Predicting Sequences: Hidden Markov Models

Slides adopted from Srikumar`s CS 6355: Structured Prediction

Outline

- Sequence models
- Hidden Markov models
  - Inference with HMM
  - Learning
- Conditional Models and Local Classifiers
- Global models
  - Conditional Random Fields
  - Structured Perceptron for sequences

Outline

- *Sequence models*
- Hidden Markov models
  - Inference with HMM
  - Learning
- Conditional Models and Local Classifiers
- Global models
  - Conditional Random Fields
  - Structured Perceptron for sequences

Sequences

- Sequences of states
  - Text is a sequence of words or even letters
  - A video is a sequence of frames

- Even with a finite set of states, the set of unique state *sequences* is infinite

- Our goal (for now): Define probability distributions over sequences

- If $x_1, x_2, \cdots, x_n$ is a sequence that has n tokens, we want to be able to define $P(x_1, x_2, \cdots, x_n)$

  …for all values of n

A history-based model

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i \mid x_1, x_2, \cdots, x_{i-1})$$

Each token is dependent on every token that came before it

- Simple conditioning
- Each $\mathrm{P}(x_i \mid x_1, x_2, \cdots, x_{i-1})$ is a multinomial probability distribution over the tokens

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$$P(\text{It})\times$$ ← Probability of a word starting a sentence

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$$P(\text{It})\times$$ ← Probability of a word starting a sentence

$$P(\text{was}|\text{It})\times$$ ← Probability of a word following "It"

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$P(\text{It})\times$ ← Probability of a word starting a sentence

$P(\text{was}|\text{It})\times$ ← Probability of a word following "It"

$P(\text{a}|\text{It was})\times$ ← Probability of a word following "It was"

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$P(\text{It})\times$ ← Probability of a word starting a sentence

$P(\text{was}|\text{It})\times$ ← Probability of a word following "It"

$P(\text{a}|\text{It was})\times$ ← Probability of a word following "It was"

$P(\text{bright}|\text{It was a})\times$ ← Probability of a word following "It was a"

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$$P(\text{It})\times$$ ← Probability of a word starting a sentence

$$P(\text{was}|\text{It})\times$$ ← Probability of a word following "It"

$$P(\text{a}|\text{It was})\times$$ ← Probability of a word following "It was"

$$P(\text{bright}|\text{It was a})\times$$ ← Probability of a word following "It was a"

$$P(\text{cold}|\text{It was a bright})\times$$

$$P(\text{day}|\text{It was a bright cold}) \times \cdots$$

What's the problem with this strategy?

A history-based model

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i \mid x_1, x_2, \cdots, x_{i-1})$$

Each token is dependent on every token that came before it

- Simple conditioning
- Each $\mathrm{P}(x_i \mid x_1, x_2, \cdots, x_{i-1})$ is a multinomial probability distribution over the tokens

What's the problem with this strategy?

- How many parameters do we have?
  - Grows with the size of the sequence!

Solution: Lose the history

Make a modeling assumption: *The first-order Markov assumption*

The state of the system at any time is ***independent*** of the full sequence history *given the previous state*

$$P\left(x_i \mid x_1, x_2, \cdots, x_{i-1}\right) = P(x_i \mid x_{i-1})$$

Solution: Lose the history

Make a modeling assumption: *The first-order Markov assumption*

The state of the system at any time is ***independent*** of the full sequence history *given the previous state*

$$P\left(x_i \mid x_1, x_2, \cdots, x_{i-1}\right) = P(x_i \mid x_{i-1})$$

This allows us to simplify

$$P(x_1, x_2, x_3, \cdots, x_n) = \prod_i P(x_i \mid x_1, x_2 \cdots, x_{i-1})$$

These dependencies are ignored

Solution: Lose the history

Make a modeling assumption: *The first-order Markov assumption*

The state of the system at any time is ***independent*** of the full sequence history *given the previous state*

$$P\left(x_i \mid x_1, x_2, \cdots, x_{i-1}\right) = P(x_i \mid x_{i-1})$$

This allows us to simplify

$$P(x_1, x_2, x_3, \cdots, x_n) = \prod_i P(x_i \mid x_{i-1})$$

First-order Markov models

Defined by two sets of probabilities

1. The initial state distribution: The probability that a sequence starts at a certain state j: $P(x_1 = \text{state}_j)$

2. The state transition distribution: The probability that the system will transition to a state k at some step if it was at a state j at the previous step: $P(x_{t+1}\text{state}_k \mid x_t = \text{state}_j)$

Example: Another language model

It was a bright cold day in April

$$P(\text{It was a bright cold day in April}) =$$

$P(\text{It})\times$ ← Probability of a word starting a sentence

$P(\text{was}|\text{It})\times$ ← Probability of a word following "It"

$P(\text{a}|\text{was})\times$ ← Probability of a word following "was"

$P(\text{bright}|\text{a})\times$ ← Probability of a word following "a"

$P(\text{cold}|\text{bright})\times$

$P(\text{day}|\text{cold}) \times \cdots$

Example: Another language model

It was a bright cold day in April

$$P(\text{It was a bright cold day in April}) =$$

$P(\text{It})\times$ ← Probability of a word starting a sentence

$P(\text{was}|\text{It})\times$ ← Probability of a word following “It”

$P(\text{a}|\text{was})\times$ ← Probability of a word following “was”

$P(\text{bright}|\text{a})\times$ ← Probability of a word following “a”

$P(\text{cold}|\text{bright})\times$

$P(\text{day}|\text{cold}) \times \cdots$

If there are K tokens/states, how many parameters do we need?

Example: Another language model

It was a bright cold day in April

$$P(\text{It was a bright cold day in April}) =$$

$$P(\text{It})\times$$ ← Probability of a word starting a sentence

$$P(\text{was}|\text{It})\times$$ ← Probability of a word following "It"

$$P(\text{a}|\text{was})\times$$ ← Probability of a word following "was"

$$P(\text{bright}|\text{a})\times$$ ← Probability of a word following "a"

$$P(\text{cold}|\text{bright})\times$$

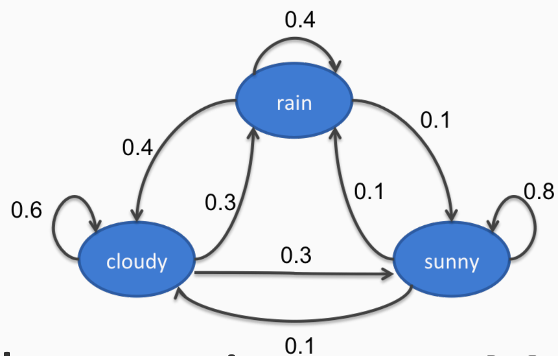
$$P(\text{day}|\text{cold}) \times \cdots$$

If there are K tokens/states, how many parameters do we need? $O(K^2)$

Example: The weather

Three states: rain, cloudy, sunny

State transitions:



Suppose the observations are Markov chains:

Eg: *cloudy sunny sunny rain*

- Probability of the sequence =

  P(cloudy) P(sunny|cloudy) P(sunny | sunny) P(rain | sunny)

Initial probability

Transition probabilities

These probabilities define the model; can find P(any sequence)

m^{th} order Markov Model

A generalization of the first order Markov Model

- Each state is only dependent on m previous states

- More parameters
- But still less than storing entire history

m^{th} order Markov Model

A generalization of the first order Markov Model

- Each state is only dependent on m previous states

- More parameters
- But still less than storing entire history

Questions?

Outline

- Sequence models
- *Hidden Markov models*
  - Inference with HMM
  - Learning
- Conditional Models and Local Classifiers
- Global models
  - Conditional Random Fields
  - Structured Perceptron for sequences

Hidden Markov Model

- Discrete Markov Model:
  - States follow a Markov chain
  - *Each state is an observation*

- Hidden Markov Model:
  - States follow a Markov chain
  - States are not observed
  - Each state stochastically emits an observation

Example: Part of speech tagging

Given a sentence, find parts of speech of all the words

The Fed raises interest rates

Example: Part of speech tagging

Given a sentence, find parts of speech of all the words

The	Fed	raises	interest	rates
Determiner	Noun	Verb	Noun	Noun

Example: Part of speech tagging

Given a sentence, find parts of speech of all the words

	The	Fed	raises	interest	rates
	Determiner	Noun	Verb	Noun	Noun
Other possible tags in different contexts		Verb	Noun	Verb	Verb
		(I *fed* the dog)	(Annual *raises*)	(Poems *interest* me)	(He *rates* movies online)

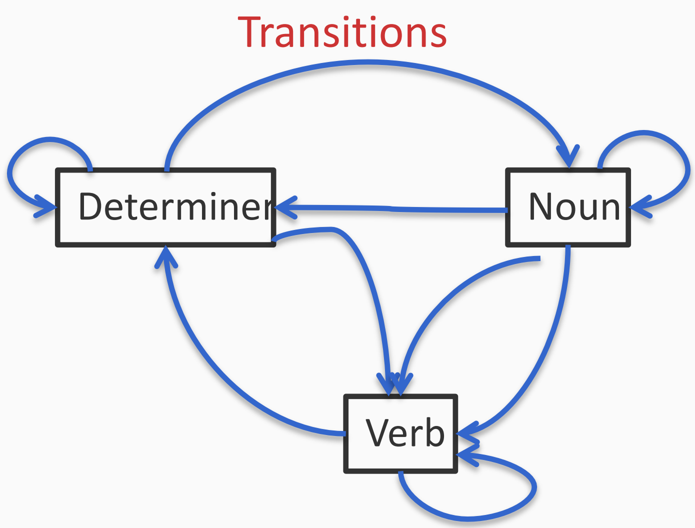
Example: Part of speech tagging

Given a sentence, find parts of speech of all the words

	The	Fed	raises	interest	rates
	Determiner	Noun	Verb	Noun	Noun
Other possible tags in different contexts		Verb	Noun	Verb	Verb
		(I *fed* the dog)	(Annual *raises*)	(Poems *interest* me)	(He *rates* movies online)

If these were the only options allowed, we will have $1 \times 2 \times 2 \times 2 \times 2 = 16$ possible output sequences

