial a^{[3]}}{\partial z^{[3]}} = \frac{\partial \sigma\left(z^{[3]}\right)}{\partial z^{[3]}} = \sigma\left(z^{[3]}\right)\left(1 - \sigma\left(z^{[3]}\right)\right)$$

$$\frac{\partial z^{[3]}}{\partial a^{[2]}} = \frac{\partial \left(w^{[3]} a^{[2]} + b^{[3]}\right)}{\partial a^{[2]}} = w^{[3]}$$



$$\Rightarrow \frac{\partial C_i}{\partial a^{[2]}} = 2\left(a^{[3]} - y_i\right)\sigma\left(z^{[3]}\right)\left(1 - \sigma\left(z^{[3]}\right)\right) w^{[3]}$$

Back Propagation: Chain Rule



- We can now *propagate* the gradient calculations we have done *backwards* in order to find the gradients for weights/biases in layer 2

Has been computed already in previous slide

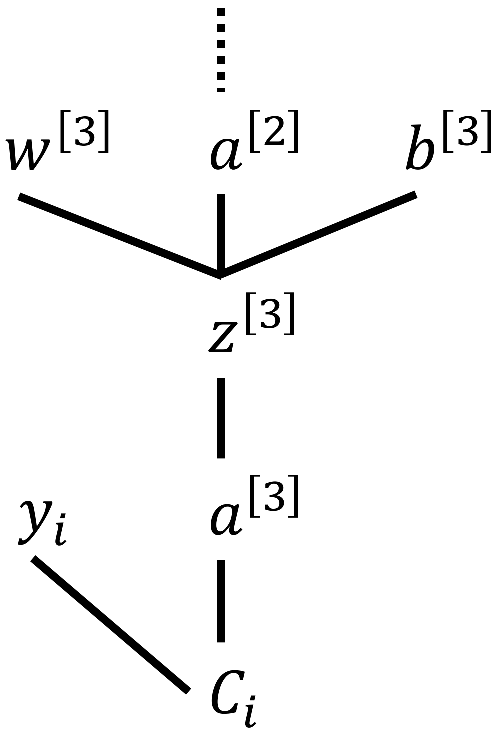
$$\frac{\partial C_i}{\partial w^{[2]}} = \frac{\partial C_i}{\partial a^{[3]}} \frac{\partial a^{[3]}}{\partial z^{[3]}} \frac{\partial z^{[3]}}{\partial a^{[2]}} \frac{\partial a^{[2]}}{\partial z^{[2]}} \frac{\partial z^{[2]}}{\partial w^{[2]}}$$

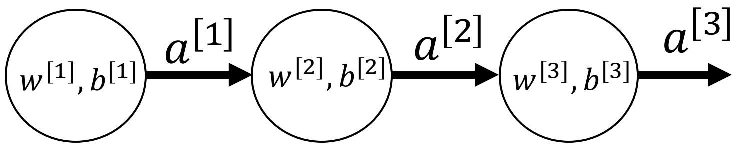
$$= \frac{\partial C_i}{\partial a^{[2]}} \sigma(z^{[2]})(1 - \sigma(z^{[2]}))a^{[1]}$$

$$\frac{\partial C_i}{\partial b^{[2]}} = \frac{\partial C_i}{\partial a^{[3]}} \frac{\partial a^{[3]}}{\partial z^{[3]}} \frac{\partial z^{[3]}}{\partial a^{[2]}} \frac{\partial a^{[2]}}{\partial z^{[2]}} \frac{\partial z^{[2]}}{\partial b^{[2]}}$$

$$= \frac{\partial C_i}{\partial a^{[2]}} \sigma(z^{[2]})(1 - \sigma(z^{[2]}))$$

Back Propagation: Full Cost Function





- In order to the partial derivative of the full cost function C with respect to a weight (say, $w^{[3]}$), we need to average the gradients of cost with respect to that weight for all n training samples

$$\frac{\partial C}{\partial w^{[3]}} = \frac{1}{n}\sum_{i=1}^{n}\frac{\partial C_i}{\partial w^{[3]}}$$

