10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Hidden Markov Models

Matt Gormley
Lecture 19
Nov. 5, 2018

Reminders

- Homework 6: PAC Learning / Generative Models
  - Out: Wed, Oct 31
  - Due: Wed, Nov 7 at 11:59pm (1 week)
- Homework 7: HMMs
  - Out: Wed, Nov 7
  - Due: Mon, Nov 19 at 11:59pm

- Grades are up on Canvas

Q&A

Q: Why would we use Naïve Bayes? Isn't it too Naïve?

A: Naïve Bayes has one **key advantage** over methods like Perceptron, Logistic Regression, Neural Nets:

Training is lightning fast!

While other methods require slow iterative training procedures that might require hundreds of epochs, Naïve Bayes computes its parameters in closed form by counting.

DISCRIMINATIVE AND GENERATIVE CLASSIFIERS

Generative vs. Discriminative

- **Generative Classifiers:**
  - Example: Naïve Bayes
  - Define a joint model of the observations **x** and the labels y: $p(\boldsymbol{x}, y)$
  - Learning maximizes (joint) likelihood
  - Use Bayes' Rule to classify based on the posterior:

$$p(y|\mathbf{x}) = p(\mathbf{x}|y)p(y)/p(\mathbf{x})$$

- **Discriminative Classifiers:**
  - Example: Logistic Regression
  - Directly model the conditional: $p(y|\mathbf{x})$
  - Learning maximizes conditional likelihood

Generative vs. Discriminative

Whiteboard

– Contrast: To model p(x) or not to model p(x)?

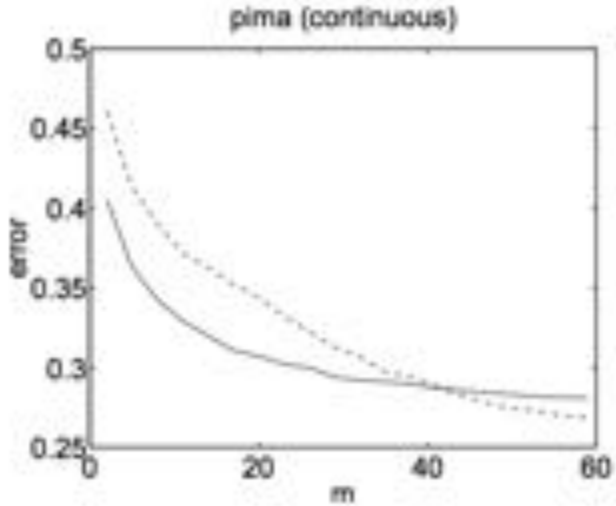
Generative vs. Discriminative

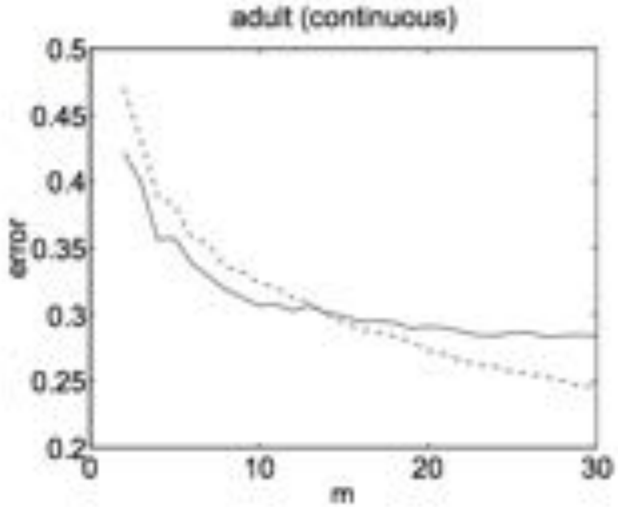
Finite Sample Analysis (Ng & Jordan, 2002)

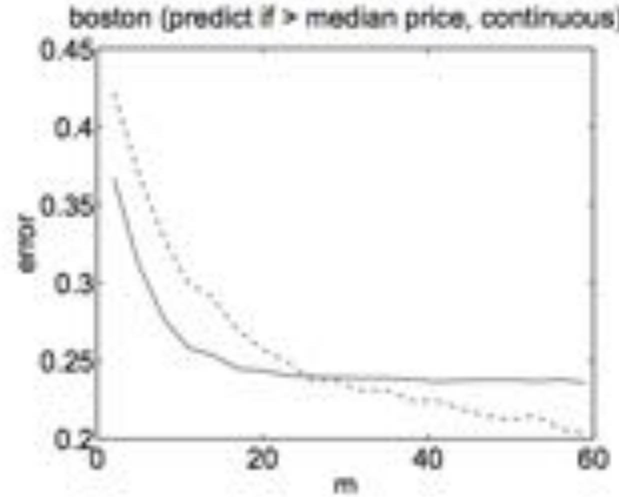
[Assume that we are learning from a finite training dataset]

If model assumptions are correct: Naive Bayes is a more efficient learner (requires fewer samples) than Logistic Regression

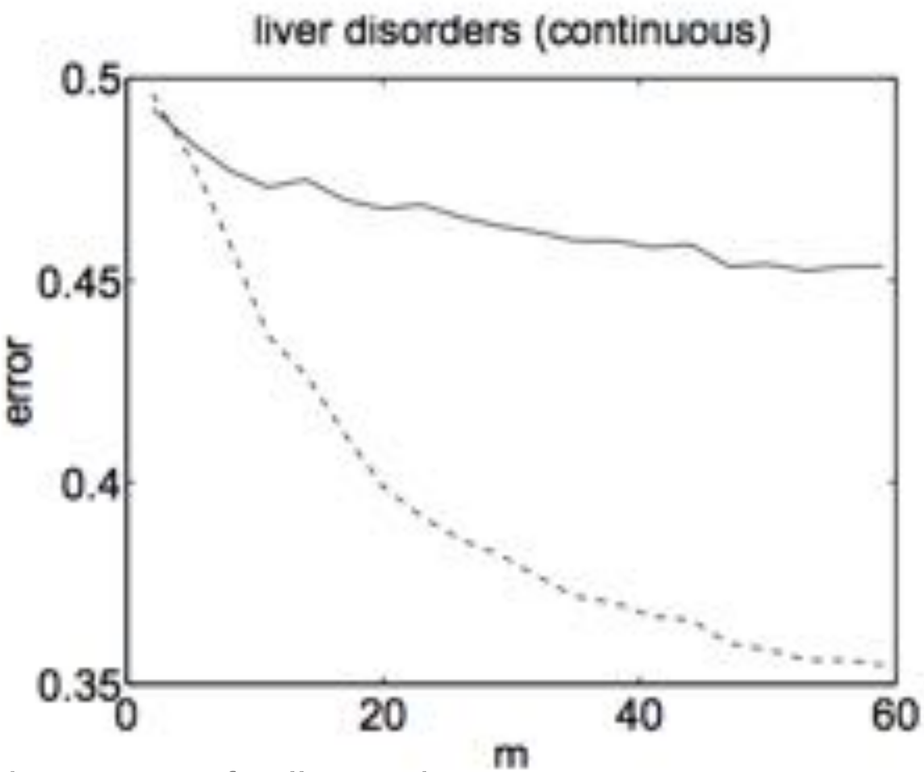
If model assumptions are incorrect: Logistic Regression has lower asymtotic error, and does better than Naïve Bayes