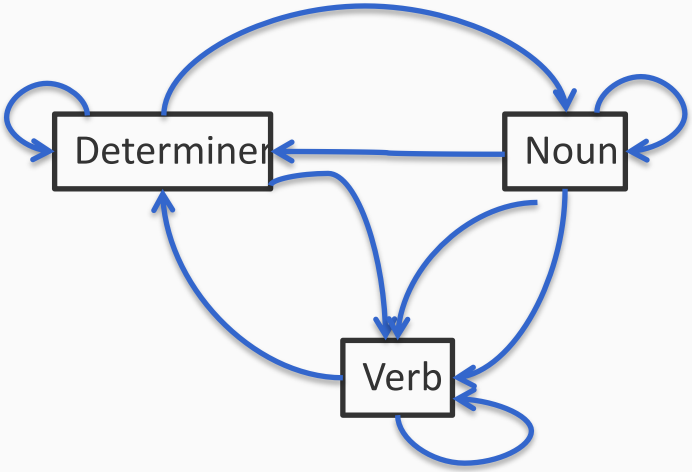
Toy part of speech example



*Each edge here is associated with a **transition probability***

Toy part of speech example

Transitions



*Each edge here is associated with a **transition probability***

Emissions

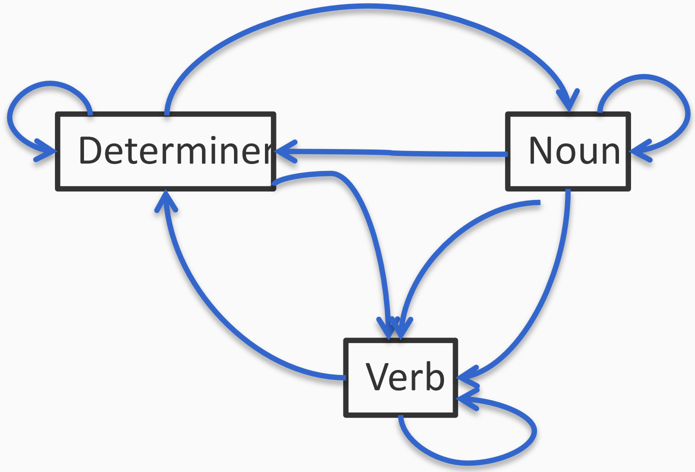
P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…

***Emission probabilities:** Given that the system is in a certain state, these are probabilities that it will emit a certain observation*

Toy part of speech example

Transitions



*Each edge here is associated with a **transition probability***

Emissions

P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…

***Emission probabilities:** Given that the system is in a certain state, these are probabilities that it will emit a certain observation*