Apply *chain rule of derivative* to update perceptron Model parameters:, $\boldsymbol{b}^{[1]}, \boldsymbol{w}^{[2]}, \boldsymbol{b}^{[2]}$

- The core of the neural network is perceptron model

Forward

$\boldsymbol{x} = [x_1, x_2, x_3]$

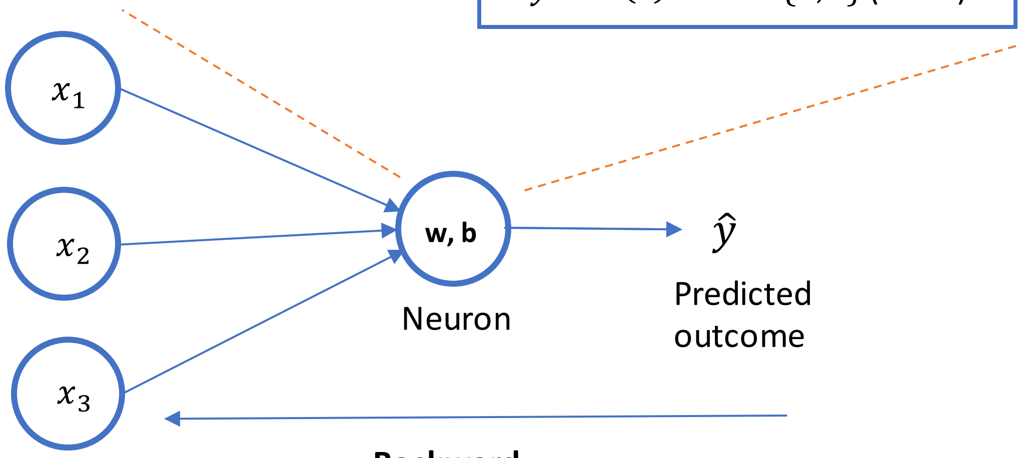
$z = \boldsymbol{w}^T\boldsymbol{x} + \mathrm{b}$

Activation Unit

$\hat{y} = \sigma(z) = \frac{1}{1+e^{-z}}$ (Sigmoid)

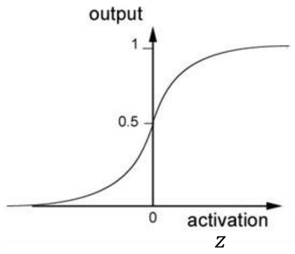
Or

$\hat{y} = R(z) = \max\{0, z\}$ (ReLU)



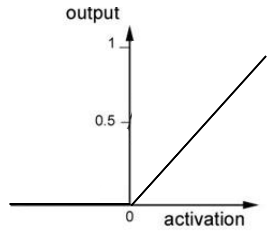
Backward

Sigmoid Function



$$\sigma(z) = \frac{1}{1+e^{-z}}$$

ReLU Function



$\mathrm{R}(z) = \max\{0, z\}$

Update Rule (Backward):