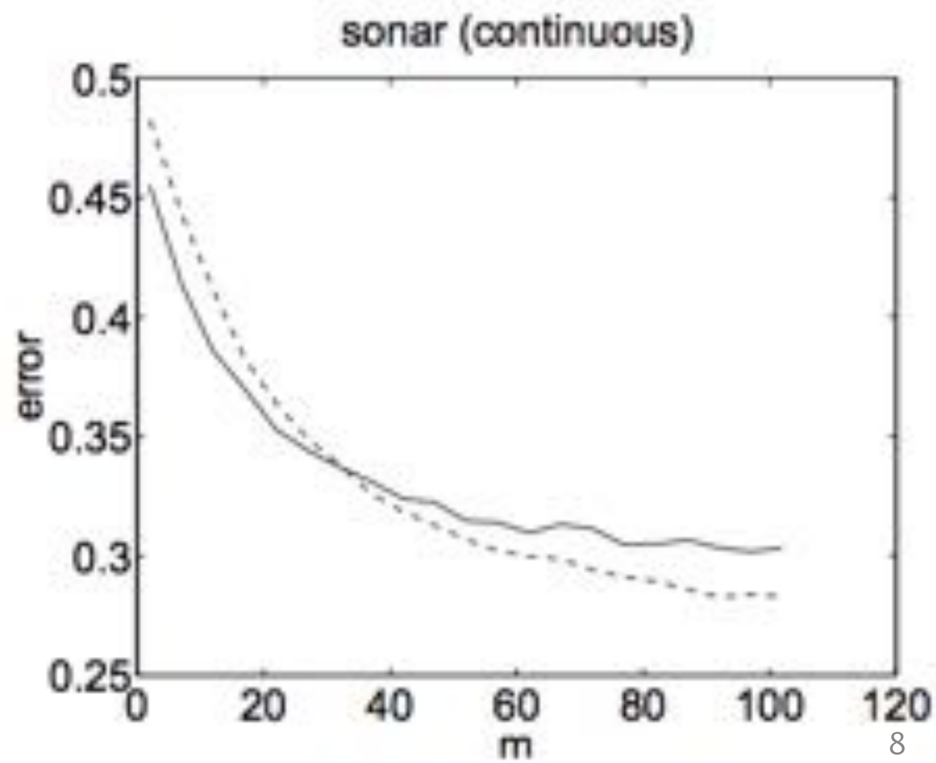






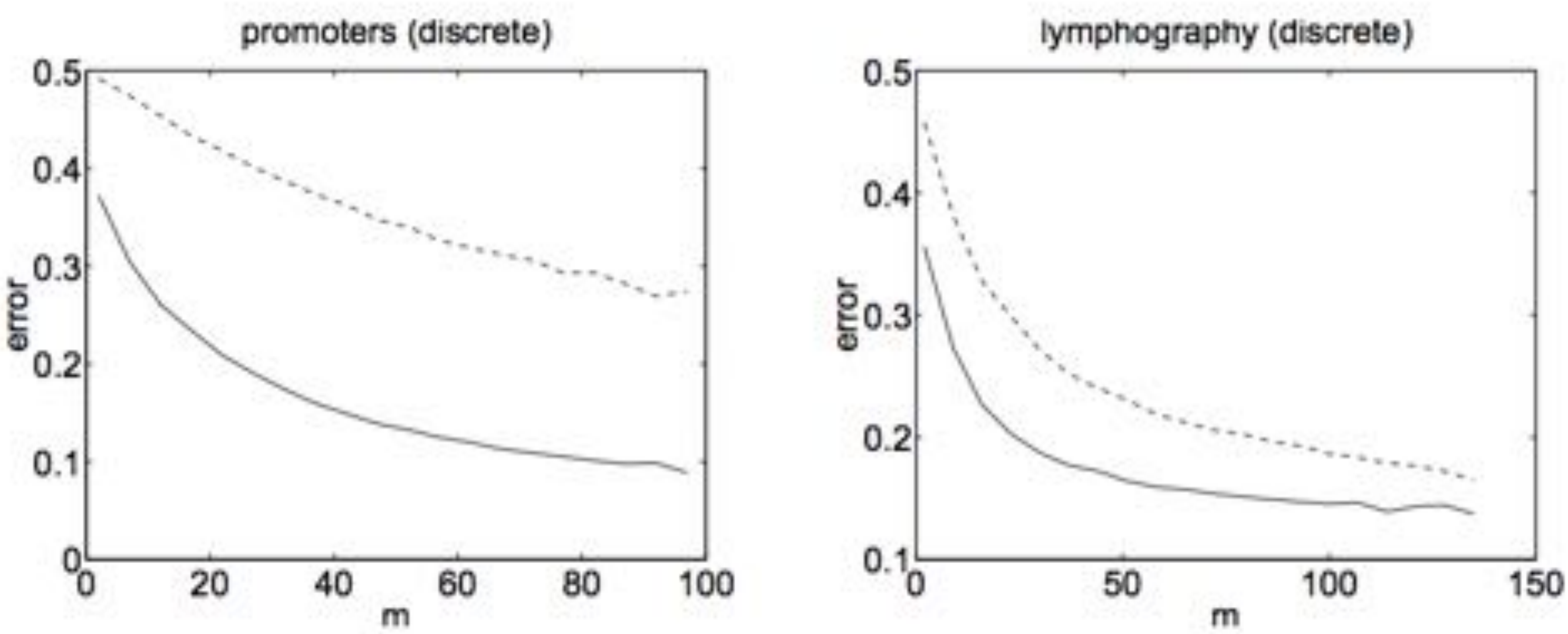
solid: NB dashed: LR





Slide courtesy of William Cohen

solid: NB dashed: LR



Naïve Bayes makes stronger assumptions about the data but needs fewer examples to estimate the parameters

"On Discriminative vs Generative Classifiers: …." Andrew Ng and Michael Jordan, NIPS 2001.

Generative vs. Discriminative

Learning (Parameter Estimation)

Naïve Bayes:
Parameters are decoupled → Closed form solution for MLE

Logistic Regression:
Parameters are coupled → No closed form solution – must use iterative optimization techniques instead

Naïve Bayes vs. Logistic Reg.

Learning (MAP Estimation of Parameters)

Bernoulli Naïve Bayes:
Parameters are probabilities → Beta prior (usually) pushes probabilities away from zero / one extremes

Logistic Regression:
Parameters are not probabilities → Gaussian prior encourages parameters to be close to zero

(effectively pushes the probabilities away from zero / one extremes)

Naïve Bayes vs. Logistic Reg.

Features

Naïve Bayes:
Features $\boldsymbol{x}$ are assumed to be conditionally independent given y. (i.e. Naïve Bayes Assumption)

Logistic Regression:
No assumptions are made about the form of the features $\boldsymbol{x}$. They can be dependent and correlated in any fashion.

MOTIVATION: STRUCTURED PREDICTION

Structured Prediction

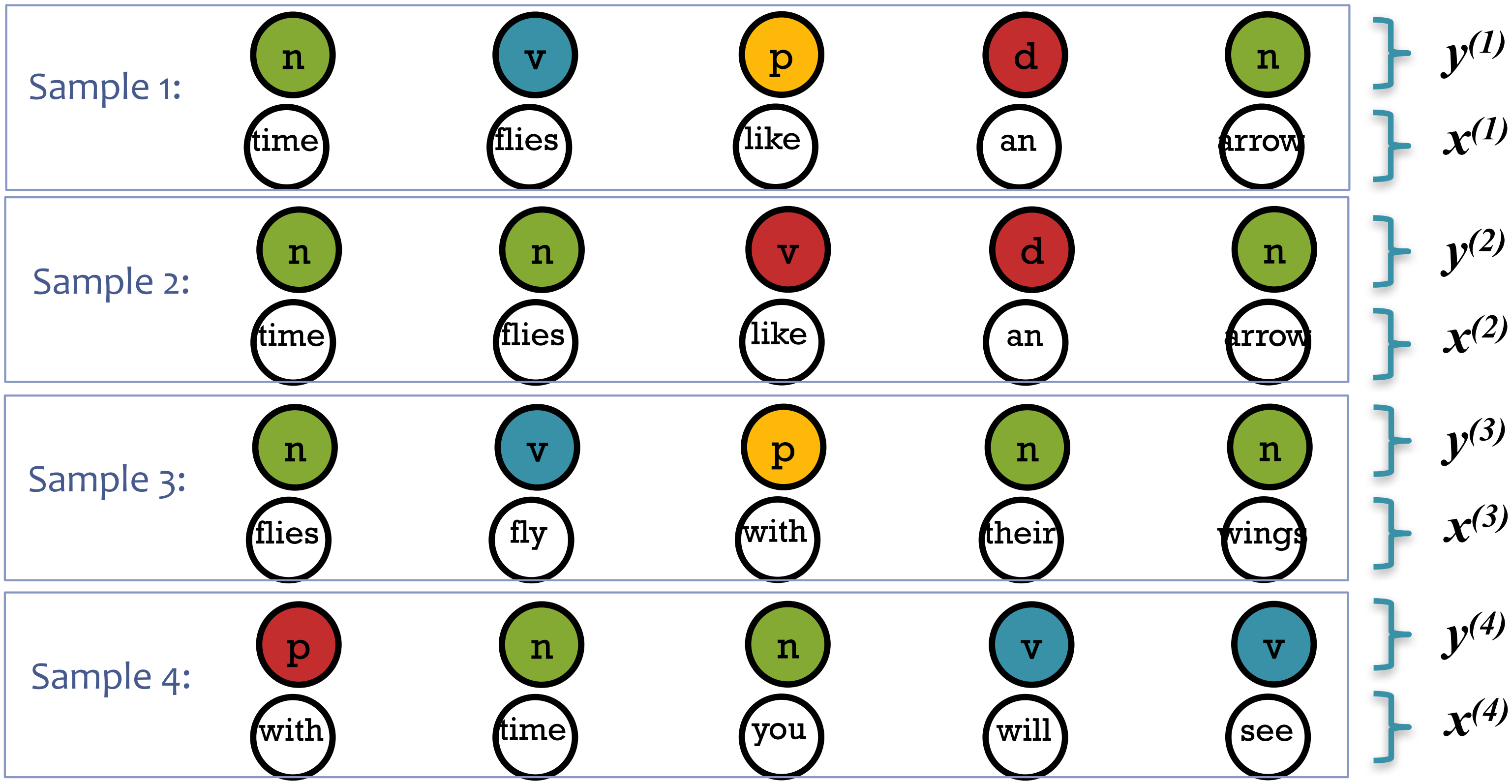
- Most of the models we've seen so far were for **classification**
  - Given observations: $\boldsymbol{x} = (x_1, x_2, \ldots, x_K)$
  - Predict a (binary) **label:** y
- Many real-world problems require **structured prediction**
  - Given observations: $\boldsymbol{x} = (x_1, x_2, \ldots, x_K)$
  - Predict a **structure:** $\boldsymbol{y} = (y_1, y_2, \ldots, y_J)$
- Some *classification* problems benefit from **latent structure**

Structured Prediction Examples

- **Examples of structured prediction**
  - Part-of-speech (POS) tagging
  - Handwriting recognition
  - Speech recognition
  - Word alignment
  - Congressional voting
- **Examples of latent structure**
  - Object recognition