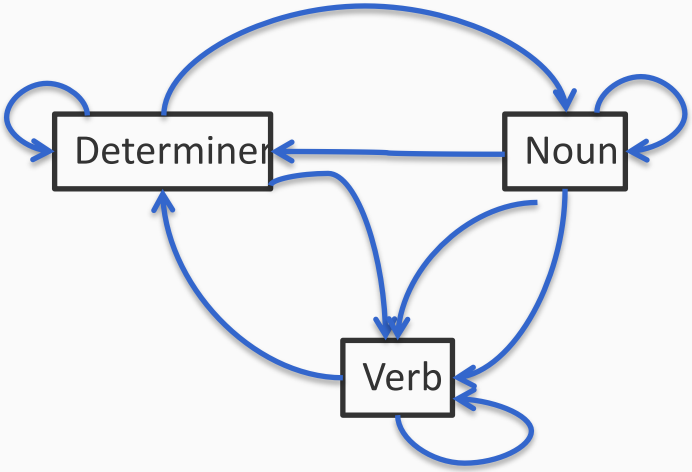
Initial

P(Determiner) = 0.9
P(Noun) = 0.08
P(Verb) = 0.02

***Initial probabilities**: What is the probability that the sequence starts in a certain state?*

Toy part of speech example

Transitions



start

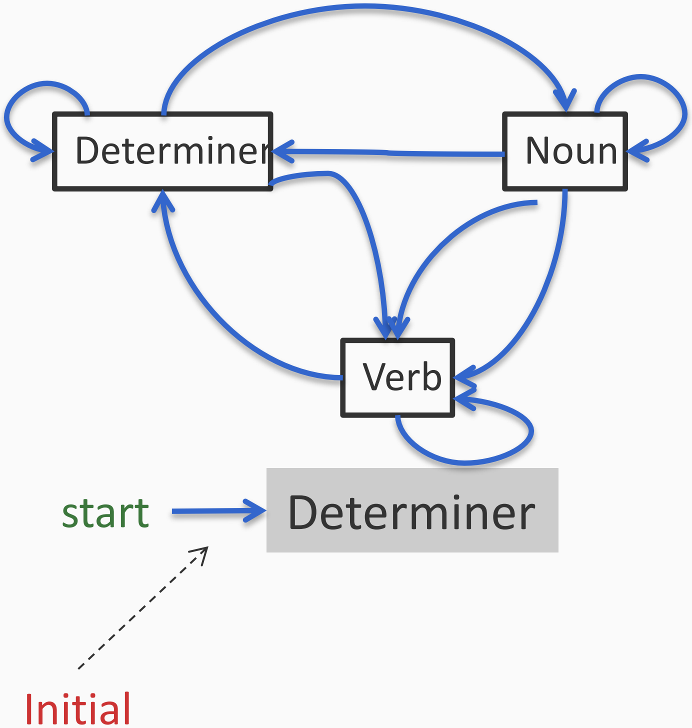
Emissions

P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…

Toy part of speech example

Transitions



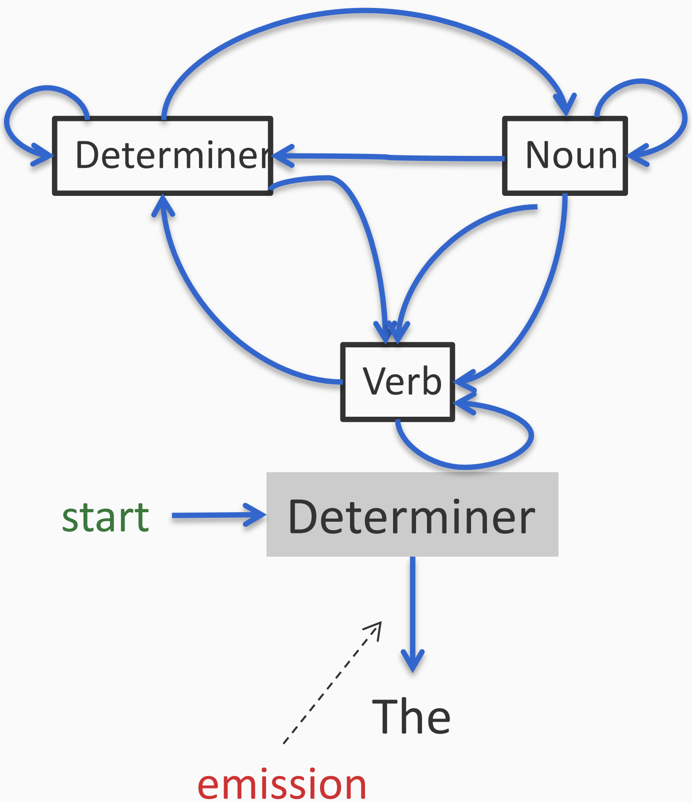
Emissions

P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…

Toy part of speech example

Transitions



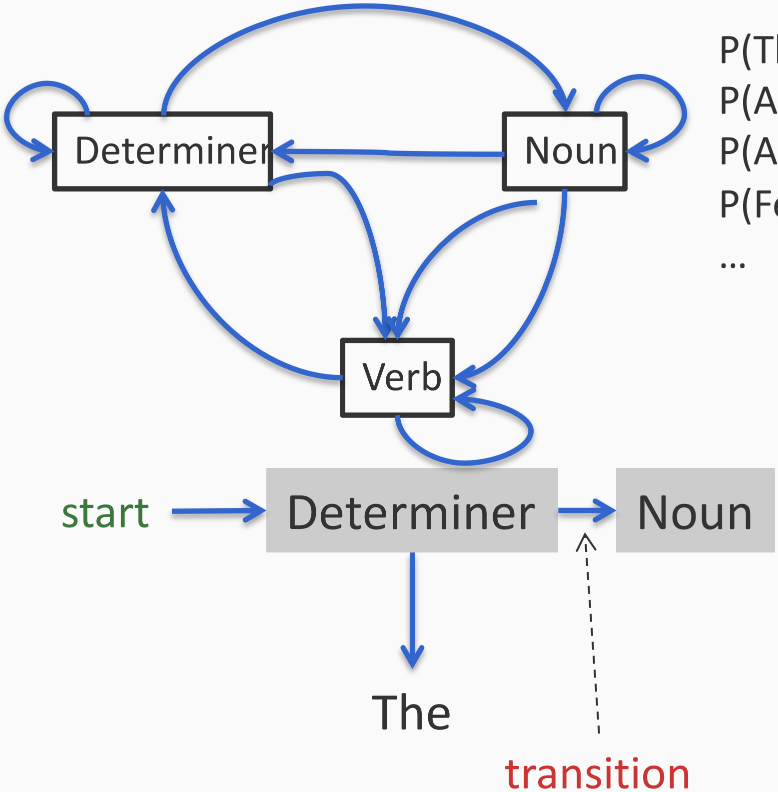
Emissions

P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…

Toy part of speech example

Transitions



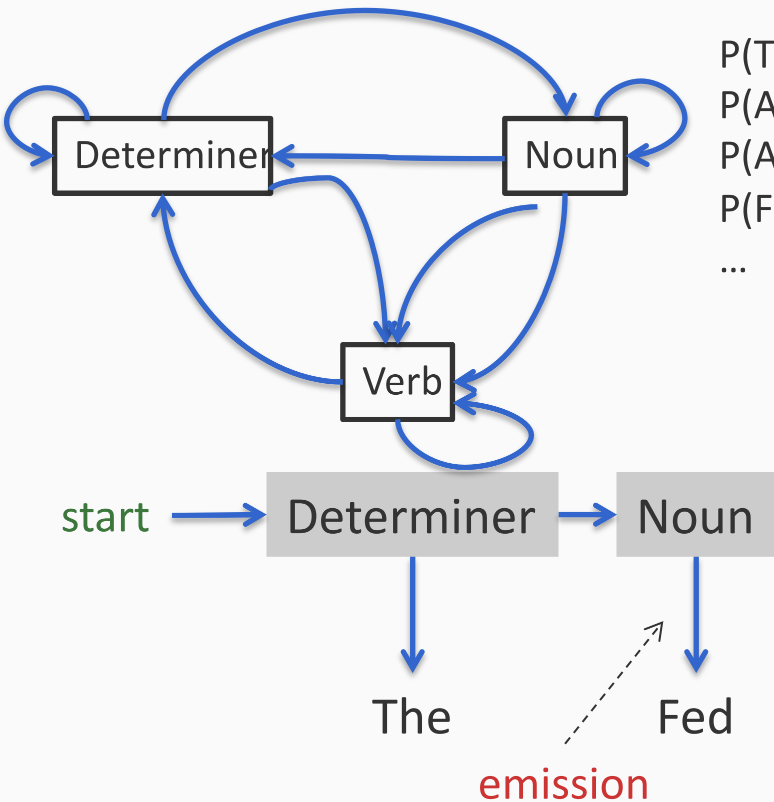
Emissions

P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…

Toy part of speech example

Transitions



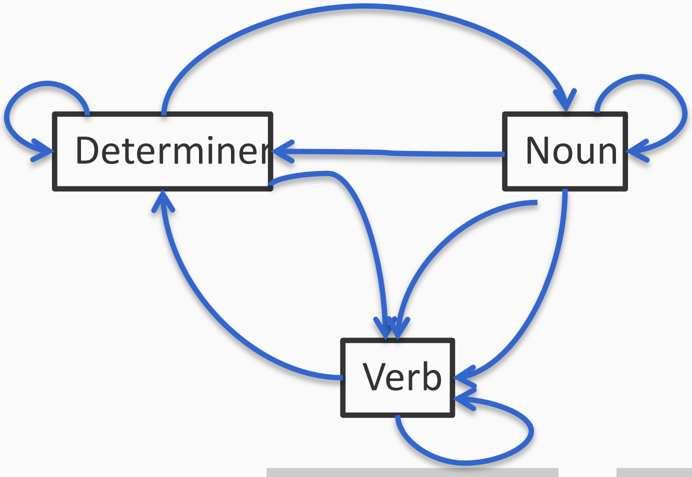
Emissions

P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…

Toy part of speech example

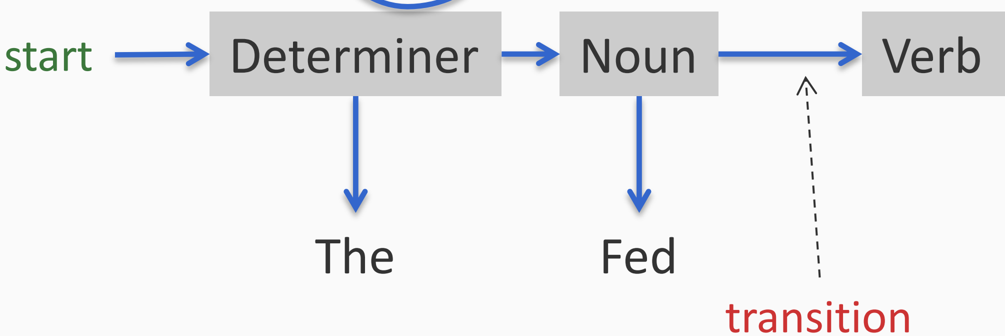
Transitions



Emissions

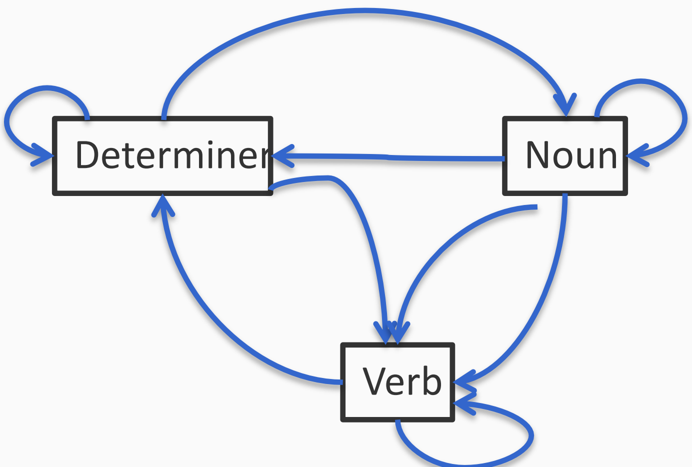
P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…



Toy part of speech example

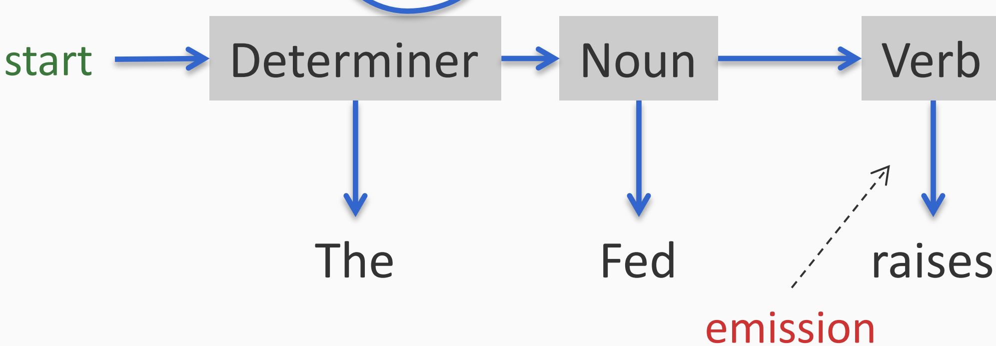
Transitions



Emissions

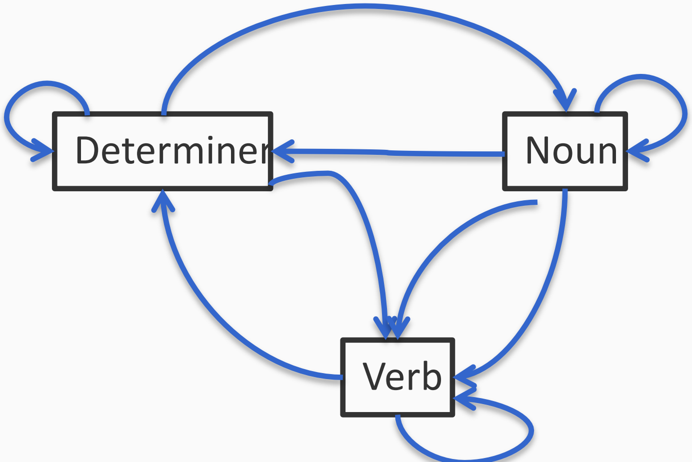
P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…



Toy part of speech example

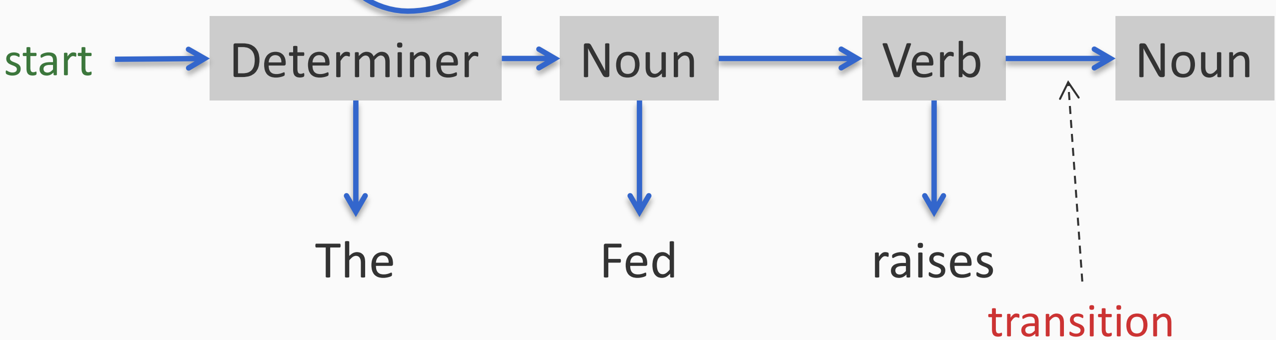
Transitions



Emissions

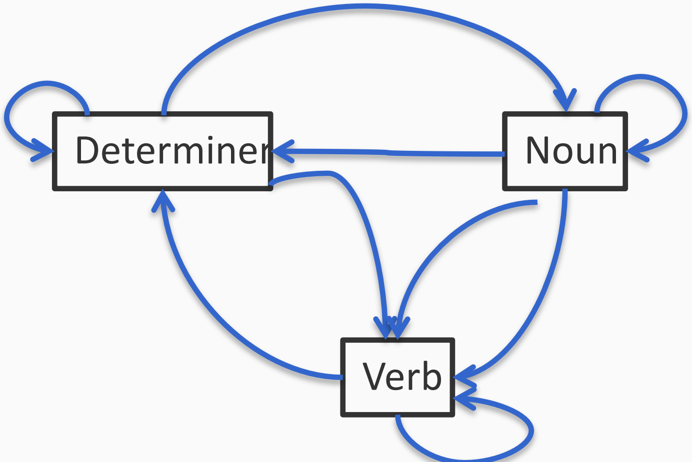
P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…