$$w_{t+1} = w_t - \eta \nabla J(w_t)$$

η **: Learning rate**

Loss

$$J(w) = \frac{1}{N}\sum_{i=1}^{N} L(w.x^{(i)}, y^{(i)})$$

N: number of samples $\quad$ $x^{(i)}$: feature of i^{th} sample

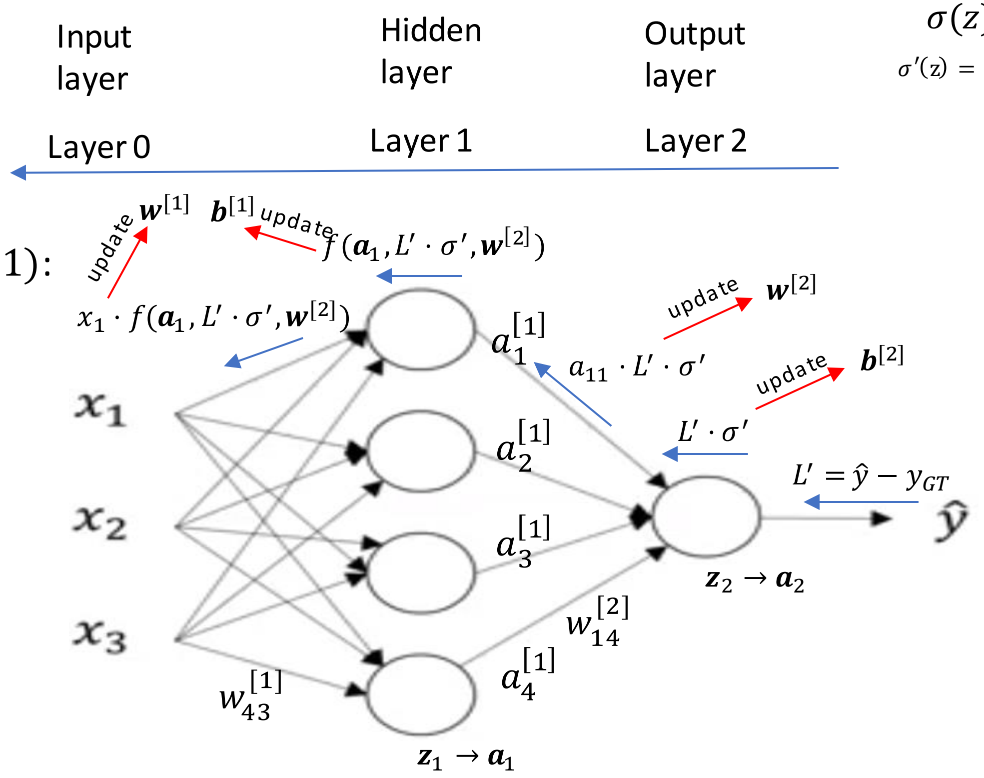
Computing Gradient

$$\nabla J(\mathbf{w}_0) = \left(\frac{\partial J(\mathbf{w})}{\partial w_0}, \frac{\partial J(\mathbf{w})}{\partial w_1}, \cdots, \frac{\partial J(\mathbf{w})}{\partial w_n}\right)_{\mathbf{w}_0}$$

Backpropagation: Gradient Descent

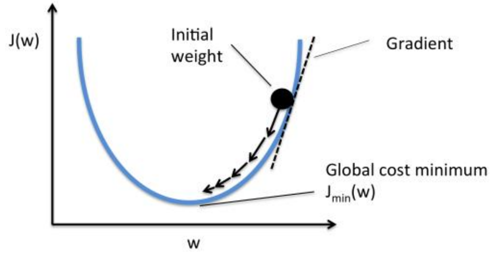
Backward

Assume 1 sample ($N = 1$):



Derivative of Sigmoid function $\sigma(z)$:

$$
\begin{aligned}
\sigma'(z) = \frac{\partial \sigma(z)}{\partial z} &= \frac{\partial}{\partial z}\frac{1}{1+e^{-z}} \\
&= \frac{e^{-z}}{(1+e^{-z})^2} \\
&= \frac{1+e^{-z}-1}{(1+e^{-z})^2} \\
&= \frac{1}{1+e^{-z}} - \frac{1}{(1+e^{-z})^2} \\
&= \sigma(z) - \sigma^2(z) \\
&= \sigma(z)\big(1-\sigma(z)\big)
\end{aligned}
$$



Purpose of backpropagation:

Apply *chain rule of derivative* to update parameters: $w^{[1]}, b^{[1]}, w^{[2]}, b^{[2]}$