Dataset for Supervised Part-of-Speech (POS) Tagging

Data: $\mathcal{D} = \{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N}$



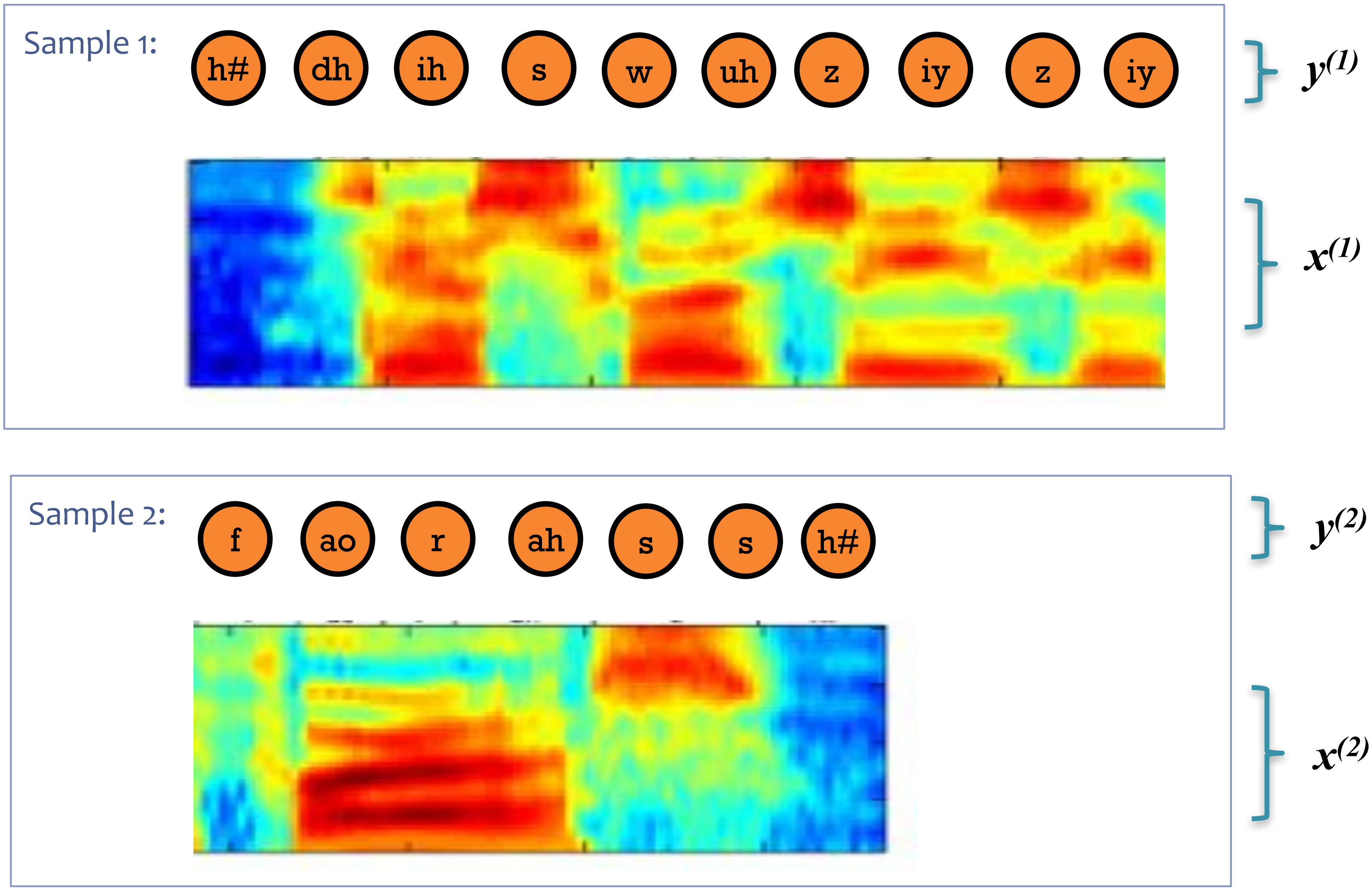
Dataset for Supervised Handwriting Recognition

Data: $\mathcal{D} = \{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N}$

Figures from (Chatzis & Demiris, 2013)

Dataset for Supervised Phoneme (Speech) Recognition

Data: $\mathcal{D} = \{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N}$

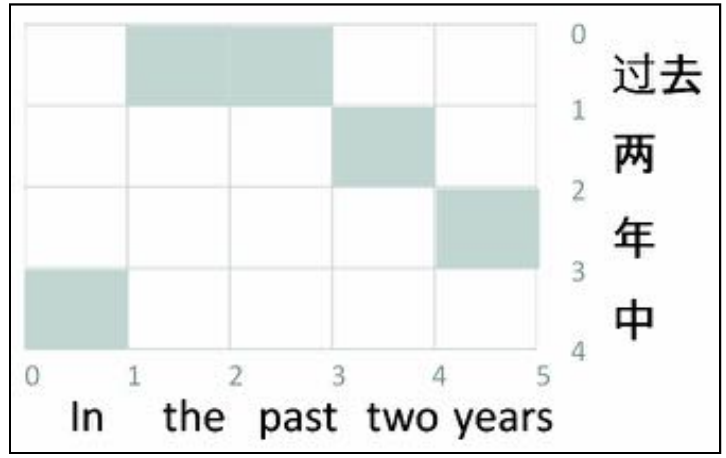


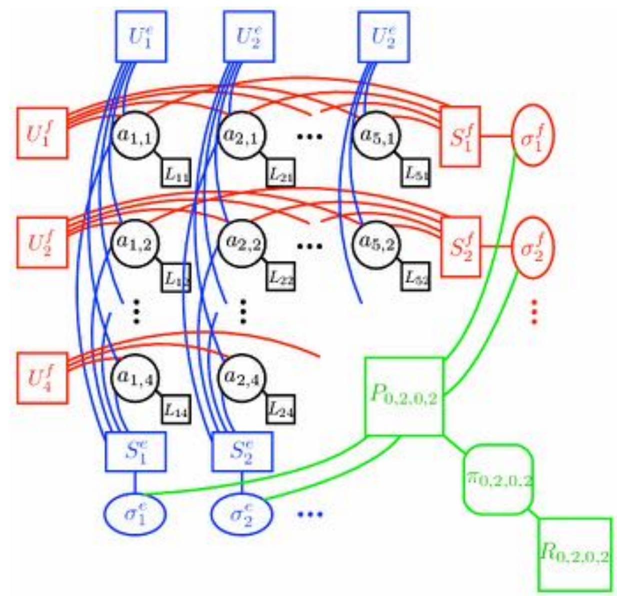
Figures from (Jansen & Niyogi, 2013)

Application:

Word Alignment / Phrase Extraction

- **Variables (boolean):**
  - For each (Chinese phrase, English phrase) pair, are they linked?
- **Interactions:**
  - Word fertilities
  - Few “jumps” (discontinuities)
  - Syntactic reorderings
  - “ITG contraint” on alignment
  - Phrases are disjoint (?)

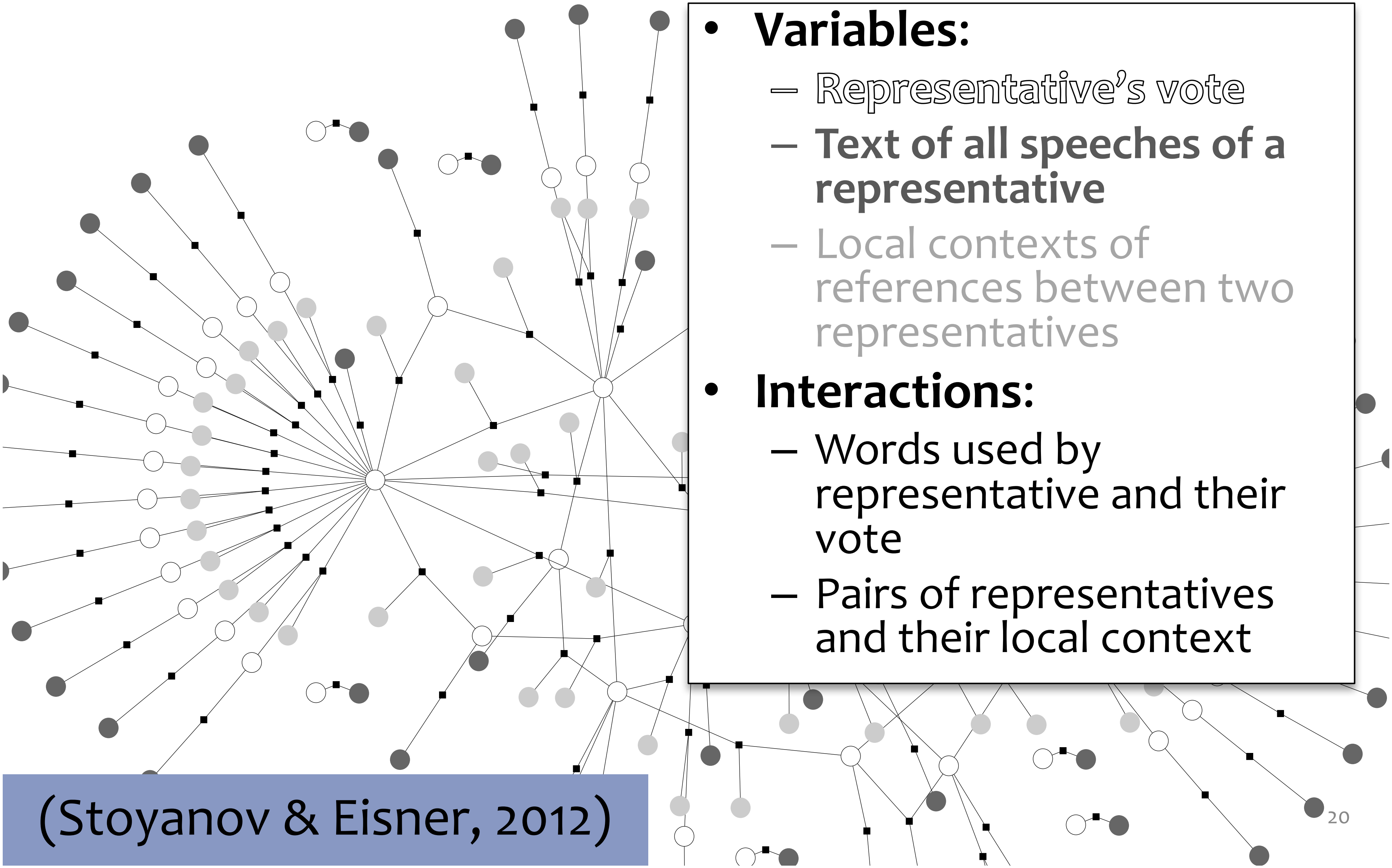




(Burkett & Klein, 2012)