Toy part of speech example

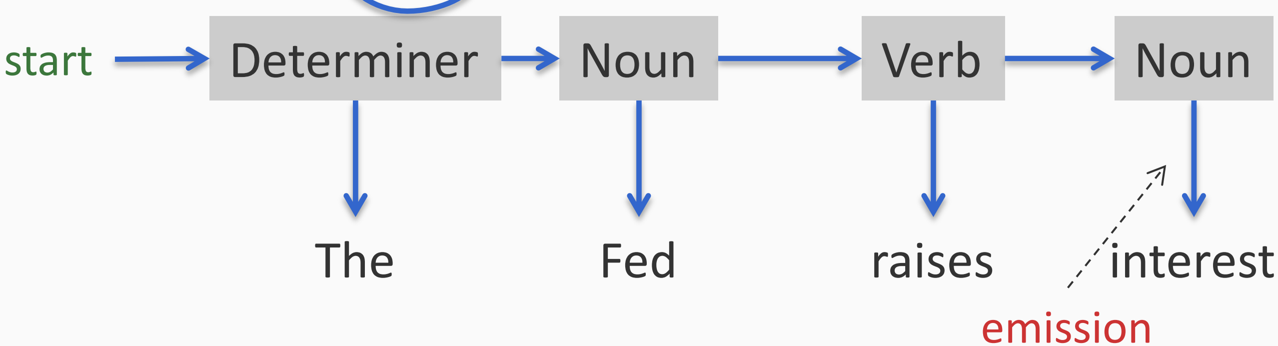
Transitions



Emissions

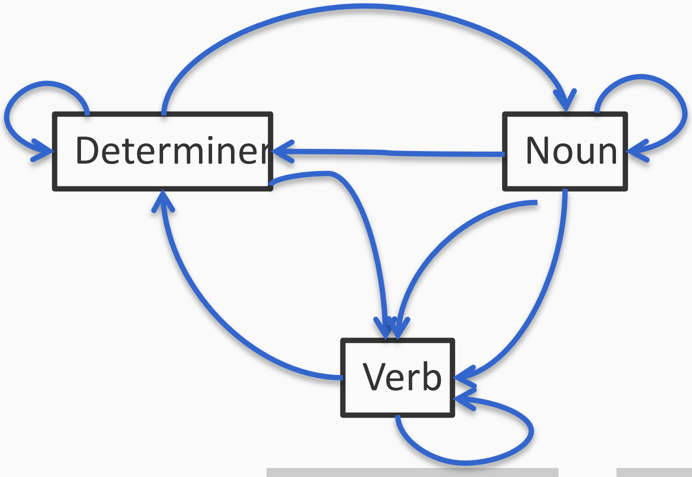
P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…



Toy part of speech example

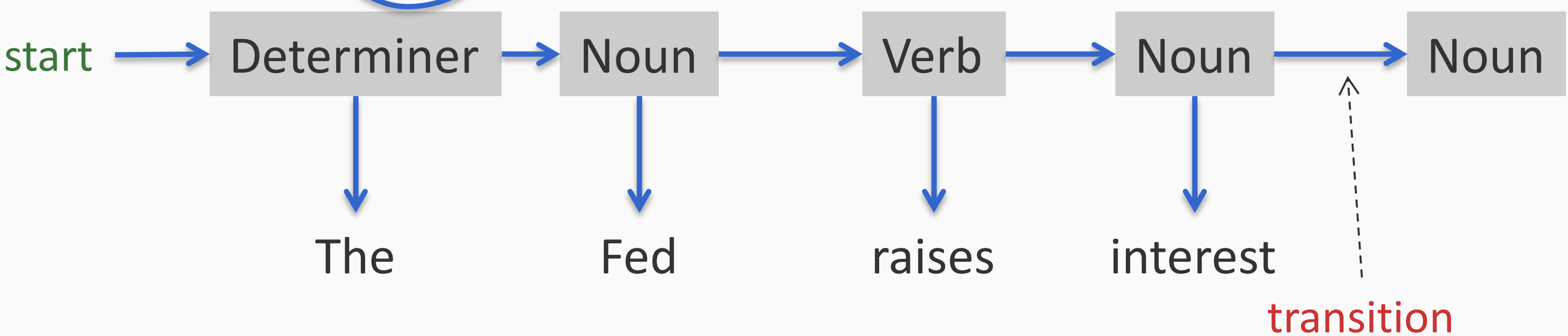
Transitions



Emissions

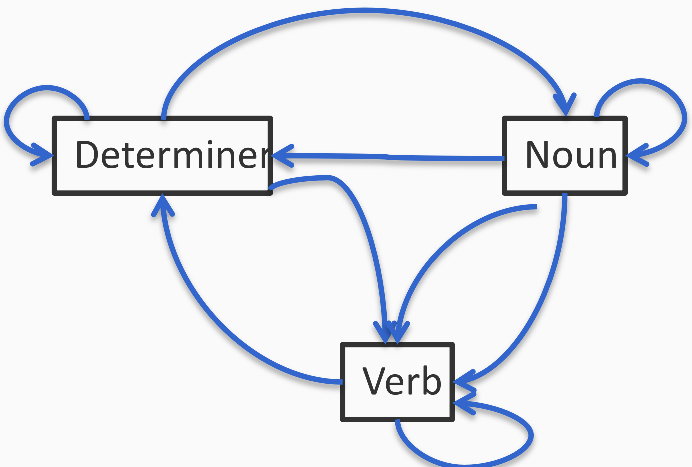
P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…



Toy part of speech example

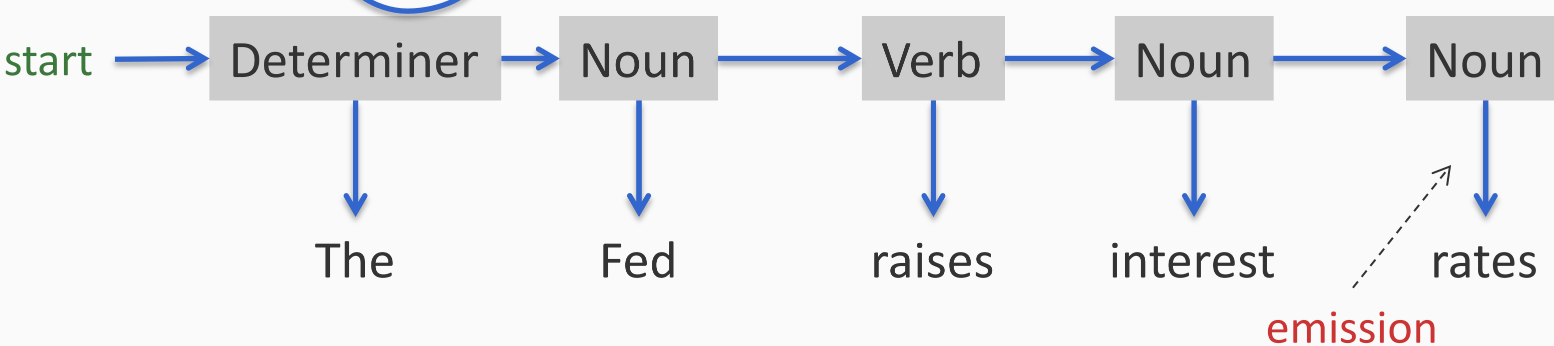
Transitions



Emissions

P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…



Joint model over states and observations

- Notation
  - Number of states = K
  - Number of possible observations for any state = M
  - π: Initial probability over states ($K-1$ numbers)
  - A: Transition probabilities ($K \times K$ matrix)
  - B: Emission probabilities ($K \times M$ matrix)

Joint model over states and observations

- Notation
  - Number of states = K
  - Number of possible observations for any state = M
  - π: Initial probability over states ($K-1$ numbers)
  - A: Transition probabilities ($K \times K$ matrix)
  - B: Emission probabilities ($K \times M$ matrix)

- Probability of states and observations
  - Denote states by $y_1, y_2, \cdots$ and observations by $x_1, x_2, \cdots$

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$= \pi_{y_1} \prod_{i=1}^{n-1} A_{y_i, y_{i+1}} \prod_{i=1}^{n} B_{y_i, x_i}$$

Example: Named Entity Recognition

Goal: To identify persons, locations and organizations in text

Observations

Facebook CEO Mark Zuckerberg announced new

privacy features in the conference in San Francisco

Example: Named Entity Recognition

Goal: To identify persons, locations and organizations in text

States

Observations

B-org O B-per I-per O O

Facebook CEO Mark Zuckerberg announced new

O O O O O O B-loc I-loc

privacy features in the conference in San Francisco

Numerous other applications

- Speech recognition
  - Input: Speech signal
  - Output: Sequence of words
- NLP applications
  - Information extraction
  - Text chunking
- Computational biology
  - Aligning protein sequences
  - Labeling nucleotides in a sequence as exons, introns, etc.

Questions?

Three questions for HMMs

[Rabiner 1999]

1. Given an observation sequence $x_1, x_2, \cdots, x_n$ and a model (π, A, B), how to efficiently calculate the probability of the observation?

2. Given an observation sequence $x_1, x_2, \cdots, x_n$ and a model (π, A, B), how to efficiently calculate the most probable state sequence?

3. How to calculate (π, A, B) from observations?

Three questions for HMMs

[Rabiner 1999]

1. Given an observation sequence $x_1, x_2, \cdots, x_n$ and a model (π, A, B), how to efficiently calculate the probability of the observation?

2. Given an observation sequence $x_1, x_2, \cdots, x_n$ and a model (π, A, B), how to efficiently calculate the most probable state sequence?

3. How to calculate (π, A, B) from observations?

Outline

- Sequence models
- Hidden Markov models
  - *Inference with HMM*
  - Learning
- Conditional Models and Local Classifiers
- Global models
  - Conditional Random Fields
  - Structured Perceptron for sequences

Most likely state sequence

- Input:
  - A hidden Markov model (π, A, B)
  - An observation sequence $\mathbf{x} = (x_1, x_2, \cdots, x_n)$
- Output: A state sequence $\mathbf{y} = (y_1, y_2, \cdots, y_n)$ that corresponds to $\underset{y}{\operatorname{argmax}} P(\mathbf{y} \mid \mathbf{x}, \pi, A, B)$
  - Maximum *a posteriori* inference (MAP inference)
- Computationally: combinatorial optimization

MAP inference

- We want to find $\underset{y}{\operatorname{argmax}} P(\mathbf{y} \mid \mathbf{x}, \pi, A, B)$

- We have defined

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots, y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1} \mid y_i) \prod_{i=1}^{n} P(x_i \mid y_i)$$

- But, $P(\mathbf{y} \mid \mathbf{x}, \pi, A, B) \propto P(\mathbf{x}, \mathbf{y} \mid \pi, A, B)$
  - And we don't care about $P(\mathbf{x})$ we are maximizing over $\mathbf{y}$

- That is

$$\underset{y}{\operatorname{argmax}} P(\mathbf{y} \mid \mathbf{x}, \pi, A, B) = \underset{y}{\operatorname{argmax}} P(\mathbf{x}, \mathbf{y} \mid \pi, A, B)$$

How many possible sequences?