Gradient Descent

1. Find the gradient and gradient descent update equation to find $\boldsymbol{w}$.

$$L = \sum_{i=1}^{N} (y^i - w^T x^i)^4$$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} \qquad \nabla_w L = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial w_2} \\ \vdots \\ \frac{\partial L}{\partial w_d} \end{bmatrix}$$

$$\frac{\partial L}{\partial w_1} = \sum_{i=1}^{N} 4(y^i - w^T x^i)^3 (-x_1^i)$$

$$\frac{\partial L}{\partial w_2} = \sum_{i=1}^{N} 4(y^i - w^T x^i)^3 (-x_2^i)$$

$$\vdots$$

$$\frac{\partial L}{\partial w_d} = \sum_{i=1}^{N} 4(y^i - w^T x^i)^3 (-x_d^i)$$

$$\nabla_w L = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial w_2} \\ \vdots \\ \frac{\partial L}{\partial w_d} \end{bmatrix} = \sum_{i=1}^{N} 4(y^i - w^T x^i)^3 \begin{bmatrix} -x_1^i \\ -x_2^i \\ \vdots \\ -x_d^i \end{bmatrix} = -\sum_{i=1}^{N} 4(y^i - w^T x^i)^3 x^i$$

$$w^{t+1} = w^t - \eta \nabla_w L$$

$$= w^t + \eta \sum_{i=1}^{N} 4(y^i - w^{tT} x^i)^3 x^i$$

(Note x^i is a k-dimensional vector)

2. Suppose you also want to include a penalty term $\lambda||\boldsymbol{w}||^2$ to the overall loss function. Derive the gradient for gradient descent to update $\boldsymbol{w}$.

$$L = \sum_{i=1}^{N} (y^i - w^T x^i)^4 + \lambda ||w||^2$$

gradient: $\nabla_w L = -\sum_{i=1}^{N} 4(y^i - w^T x^i)^3 x^i + 2\lambda w$

gradient update: $w^{t+1} = w^t - \eta \nabla_w L$

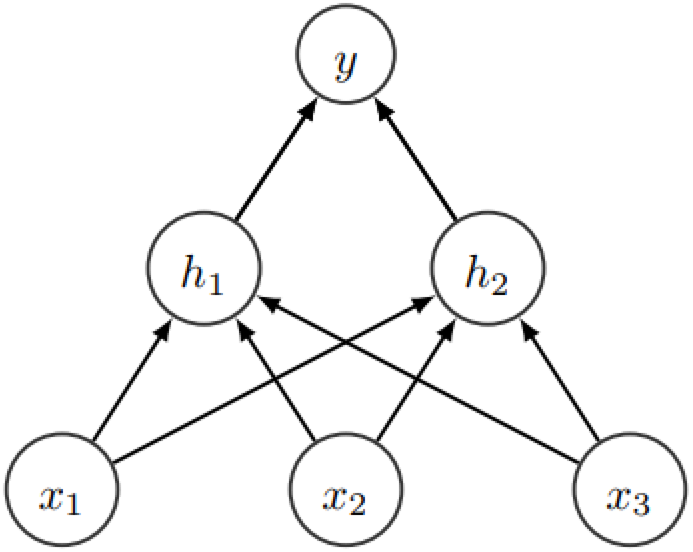
$$= w^t - \eta\left(-\sum_{i=1}^{N} 4(y^i - w^{tT} x^i)^3 x^i + 2\lambda w^t\right)$$

$$= (1 - 2\eta\lambda) w^t + \eta \sum_{i=1}^{N} 4(y^i - w^{tT} x^i)^3 x^i$$

(Note both w^t and x^i are k-dimensional vectors)

Neural Networks Backpropagation

Consider the neural network given alongside. The hidden units and output layer has ReLU activation function. The loss function is given by $L(y, y) = \frac{1}{2}(y - t)^2$ where t is the target value. For simplicity, assume that the bias terms are 0. Weights connecting input to hidden layer and hidden layer to output layer are given by W and V respectively.