Application:

Congressional Voting



- **Variables:**
  - Representative's vote
  - **Text of all speeches of a representative**
  - Local contexts of references between two representatives
- **Interactions:**
  - Words used by representative and their vote
  - Pairs of representatives and their local context

(Stoyanov & Eisner, 2012)

Structured Prediction Examples

- **Examples of structured prediction**
  - Part-of-speech (POS) tagging
  - Handwriting recognition
  - Speech recognition
  - Word alignment
  - Congressional voting
- **Examples of latent structure**
  - Object recognition

Case Study: Object Recognition

Data consists of images $\boldsymbol{x}$ and labels y.

$\boldsymbol{x}^{(1)}$

pigeon

$y^{(1)}$

$\boldsymbol{x}^{(2)}$

rhinoceros

$y^{(2)}$

$\boldsymbol{x}^{(3)}$

leopard

$y^{(3)}$

$\boldsymbol{x}^{(4)}$

llama

$y^{(4)}$

Case Study: Object Recognition

Data consists of images $\boldsymbol{x}$ and labels y.

- Preprocess data into "patches"
- Posit a latent labeling z describing the object's parts (e.g. head, leg, tail, torso, grass)

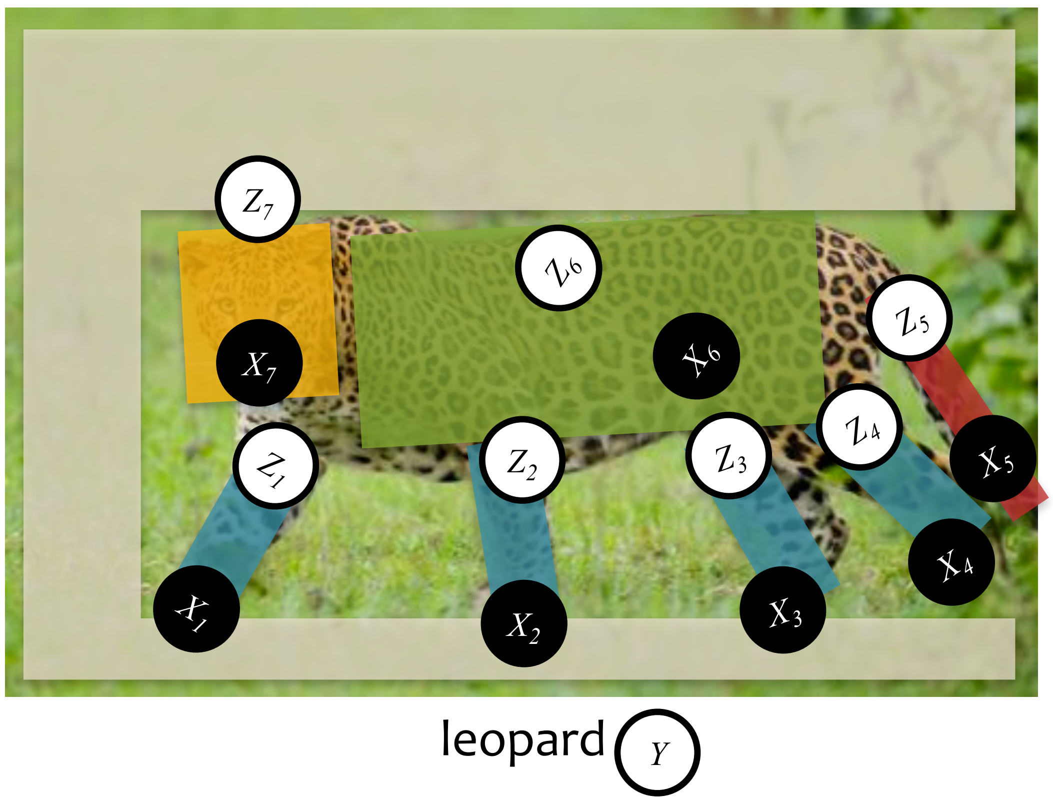
- Define graphical model with these latent variables in mind
- z is not observed at train or test time

leopard

Case Study: Object Recognition

Data consists of images $\boldsymbol{x}$ and labels y.

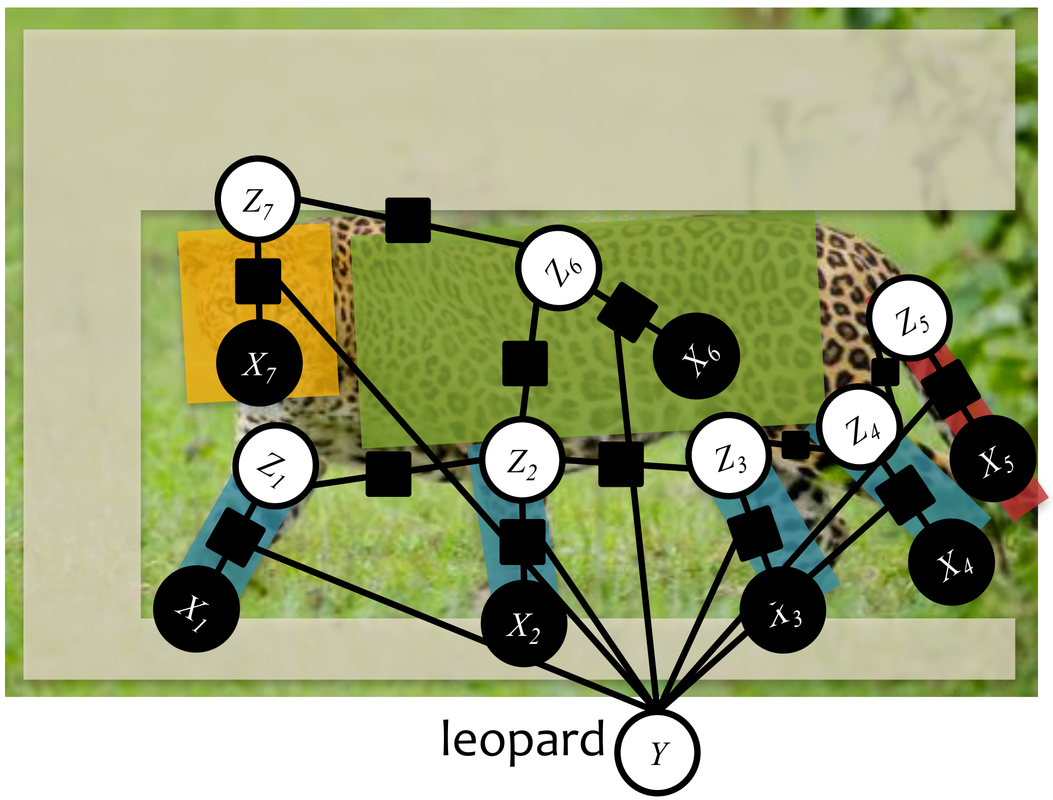
- Preprocess data into "patches"
- Posit a latent labeling z describing the object's parts (e.g. head, leg, tail, torso, grass)
- Define graphical model with these latent variables in mind
- z is not observed at train or test time



Case Study: Object Recognition

Data consists of images $\boldsymbol{x}$ and labels y.