The	Fed	raises	interest	rates

Suppose each word allows only the following tags

Determiner	Verb	Verb	Verb	Verb
	Noun	Noun	Noun	Noun
1	2	2	2	2

In this simple case, $1 \times 2 \times 2 \times 2 \times 2 = 16$ possible sequences exist

How many possible sequences?

Observations	x_1	x_2	…	x_n

Suppose each observation allows any of the following k states

s_1	s_1	…	s_1
s_2	s_2		s_2
s_3	s_2		s_3
.	.		.
.	.		.
s_K	s_K		s_K

Output: One state per observation $y_i = s_j$

K^n possible sequences to consider for $\underset{\mathbf{y}}{\operatorname{argmax}}\, P(\mathbf{y} \mid \mathbf{x}, \pi, A, B)$

Naïve approaches

1. Try out every sequence
   - Score the sequence $\mathbf{y}$ as $P(\mathbf{y} \mid \mathbf{x}, \pi, A, B)$
   - Return the highest scoring one
   - Correct, but slow, $O(K^n)$

2. Greedy search
   - Construct the output left to right
   - For each i, elect the best y_i using y_{i-1} and x_i
   - Incorrect but fast, $O(n)$

Solution: Use the independence assumptions

Take advantage of the first order Markov assumption

The state for any observation is only influenced by the previous state, the next state and the observation itself

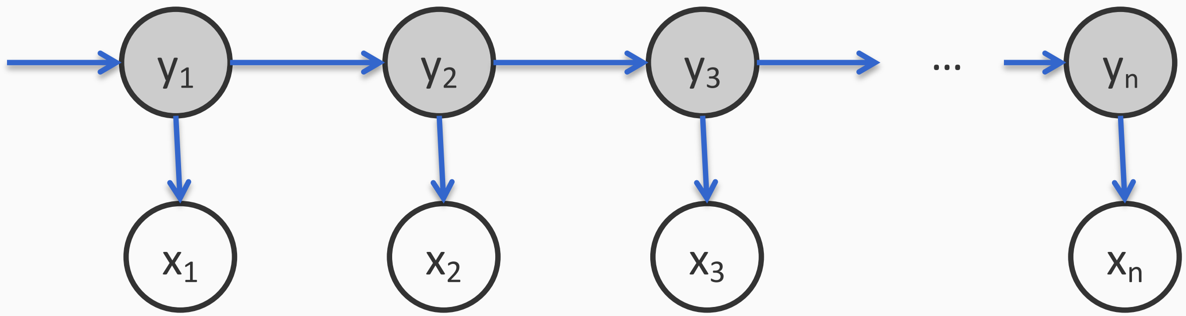
Given the adjacent labels, the others do not matter

Suggests a recursive algorithm

Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

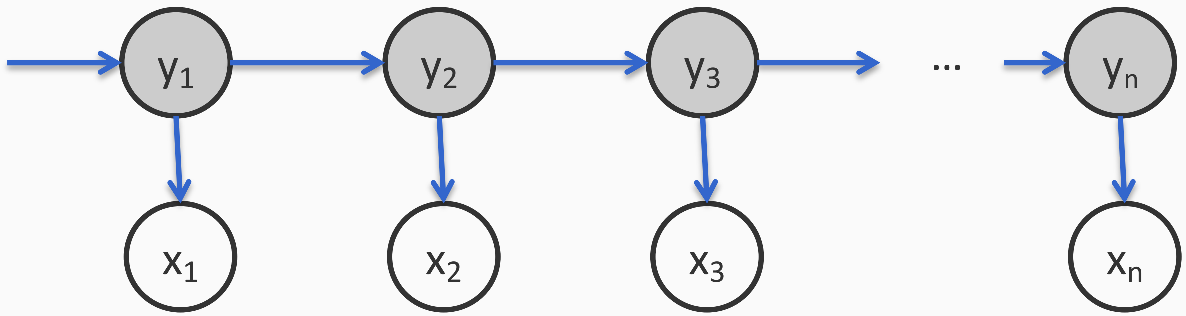
What we want: An assignment to all the y_i's that maximizes this product



Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

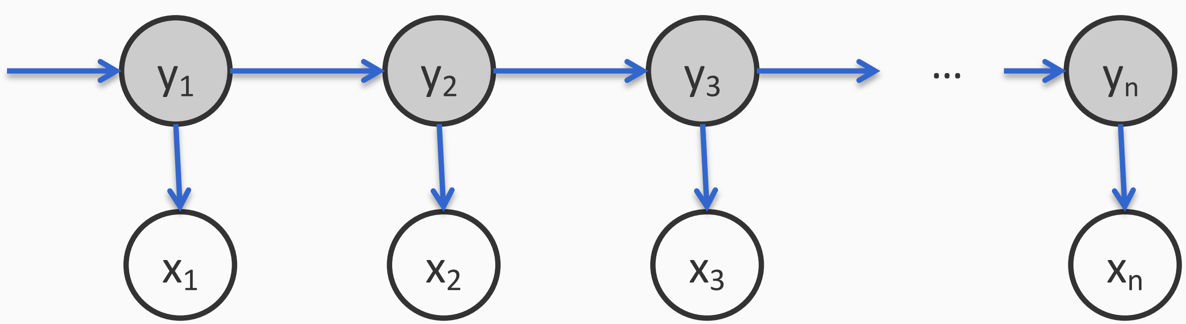


Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

Initial probability



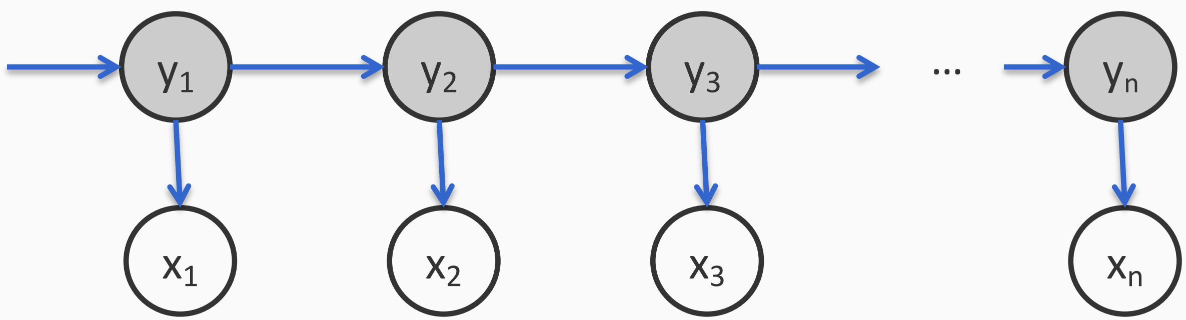
Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