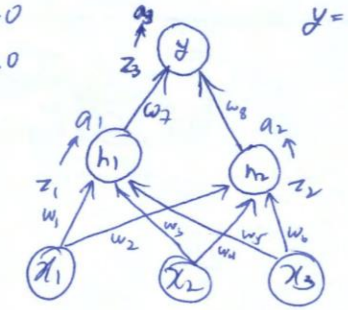
1. Write the forward equation to map input to output.
2. Compute the output and backpropagation for $x = [1,2,1]$ and $t = 1$.

$$W = \begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$$

$$V = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

Neural Network Backprop:

1. $a_3 = R(z_3)$ where $R(x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases}$



$y = a_3$.

$$z_3 = a_1 w_7 + a_2 w_8$$
$$a_1 = R(z_1) \ ; \ a_2 = R(z_2)$$
$$z_1 = w_1 x_1 + w_3 x_2 + w_5 x_3$$
$$z_2 = w_2 x_1 + w_4 x_2 + w_6 x_3$$

$$\therefore y = R\left(w_7 \cdot R(w_1 x_1 + w_3 x_2 + w_5 x_3) + w_8 \cdot R(w_2 x_1 + w_4 x_2 + w_6 x_3)\right)$$

2. $W = \begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 & w_5 \\ w_2 & w_4 & w_6 \end{bmatrix}$ $\quad V = \begin{bmatrix} 0 & 1 \end{bmatrix} = [w_7 \ w_8$

substituting $x = [1, 2, 1] = [x_1, x_2, x_3]$, we get:

$z_1 = 2$, $z_2 = -1 \Rightarrow a_1 = 2$, $a_2 = 0$

$\Rightarrow z_3 = 0 \Rightarrow a_3 = 0 \Rightarrow y = 0$.

Backpropagation:

$$L = \frac{1}{2}(y-t)^2 = \frac{1}{2}(0-1)^2 = \frac{1}{2}.$$

use: $\delta(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$

$$\frac{\partial L}{\partial w_7} = \frac{\partial L}{\partial a_3} \cdot \frac{\partial a_3}{\partial z_3} \cdot \frac{\partial z_3}{\partial w_7} = (y-t) \cdot \delta(z_3) \cdot a_1$$
$$= (0-1) \cdot \delta(0) \cdot 2$$
$$= 0.$$

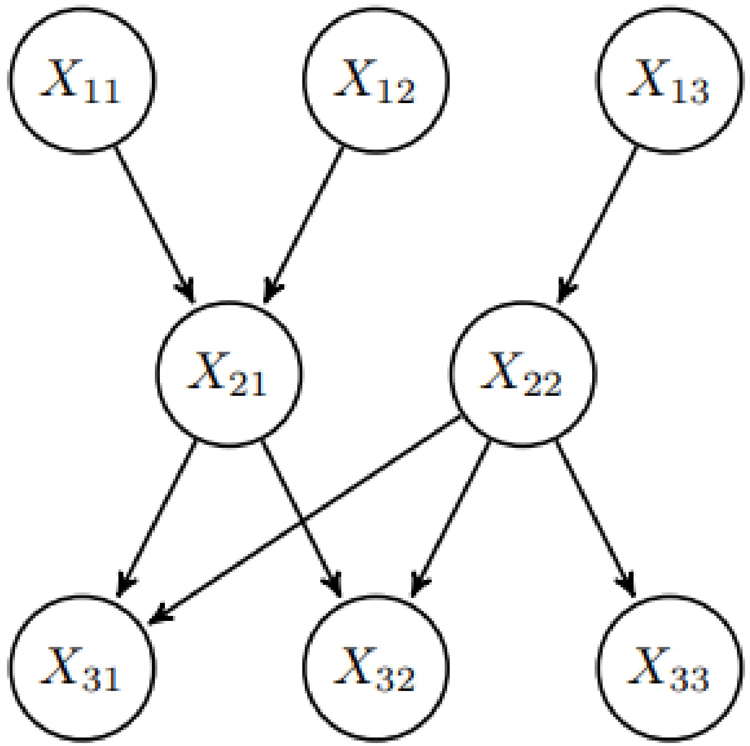
$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial a_3} \cdot \frac{\partial a_3}{\partial z_3} \cdot \frac{\partial z_3}{\partial a_1} \cdot \frac{\partial a_1}{\partial z_1} \cdot \frac{\partial z_1}{\partial w_1} = (y-t) \cdot \delta(z_3) \cdot w_7 \cdot \delta(z_1) \cdot x_1$$
$$= (0-1) \cdot \delta(0) \cdot 0 \cdot \delta(2) \cdot 1$$
$$= 0$$