- Preprocess data into "patches"
- Posit a latent labeling z describing the object's parts (e.g. head, leg, tail, torso, grass)
- Define graphical model with these latent variables in mind
- z is not observed at train or test time



Structured Prediction

Preview of challenges to come…

- Consider the task of finding the **most probable assignment** to the output

Classification

$$\hat{y} = \operatorname*{argmax}_{y} p(y|\mathbf{x})$$

where $y \in \{+1, -1\}$

Structured Prediction

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$$

where $\mathbf{y} \in \mathcal{Y}$

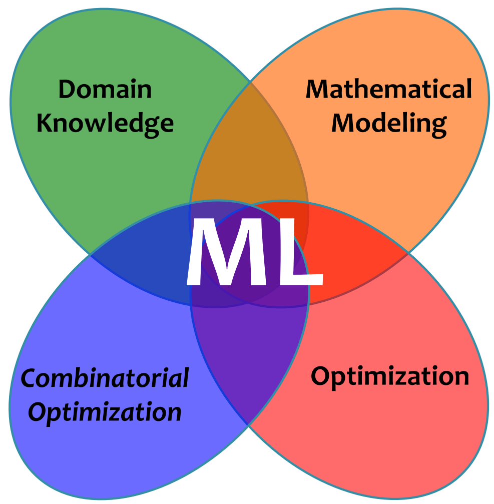
and $|\mathcal{Y}|$ is very large

Machine Learning

The **data** inspires the structures we want to predict

Our **model** defines a score for each structure

It also tells us what to optimize



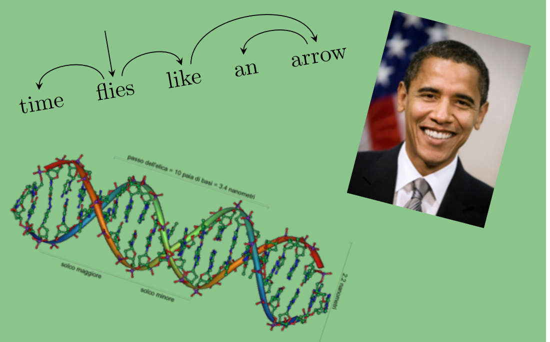
Learning tunes the parameters of the model

Inference finds {best structure, marginals, partition function} for a new observation

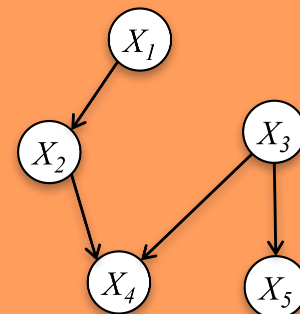
(**Inference** is usually called as a subroutine in learning)

Machine Learning

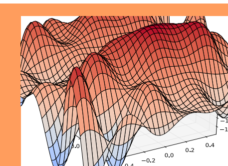
Data



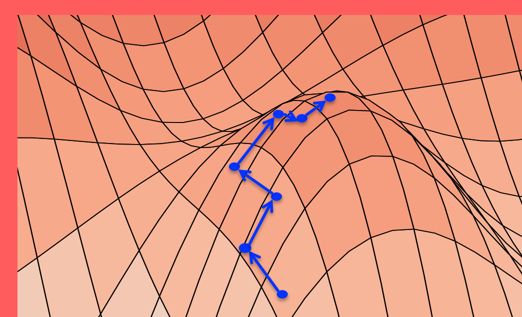
Model



Objective



Learning



Inference

(**Inference** is usually called as a subroutine in learning)

BACKGROUND

Background: Chain Rule of Probability

For random variables A and B:

$$P(A, B) = P(A|B)P(B)$$

For random variables X_1, X_2, X_3, X_4:

$$\begin{aligned} P(X_1, X_2, X_3, X_4) = & P(X_1|X_2, X_3, X_4) \\ & P(X_2|X_3, X_4) \\ & P(X_3|X_4) \\ & P(X_4) \end{aligned}$$

Background: Conditional Independence

Random variables A and B are conditionally independent given C if:

$$P(A, B|C) = P(A|C)P(B|C) \qquad (1)$$

or equivalently:

$$P(A|B, C) = P(A|C) \qquad (2)$$

We write this as:

$$A \perp\!\!\!\perp B | C$$

Later we will also write: $I\langle A, \{C\}, B\rangle$

HIDDEN MARKOV MODEL (HMM)

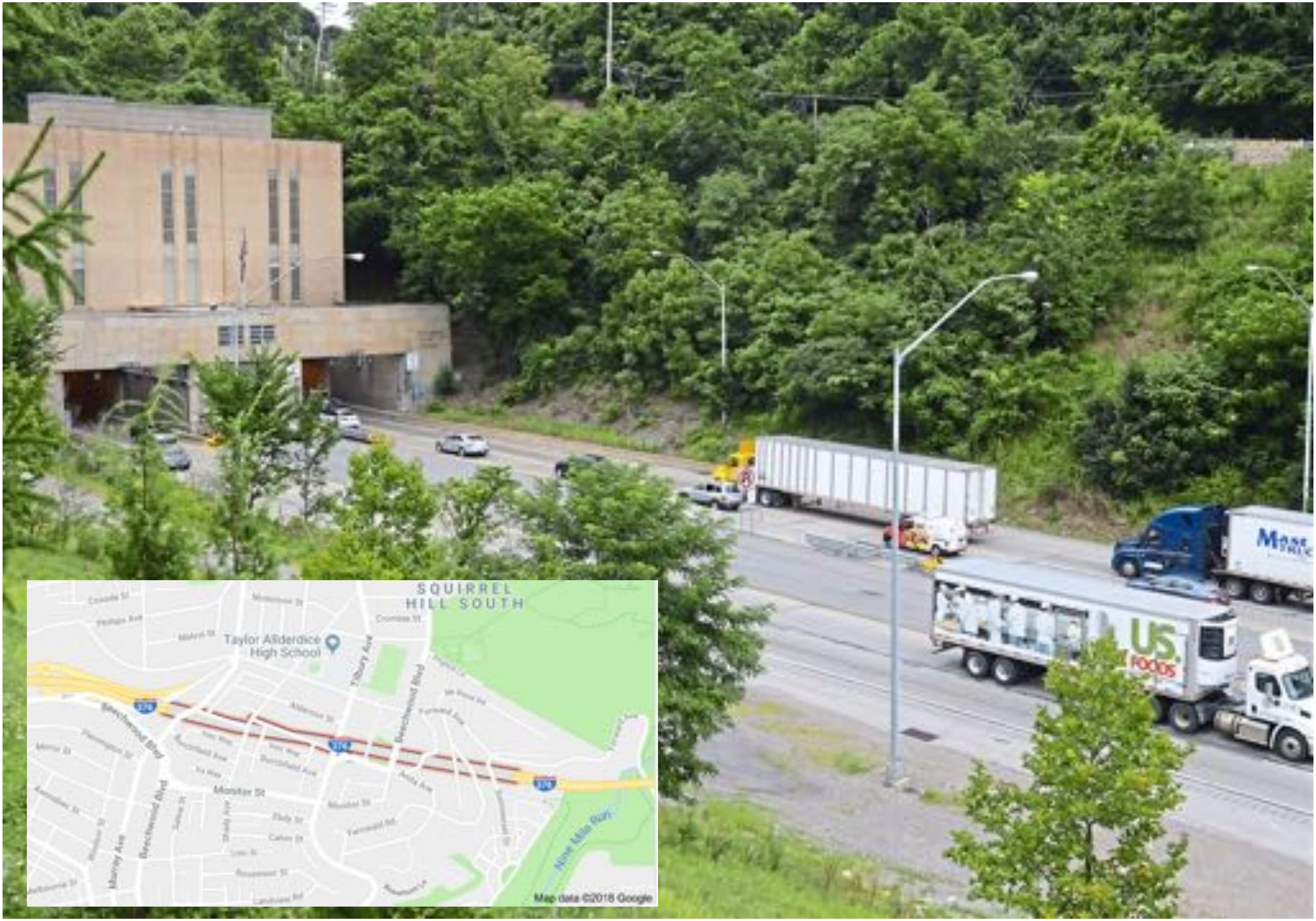
HMM Outline

- **Motivation**
  - Time Series Data
- **Hidden Markov Model (HMM)**
  - Example: Squirrel Hill Tunnel Closures [courtesy of Roni Rosenfeld]
  - Background: Markov Models
  - From Mixture Model to HMM
  - History of HMMs
  - Higher-order HMMs
- **Training HMMs**
  - (Supervised) Likelihood for HMM
  - Maximum Likelihood Estimation (MLE) for HMM
  - EM for HMM (aka. Baum-Welch algorithm)
- **Forward-Backward Algorithm**
  - Three Inference Problems for HMM
  - Great Ideas in ML: Message Passing
  - Example: Forward-Backward on 3-word Sentence
  - Derivation of Forward Algorithm
  - Forward-Backward Algorithm
  - Viterbi algorithm

Markov Models

Whiteboard

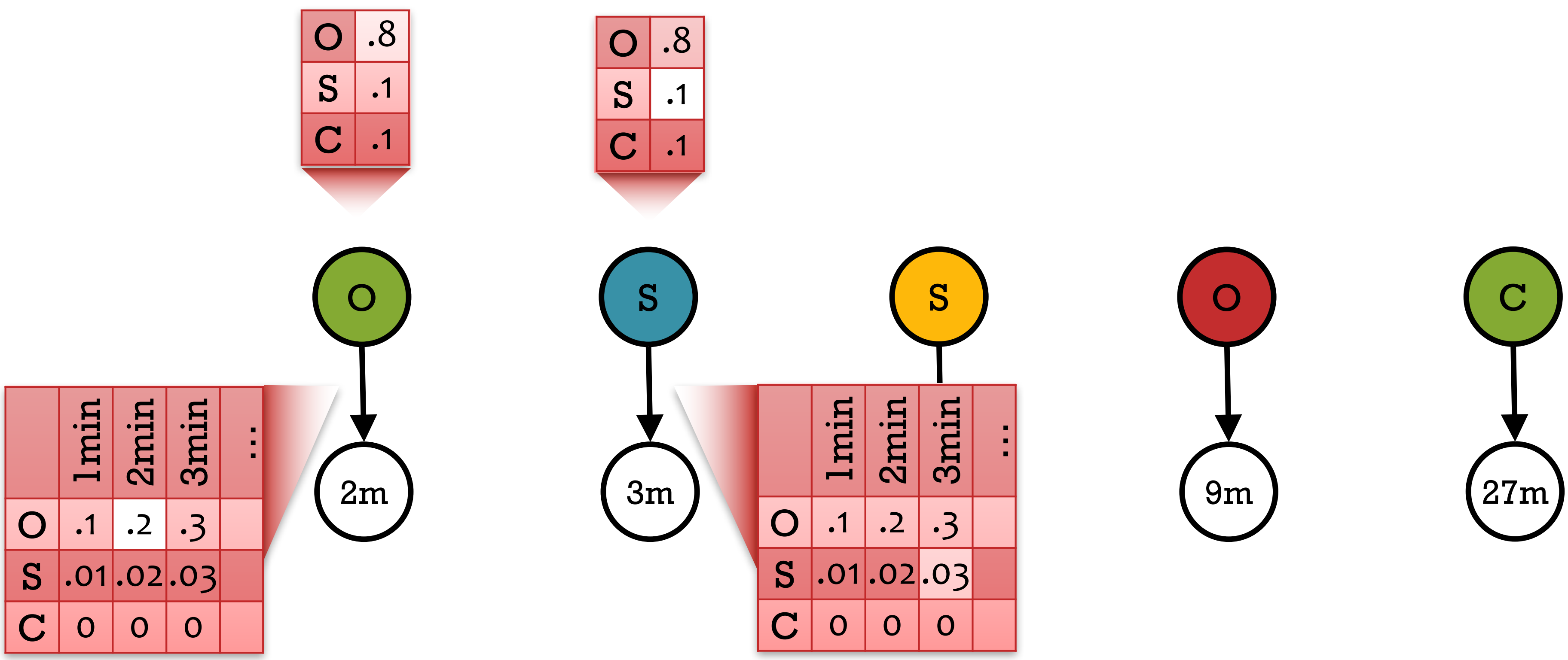
- Example: Squirrel Hill Tunnel Closures [courtesy of Roni Rosenfeld]
- First-order Markov assumption
- Conditional independence assumptions