Emission probabilities

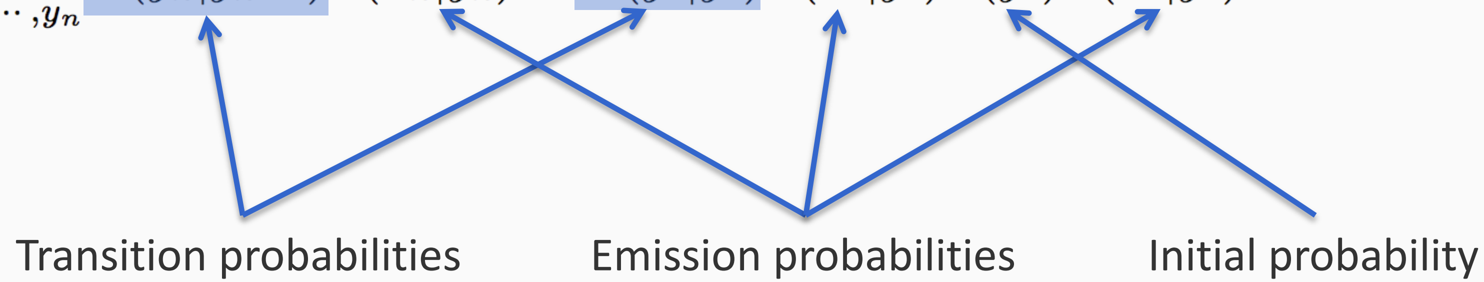
Initial probability

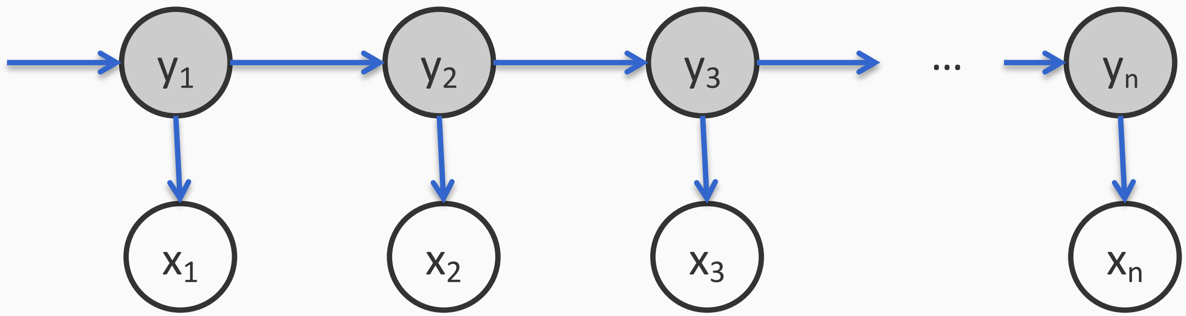


Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$



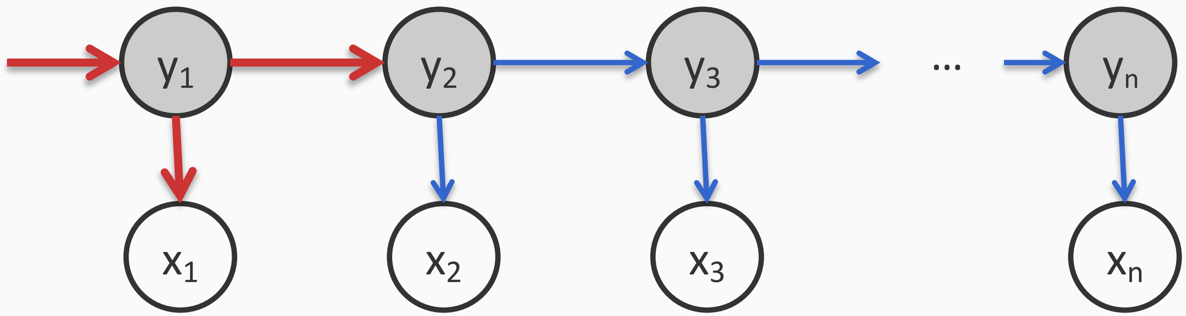


Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$



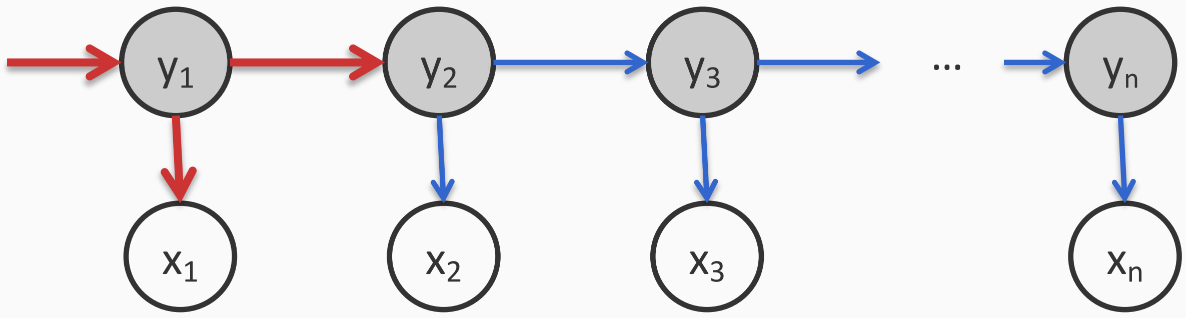
Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

The only terms that depend on y_1



Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

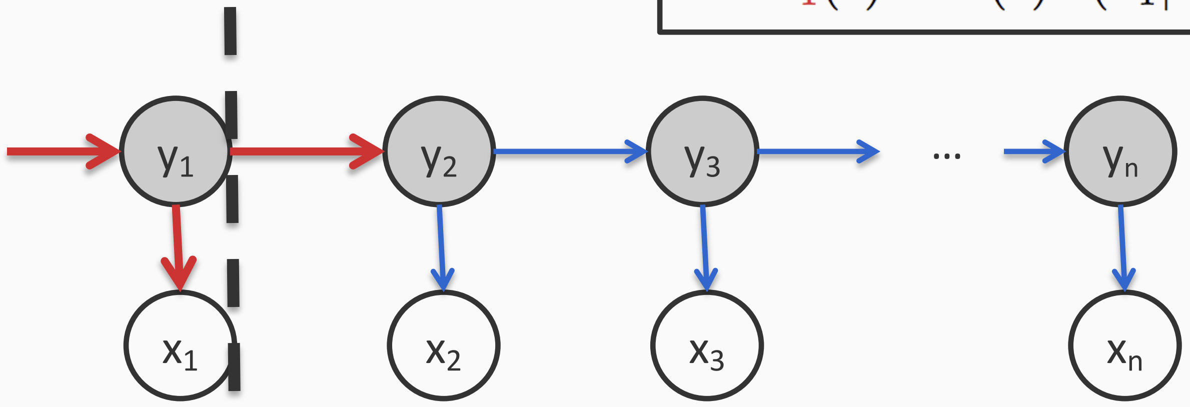
$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

Abstract away the score for all decisions till here into score_1

$$\text{score}_1(s) = P(s)P(x_1|s)$$



Deriving the recursive algorithm

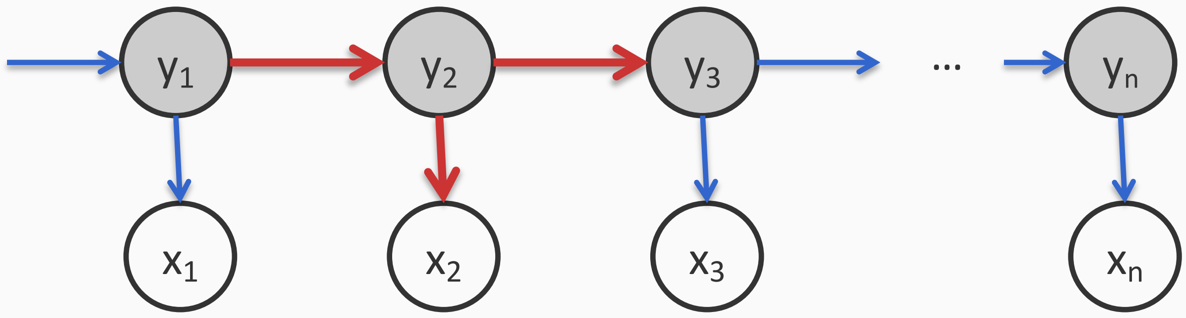
$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3) \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$



Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

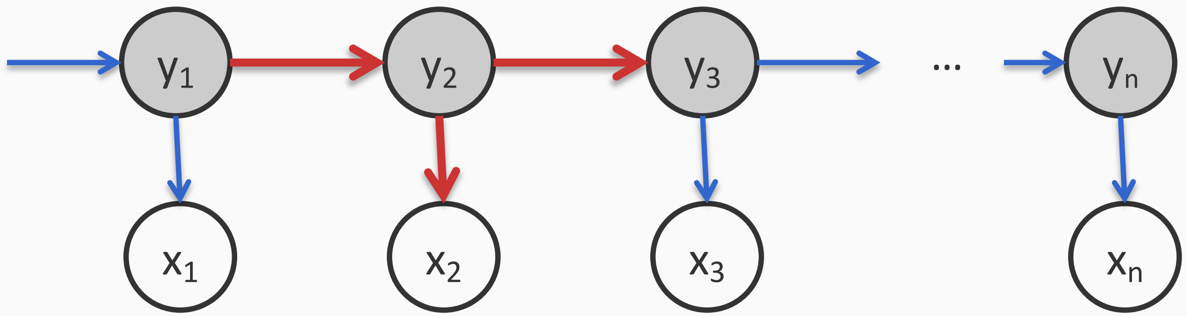
$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3) \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

Only terms that depend on y_2



Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

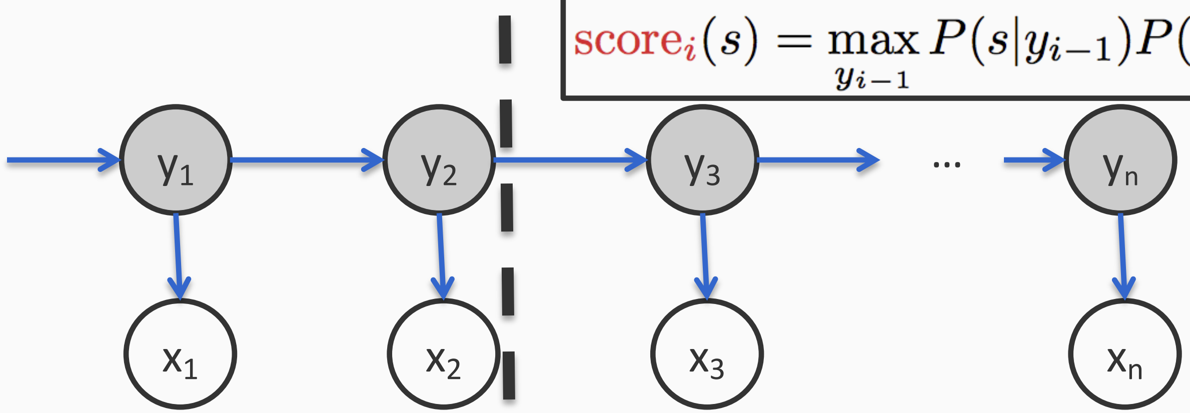
$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3) \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3)\text{score}_2(y_2)$$

$$\text{score}_i(s) = \max_{y_{i-1}} P(s|y_{i-1})P(x_i|s)\text{score}_{i-1}(y_{i-1})$$



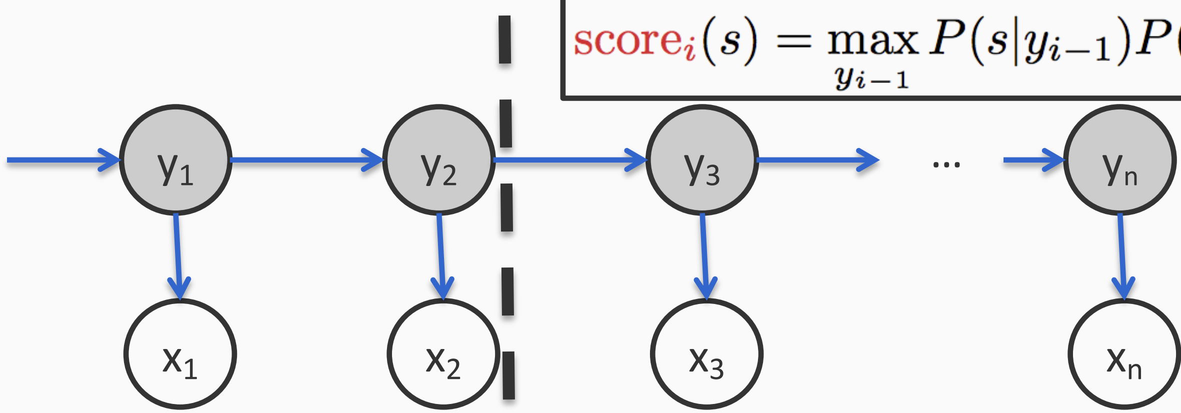
Abstract away the score for all decisions till here into score

Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\begin{aligned}
&\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1) \\
&= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1) \\
&= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1) \\
&= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3) \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1) \\
&= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3)\text{score}_2(y_2)
\end{aligned}$$

$$\text{score}_i(s) = \max_{y_{i-1}} P(s|y_{i-1})P(x_i|s)\text{score}_{i-1}(y_{i-1})$$



Abstract away the score for all decisions till here into score

Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

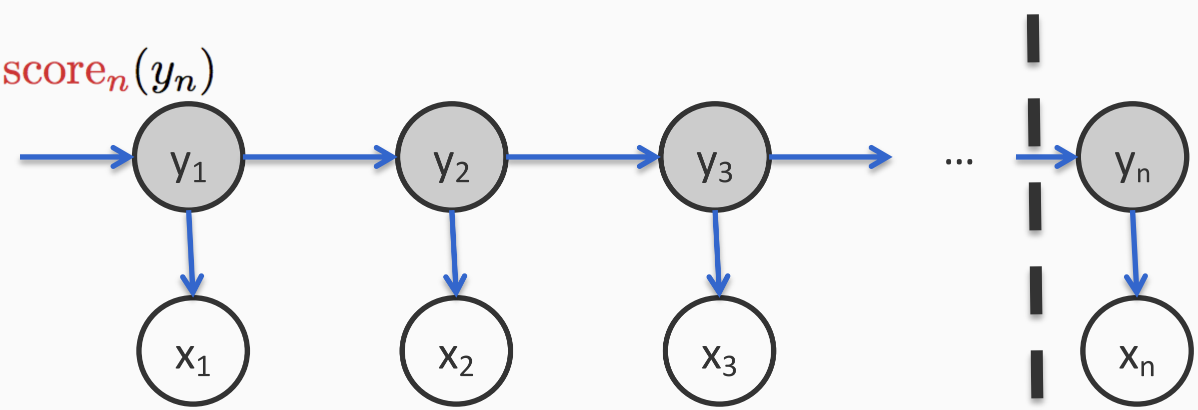
$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3) \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3)\text{score}_2(y_2)$$

$$\vdots$$

$$= \max_{y_n} \text{score}_n(y_n)$$



Abstract away the score for all decisions till here into score

Deriving the recursive algorithm

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n$$

$$= \max_{y_3, \cdots, y_n} P(y_n \quad \ldots \quad x_2|y_2)\text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n$$

$$\vdots$$

$$= \max_{y_n} \text{score}_n(y_n)$$

$$\text{score}_1(s) = P(s)P(x_1|s)$$

$$\text{score}_i(s) = \max_{y_{i-1}} P(s|y_{i-1})P(x_i|s)\text{score}_{i-1}(y_{i-1})$$

Viterbi algorithm

Max-product algorithm for first order sequences

1. Initial: For each state s, calculate

$$score_1(s) = P(s)P(x_1 \mid s)$$

2. Recurrence: For i = 2 to n, for every state s, calculate

$$score_i(s) = \max_{y_{i-1}} P(s \mid y_{i-1})P(x_i \mid s)score_{i-1}(y_{i-1})$$

3. At the final state: calculate

$$\max_{y_{i-1}} P(y, x \mid \pi, A, B) = \max_{s} score_n(s)$$

Viterbi algorithm

Max-product algorithm for first order sequences

π: Initial probabilities
A: Transitions
B: Emissions

1. Initial: For each state s, calculate

$$score_1(s) = P(s)P(\, x_1 \mid s\,) = \pi_s B_{x_1,s}$$

2. Recurrence: For i = 2 to n, for every state s, calculate

$$score_i(s) = \max_{y_{i-1}} P(\, s \mid y_{i-1}\,)P(\, x_i \mid s\,)score_{i-1}(y_{i-1})$$
$$= \max_{y_{i-1}} A_{y_{i-1},s} B_{s,x_i} score_{i-1}(y_{i-1})$$

3. At the final state: calculate

$$\max_{y_{i-1}} P(\, y, x \mid \pi, A, B\,) = \max_{s} score_n(s)$$

Viterbi algorithm

Max-product algorithm for first order sequences

π: Initial probabilities
A: Transitions
B: Emissions

1. Initial: For each state s, calculate

$$score_1(s) = P(s)P(x_1 \mid s) = \pi_s B_{x_1,s}$$

2. Recurrence: For i = 2 to n, for every state s, calculate

$$score_i(s) = \max_{y_{i-1}} P(s \mid y_{i-1}) P(x_i \mid s) score_{i-1}(y_{i-1})$$
$$= \max_{y_{i-1}} A_{y_{i-1},s} B_{s,x_i} score_{i-1}(y_{i-1})$$

3. At the final state: calculate

$$\max_{y_{i-1}} P(y, x \mid \pi, A, B) = \max_{s} score_n(s)$$

This only calculates the max. To get final answer (*argmax*):

- keep track of which state corresponds to the max at each step
- build the answer using these back pointers

Viterbi algorithm

Max-product algorithm for first order sequences

π: Initial probabilities
A: Transitions
B: Emissions

1. Initial: For each state s, calculate

$$score_1(s) = P(s)P(x_1 \mid s) = \pi_s B_{x_1,s}$$

2. Recurrence: For i = 2 to n, for every state s, calculate

$$score_i(s) = \max_{y_{i-1}} P(s \mid y_{i-1}) P(x_i \mid s) score_{i-1}(y_{i-1})$$
$$= \max_{y_{i-1}} A_{y_{i-1},s} B_{s,x_i} score_{i-1}(y_{i-1})$$

3. At the final state: calculate

$$\max_{y_{i-1}} P(y, x \mid \pi, A, B) = \max_{s} score_n(s)$$

This only calculates the max. To get final answer (*argmax*):

- keep track of which state corresponds to the max at each step
- build the answer using these back pointers

Questions?

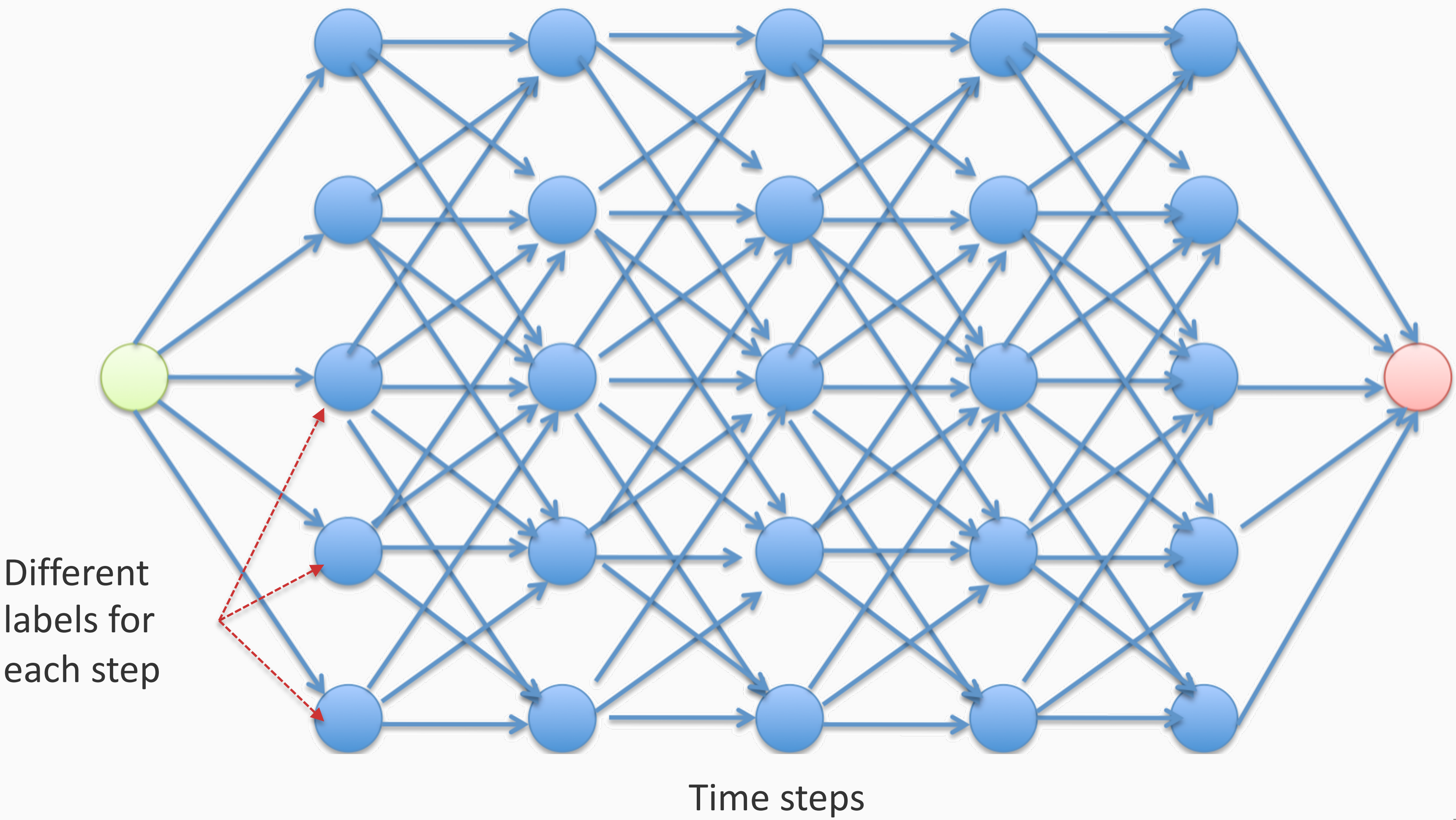
General idea

- Dynamic programming
  - The best solution for the full problem relies on best solution to sub-problems
  - Memoize partial computation