similar computations would give

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial w_4} = \frac{\partial L}{\partial w_5} = \frac{\partial L}{\partial w_6} = \frac{\partial L}{\partial w_8} = 0$$

Bayesian Network Question

Consider the Bayesian Network alongside with binary variables. Answer the following questions.



1. Is there any variable(s) conditionally independent of X_{33} given X_{11} and X_{12}? If so, list all.

2. Is there any variable(s) conditionally independent of X_{33} given X_{22}? If so, list all.

3. How many parameters are required to specify the factorized joint distribution?

4. Express $P(X_{13} = 0, X_{22} = 1, X_{33} = 0)$ in terms of the conditional probabilities from the Bayesian Network.

Bayesian Network question

1). $X_{21} \perp X_{33} \mid X_{11}, X_{12}$

by local semantics. Note that for node X_{21}, X_{11} and X_{12} are its parents and are observed, while X_{33} is its non-descena

2). For node X_{33}, X_{22} is its parent and is observed. All the nodes in the graph except X_{22} are its non-descendents. Therefore, by local semantics, X_{33} is conditionally independent of all nodes (except X_{22}) given X_{22}.

3). All variables are binary. Number of parameters is:

$$\underset{X_{11}}{1} + \underset{X_{12}}{1} + \underset{X_{13}}{1} + \underset{X_{21}}{4} + \underset{X_{22}}{2} + \underset{X_{31}}{4} + \underset{X_{32}}{4} + \underset{X_{33}}{2} = 19$$

factorized distribution: $P(X_{11})P(X_{12})P(X_{13})\,P(X_{21}|X_{11},X_{12}).\,P(X_{22}|X_{13}).$
$P(X_{31}|X_{21},X_{22}).\,P(X_{32}|X_{21},X_{22}).\,P(X_{33}|X_{22})$

4)
$$P(X_{13},X_{22},X_{33}) = \sum_{X_{11},X_{12},X_{21},X_{31},X_{32}} P(X_{11},X_{12},X_{13},X_{21},X_{22},X_{31},X_{32},X_{33})$$

$$= \sum_{X_{11},X_{12},X_{21},X_{31},X_{32}} P(X_{11})P(X_{12})P(X_{13})P(X_{21}|X_{11},X_{12})\,P(X_{22}|X_{13})\,P(X_{31}|X_{21},X_{\cdot}$$
$$P(X_{32}|X_{21},X_{22})\,P(X_{33}|.$$

$$= P(X_{13})P(X_{22}|X_{13})P(X_{33}|X_{22}).\sum_{X_{11}}P(X_{11}).\ \sum_{X_{12}}P(X_{12}).\sum_{X_{21}}P(X_{21}|X_{11},X_{12})$$
$$\sum_{X_{31}}P(X_{31}|X_{21},X_{22}).\ \sum_{X_{32}}P(X_{32}|X_{21},X_{22})$$

$$= P(X_{13})\ P(X_{22}|X_{13})\ P(X_{33}|X_{22}).$$

$\therefore P(X_{13}=0, X_{22}=1, X_{33}=0) = P(X_{13}=0).\ P(X_{22}=1|X_{13}=0).\ P(X_{33}=0|X_{22}=$