SQUIRREL
HILL SOUTH
Taylor Allderdice
High School
Tilbury Ave
Beechwood Blvd
Murray Ave
Nine Mile Run
Map data ©2018 Google
US
FOODS

Mixture Model for Time Series Data

We could treat each (tunnel state, travel time) pair as independent. This corresponds to a Naïve Bayes model with a single feature (travel time).

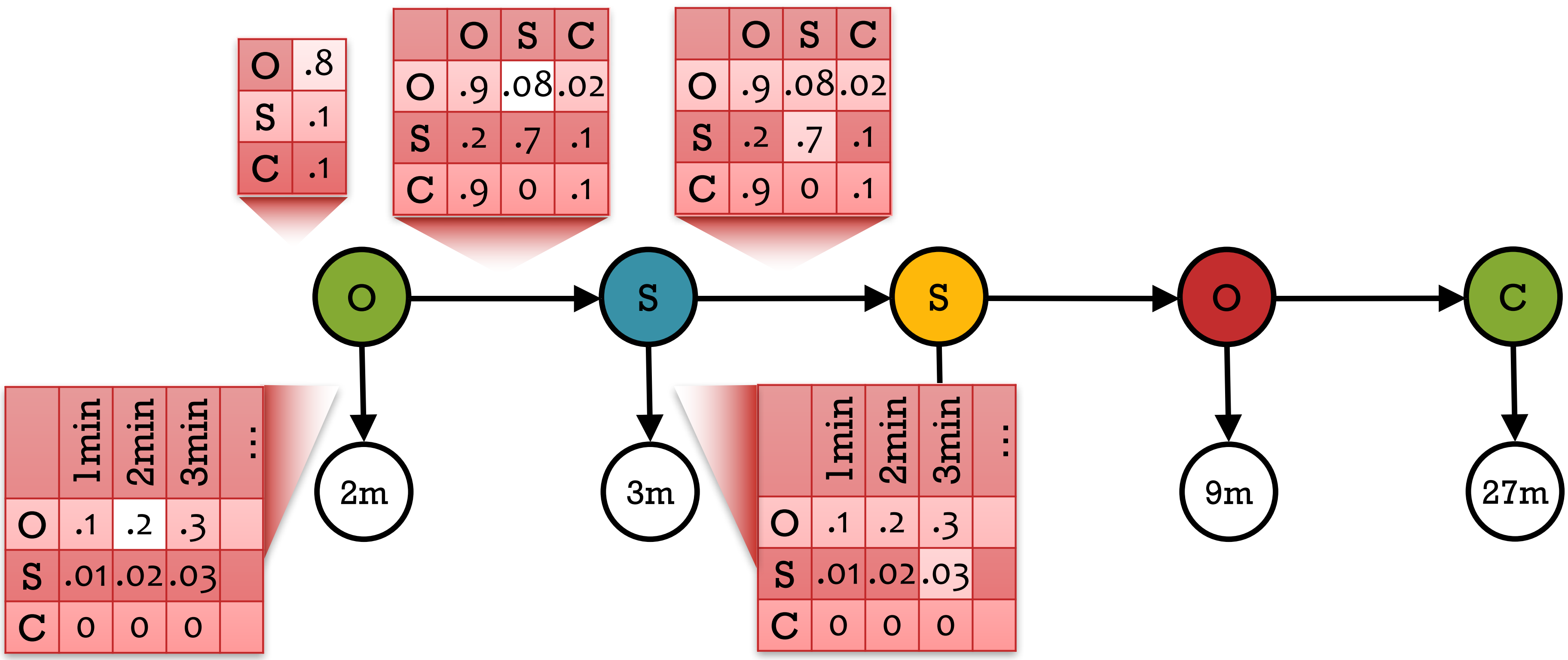
$$p(\mathrm{O}, \mathrm{S}, \mathrm{S}, \mathrm{O}, \mathrm{C}, 2\mathrm{m}, 3\mathrm{m}, 18\mathrm{m}, 9\mathrm{m}, 27\mathrm{m}) \quad = \quad (.8 * .2 * .1 * .03 * \ldots)$$



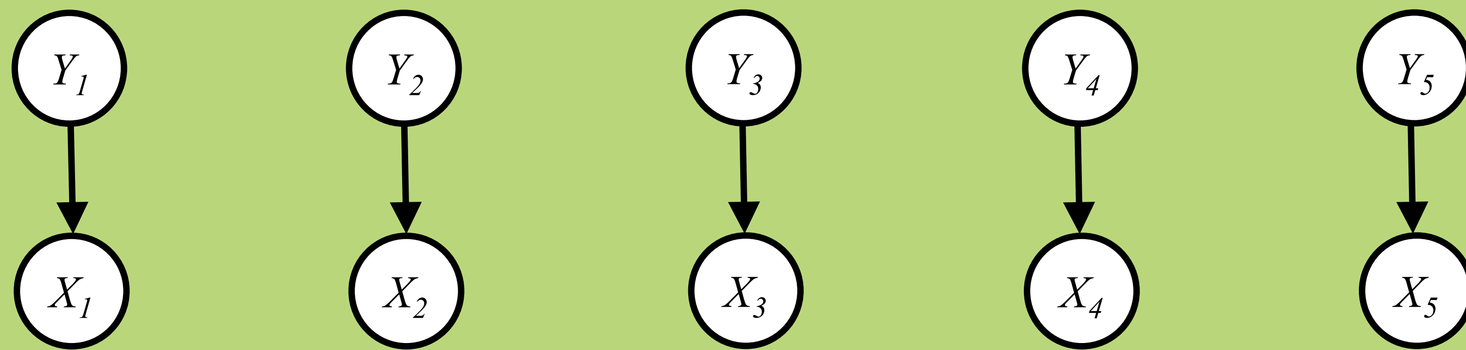
Hidden Markov Model

A Hidden Markov Model (HMM) provides a joint distribution over the the tunnel states / travel times with an assumption of dependence between adjacent tunnel states.

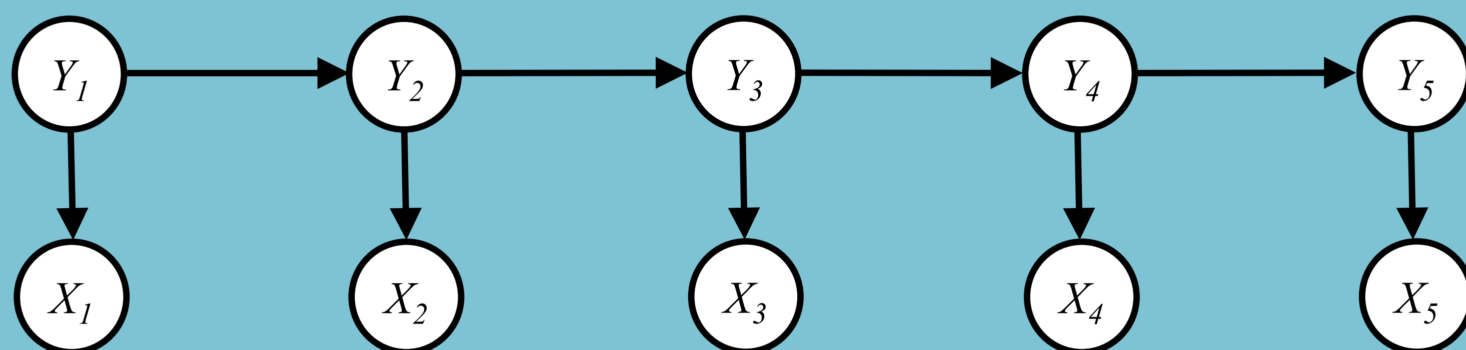
$$p(\mathrm{O}, \mathrm{S}, \mathrm{S}, \mathrm{O}, \mathrm{C}, 2\mathrm{m}, 3\mathrm{m}, 18\mathrm{m}, 9\mathrm{m}, 27\mathrm{m}) \quad = (.8 * .08 * .2 * .7 * .03 * \ldots)$$



From Mixture Model to HMM

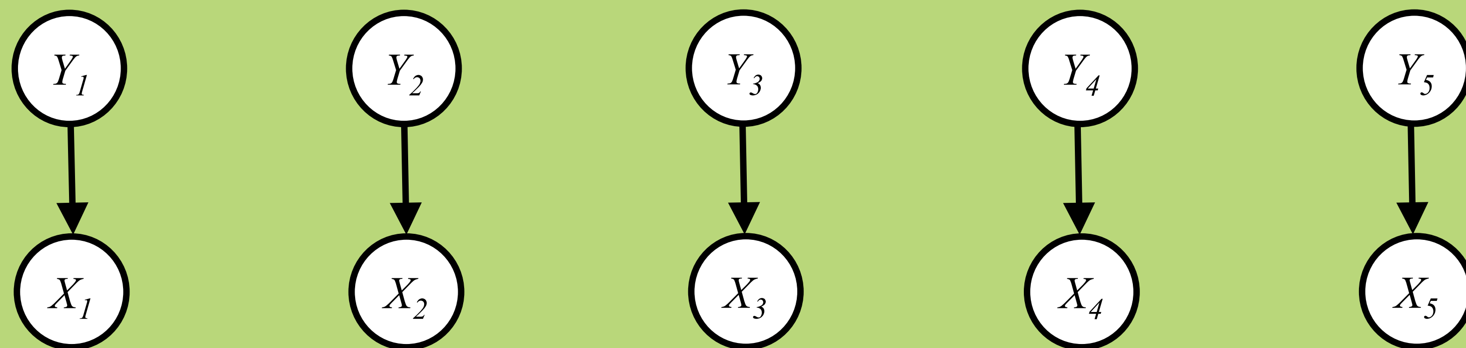


"Naïve Bayes": $$P(\mathbf{X}, \mathbf{Y}) = \prod_{t=1}^{T} P(X_t|Y_t)p(Y_t)$$

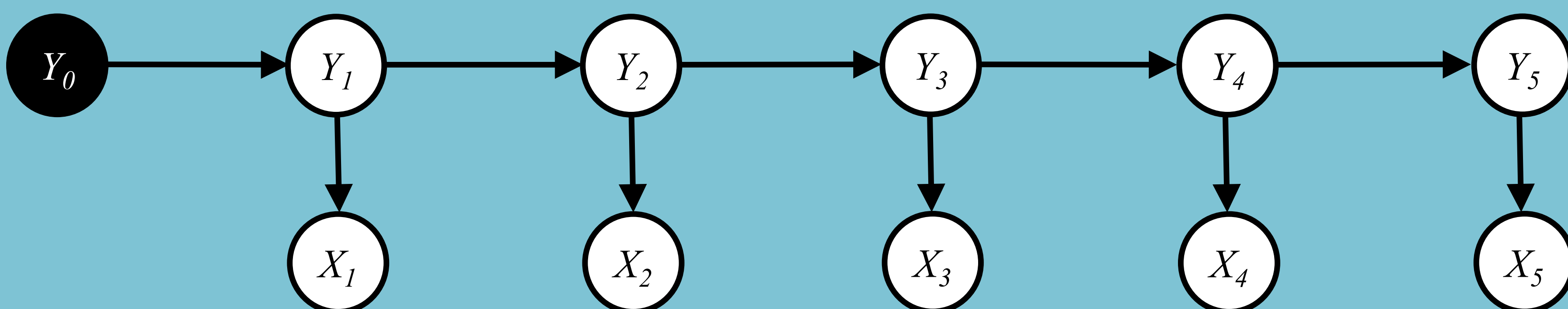


HMM: $$P(\mathbf{X}, \mathbf{Y}) = P(Y_1)\left(\prod_{t=1}^{T} P(X_t|Y_t)\right)\left(\prod_{t=2}^{T} p(Y_t|Y_{t-1})\right)$$

From Mixture Model to HMM



"Naïve Bayes": $$P(\mathbf{X},\mathbf{Y}) = \prod_{t=1}^{T} P(X_t|Y_t)p(Y_t)$$



HMM: $$P(\mathbf{X},\mathbf{Y}|Y_0) = \prod_{t=1}^{T} P(X_t|Y_t)p(Y_t|Y_{t-1})$$

SUPERVISED LEARNING FOR HMMS

Hidden Markov Model

HMM Parameters:

Emission matrix, $\mathbf{A}$, where $P(X_t = k | Y_t = j) = A_{j,k}, \forall t, k$

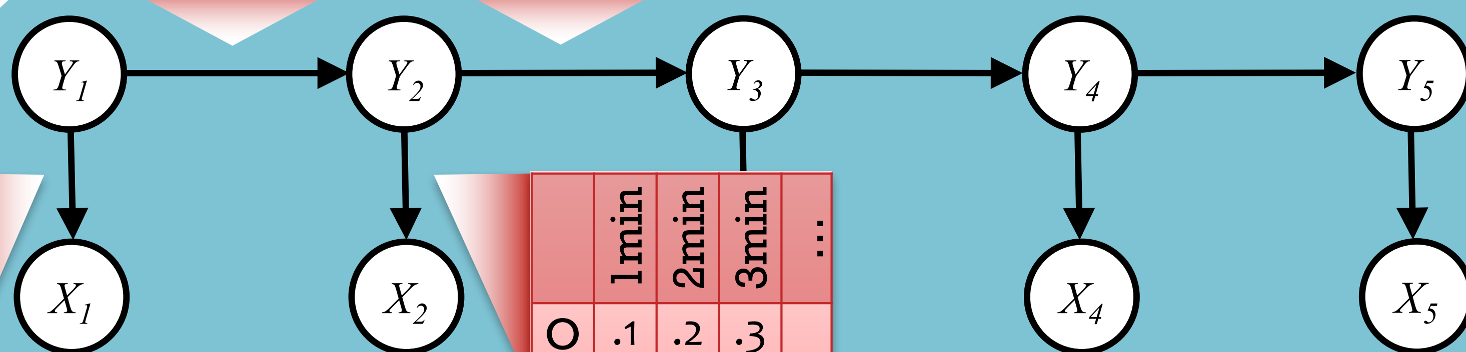
Transition matrix, $\mathbf{B}$, where $P(Y_t = k | Y_{t-1} = j) = B_{j,k}, \forall t, k$

Initial probs, $\mathbf{C}$, where $P(Y_1 = k) = C_k, \forall k$

O	.8
S	.1
C	.1

	O	S	C
O	.9	.08	.02
S	.2	.7	.1
C	.9	0	.1

	O	S	C
O	.9	.08	.02
S	.2	.7	.1
C	.9	0	.1



	1min	2min	3min	…
O	.1	.2	.3	
S	.01	.02	.03	
C	0	0	0	

	1min	2min	3min	…
O	.1	.2	.3	
S	.01	.02	.03	
C	0	0	0	

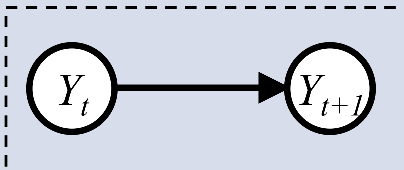
Training HMMs

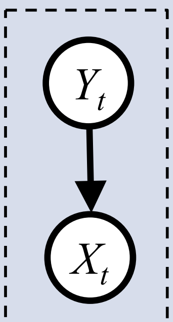
Whiteboard

- (Supervised) Likelihood for an HMM
- Maximum Likelihood Estimation (MLE) for HMM

Supervised Learning for HMMs

Learning an HMM decomposes into solving two (independent) Mixture Models





Data: $\mathcal{D} = \{(\vec{x}^{(i)}, \vec{y}^{(i)})\}_{i=1}^{N}$ $\quad \vec{x} = [x_1, \dots, x_T]^T$, $\vec{y} = [y_1, \dots, y_T]^T$

Likelihood:

$$\ell(A,B,C) = \sum_{i=1}^{N} \log p(\vec{x}^{(i)}, y^{(i)} | A,B,C)$$

$$= \sum_{i=1}^{N} \Big[\underbrace{\log p(y_1^{(i)}|C)}_{\text{initial}} + \underbrace{\left(\sum_{t=2}^{T} \log p(y_t^{(i)}|y_{t-1}^{(i)}, B)\right)}_{\text{transition}} + \underbrace{\left(\sum_{t=1}^{T} \log p(x_t^{(i)}|y_t^{(i)}, A)\right)}_{\text{emission}} \Big]$$

MLE:

$$\hat{A}, \hat{B}, \hat{C} = \underset{A,B,C}{\operatorname{argmax}}\ \ell(A,B,C)$$

$$\Rightarrow \hat{C} = \underset{C}{\operatorname{argmax}} \sum_{i=1}^{N} \log p(y_1^{(i)}|C)$$

$$\hat{B} = \underset{B}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{t=2}^{T} \log p(y_t^{(i)}|y_{t-1}^{(i)}, B)$$

$$\hat{A} = \underset{A}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{t=1}^{T} \log p(x_t^{(i)}|y_t^{(i)}, A)$$

Can solve in closed form, which yields...

$$\hat{C}_k = \frac{\#(y_1^{(i)} = k)}{N} \quad \forall i,k$$

$$\hat{B}_{jk} = \frac{\#(y_t^{(i)} = k \text{ and } y_{t-1}^{(i)} = j)}{\#(y_{t-1}^{(i)} = j)} \quad \forall i, t>1, j, k$$

$$\hat{A}_{jk} = \frac{\#(x_t^{(i)} = k \text{ and } y_t^{(i)} = j)}{\#(y_t^{(i)} = j)} \quad \forall i, t, j, k$$

Hidden Markov Model

HMM Parameters:

Emission matrix, $\mathbf{A}$, where $P(X_t = k | Y_t = j) = A_{j,k}, \forall t, k$

Transition matrix, $\mathbf{B}$, where $P(Y_t = k | Y_{t-1} = j) = B_{j,k}, \forall t, k$

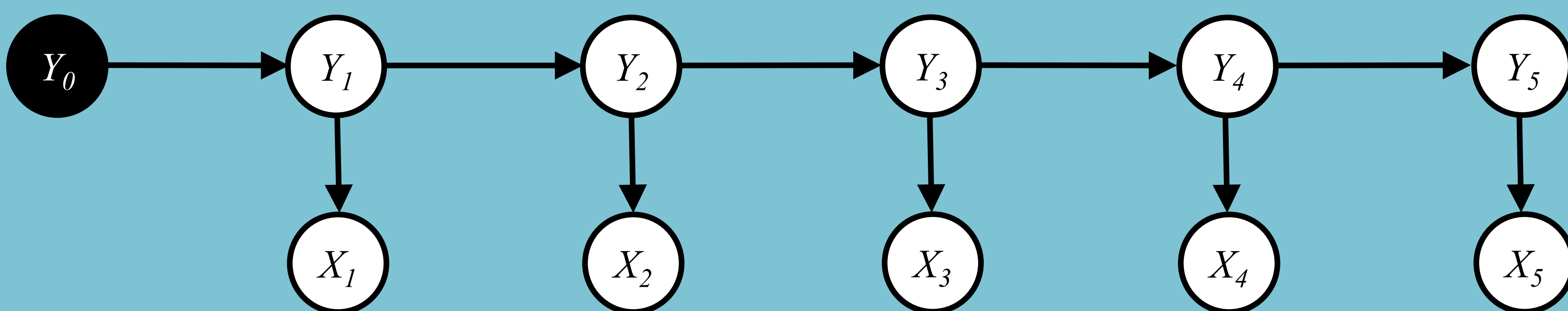
Assumption: $y_0 = \text{START}$

Generative Story:

$$Y_t \sim \text{Multinomial}(\mathbf{B}_{Y_{t-1}}) \;\forall t$$

$$X_t \sim \text{Multinomial}(\mathbf{A}_{Y_t}) \;\forall t$$

For notational convenience, we fold the *initial probabilities* **C** into the *transition matrix* **B** by our assumption.



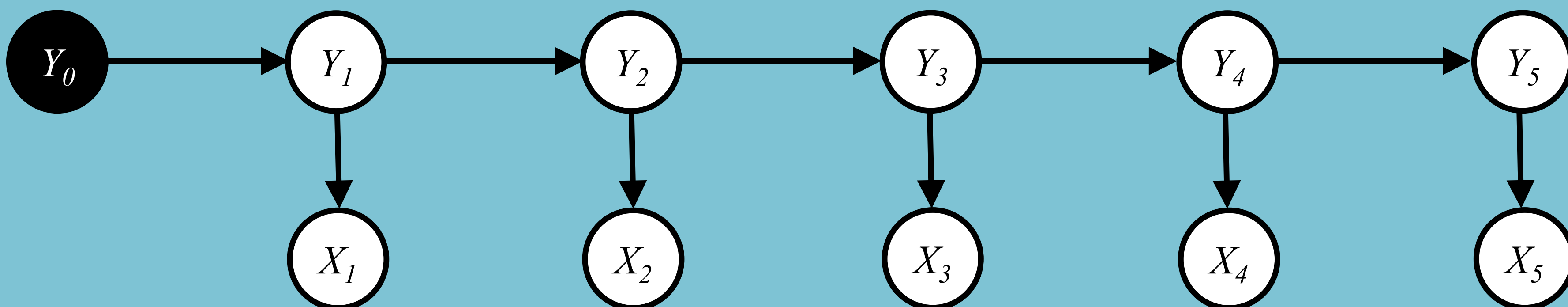
Hidden Markov Model

Joint Distribution:

$$y_0 = \text{START}$$

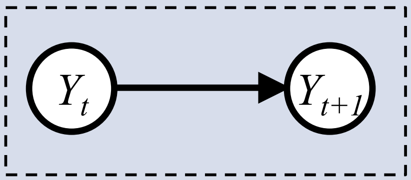
$$p(\mathbf{x}, \mathbf{y} | y_0) = \prod_{t=1}^{T} p(x_t | y_t) p(y_t | y_{t-1})$$

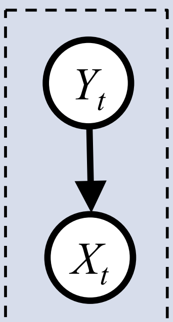
$$= \prod_{t=1}^{T} A_{y_t, x_t} B_{y_{t-1}, y_t}$$



Supervised Learning for HMMs

Learning an HMM decomposes into solving two (independent) Mixture Models





$$\mathcal{D} = \{(\vec{x}^{(i)}, \vec{y}^{(i)})\}_{i=1}^{N}$$

Likelihood:

$$\ell(A,B) = \sum_{i=1}^{N} \log p(\vec{x}^{(i)}, \vec{y}^{(i)})$$

$$= \sum_{i=1}^{N} \left[\sum_{t=1}^{T} \log p(y_t^{(i)} \mid y_{t-1}^{(i)}, B) + \log p(x_t^{(i)} \mid y_t^{(i)}, A) \right]$$

MLE:

$$\hat{A}, \hat{B} = \operatorname{argmax} \ell(A,B)$$

$$\hat{A} = \operatorname{argmax} \sum_{i=1}^{N} \left[\sum_{t=1}^{T} \log p(x_t^{(i)} \mid y_t^{(i)}, A) \right]$$

$$\hat{B} = \operatorname{argmax} \sum_{i=1}^{N} \left[\sum_{t=1}^{T} \log p(y_t^{(i)} \mid y_{t-1}^{(i)}, B) \right]$$

can solve in closed form to get...

$$\hat{B}_{jk} = \frac{\#(y_t^{(i)} = k \text{ and } y_{t-1}^{(i)} = j)}{\#(y_{t-1}^{(i)} = j)}$$

$$\hat{A}_{jk} = \frac{\#(x_t^{(i)} = k \text{ and } y_t^{(i)} = j)}{\#(y_t^{(i)} = j)}$$

HMMs: History

- Markov chains: Andrey Markov (1906)
  - Random walks and Brownian motion
- Used in Shannon’s work on information theory (1948)
- Baum-Welsh learning algorithm: late 60’s, early 70’s.
  - Used mainly for speech in 60s-70s.
- Late 80’s and 90’s: David Haussler (major player in learning theory in 80’s) began to use HMMs for modeling biological sequences
- Mid-late 1990’s: Dayne Freitag/Andrew McCallum
  - Freitag thesis with Tom Mitchell on IE from Web using logic programs, grammar induction, etc.
  - McCallum: multinomial Naïve Bayes for text
  - With McCallum, IE using HMMs on CORA
- …

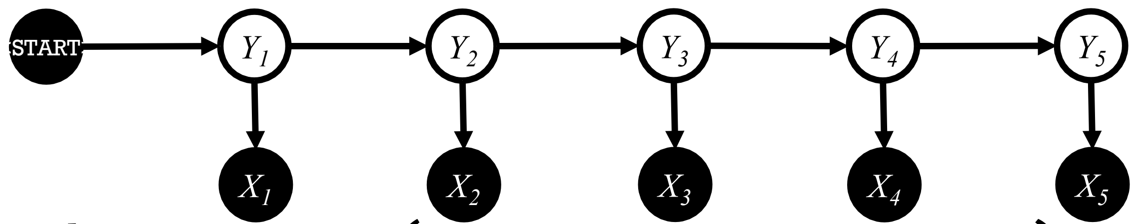




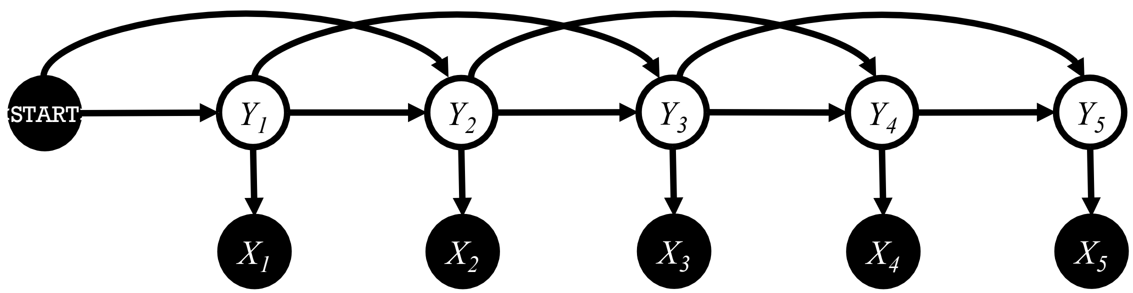
Slide from William Cohen

Higher-order HMMs

- 1st-order HMM (i.e. bigram HMM)



- 2nd-order HMM (i.e. trigram HMM)



- 3rd-order HMM