- Examples
  - Viterbi algorithm
  - Dijkstra’s shortest path algorithm
  - …

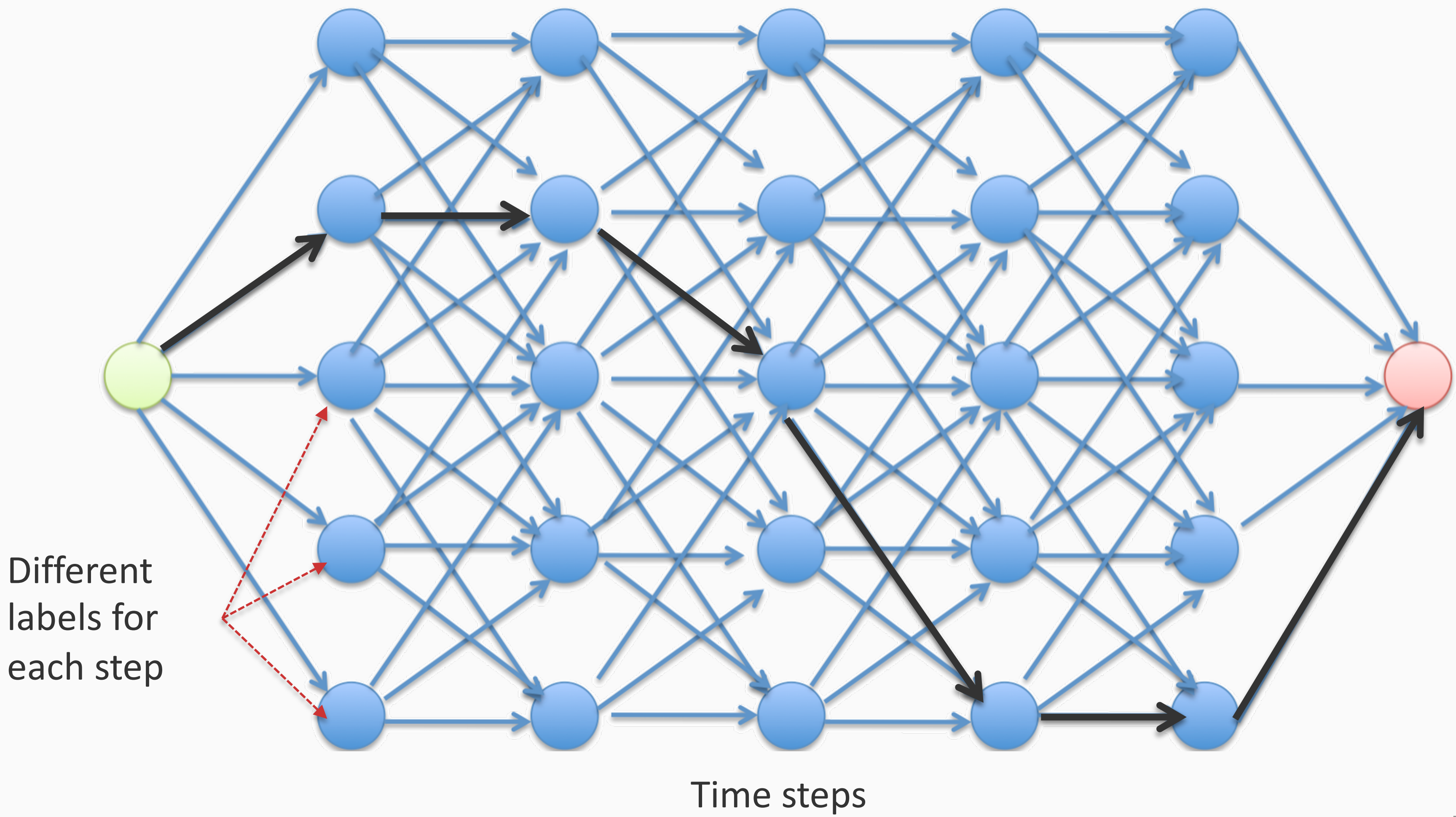
Viterbi algorithm as best path

Goal: To find the highest scoring path in this trellis



Viterbi algorithm as best path

Goal: To find the highest scoring path in this trellis



Complexity of inference

- Complexity parameters
  - Input sequence length: n
  - Number of states: K
- Memory
  - Storing the table: nK (scores for all states at each position)
- Runtime
  - At each step, go over pairs of states
  - $O(nK^2)$

Questions?

Outline

- Sequence models
- Hidden Markov models
  - Inference with HMM
  - *Learning*
- Conditional Models and Local Classifiers
- Global models
  - Conditional Random Fields
  - Structured Perceptron for sequences

Learning HMM parameters

Assume that we know the number of states in the HMM

Two possible scenarios

Learning HMM parameters

Assume that we know the number of states in the HMM

Two possible scenarios

1. We are given a data set D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>} of sequences labeled with states

   And we have to learn the parameters of the HMM (π, A, B)

Learning HMM parameters

Assume that we know the number of states in the HMM

Two possible scenarios

1. We are given a data set D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>} of sequences labeled with states

   And we have to learn the parameters of the HMM (π, A, B)

      Supervised learning with complete data

Learning HMM parameters

Assume that we know the number of states in the HMM

Two possible scenarios

1. We are given a data set D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>} of sequences labeled with states

   And we have to learn the parameters of the HMM (π, A, B)

   Supervised learning with complete data

2. We are given only a collection of sequences D = {$\mathbf{x}_i$}

   And we have to learn the parameters of the HMM (π, A, B)

Learning HMM parameters

Assume that we know the number of states in the HMM

Two possible scenarios

1. We are given a data set D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>} of sequences labeled with states

   And we have to learn the parameters of the HMM (π, A, B)

   Supervised learning with complete data

2. We are given only a collection of sequences D = {$\mathbf{x}_i$}

   And we have to learn the parameters of the HMM (π, A, B)

   Unsupervised learning, with incomplete data

   EM algorithm and its siblings: a subsequent lecture

Learning HMM parameters

Assume that we know the number of states in the HMM

Two possible scenarios

1. We are given a data set D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>} of sequences labeled with states

   And we have to learn the parameters of the HMM (π, A, B)

   Supervised learning with complete data

2. We are given only a collection of sequences D = {$\mathbf{x}_i$}

   And we have to learn the parameters of the HMM (π, A, B)

   Unsupervised learning, with incomplete data

   EM algorithm and its siblings: a subsequent lecture

Supervised learning of HMM

We are given a dataset D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>}

- Each $\mathbf{x}_i$ is a sequence of observations and $\mathbf{y}_i$ is a sequence of states that correspond to $\mathbf{x}_i$

Goal: Learn initial, transition, emission distributions (π, A, B)

Supervised learning of HMM

We are given a dataset D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>}

– Each $\mathbf{x}_i$ is a sequence of observations and $\mathbf{y}_i$ is a sequence of states that correspond to $\mathbf{x}_i$

Goal: Learn initial, transition, emission distributions (π, A, B)

- How do we learn the parameters of the HMM?

Supervised learning of HMM

We are given a dataset D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>}

- Each $\mathbf{x}_i$ is a sequence of observations and $\mathbf{y}_i$ is a sequence of states that correspond to $\mathbf{x}_i$

Goal: Learn initial, transition, emission distributions (π, A, B)

- How do we learn the parameters of the HMM?
  - The maximum likelihood principle

Where have we seen this before?

Supervised learning of HMM

We are given a dataset D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>}

– Each $\mathbf{x}_i$ is a sequence of observations and $\mathbf{y}_i$ is a sequence of states that correspond to $\mathbf{x}_i$

Goal: Learn initial, transition, emission distributions (π, A, B)

- How do we learn the parameters of the HMM?
  – The maximum likelihood principle

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi, A, B} P(D|\pi, A, B)$$

Supervised learning of HMM

We are given a dataset D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>}

- Each $\mathbf{x}_i$ is a sequence of observations and $\mathbf{y}_i$ is a sequence of states that correspond to $\mathbf{x}_i$

Goal: Learn initial, transition, emission distributions (π, A, B)

- How do we learn the parameters of the HMM?
  - The maximum likelihood principle

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi, A, B} P(D|\pi, A, B) = \max_{\pi, A, B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

Supervised learning of HMM

We are given a dataset D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>}

- Each $\mathbf{x}_i$ is a sequence of observations and $\mathbf{y}_i$ is a sequence of states that correspond to $\mathbf{x}_i$

Goal: Learn initial, transition, emission distributions (π, A, B)

- How do we learn the parameters of the HMM?
  - The maximum likelihood principle

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi, A, B} P(D|\pi, A, B) = \max_{\pi, A, B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i | \pi, A, B)$$

And we know how to write this in terms of the parameters of the HMM

Supervised learning details

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi, A, B} P(D|\pi, A, B) = \max_{\pi, A, B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

(π, A, B) can be estimated separately just by counting

- Makes learning simple and fast

Supervised learning details

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi,A,B} P(D|\pi, A, B) = \max_{\pi,A,B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

(π, A, B) can be estimated separately just by counting

– Makes learning simple and fast

Initial probabilities

$$\pi_s = \frac{\text{count}(\text{start} \rightarrow s)}{n}$$

Supervised learning details

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi, A, B} P(D|\pi, A, B) = \max_{\pi, A, B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

(π, A, B) can be estimated separately just by counting

- Makes learning simple and fast

Initial probabilities

$$\pi_s = \frac{\text{count}(\text{start} \to s)}{n}$$

Number of instances where the first state is s

Number of examples

Supervised learning details

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi, A, B} P(D|\pi, A, B) = \max_{\pi, A, B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

(π, A, B) can be estimated separately just by counting

- Makes learning simple and fast

Initial probabilities

$$\pi_s = \frac{\text{count}(\text{start} \to s)}{n}$$

Transition probabilities

$$A_{s',s} = \frac{\text{count}\left(s \to s'\right)}{\text{count}\left(s\right)}$$

Supervised learning details

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi, A, B} P(D|\pi, A, B) = \max_{\pi, A, B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

(π, A, B) can be estimated separately just by counting

- Makes learning simple and fast

Initial probabilities

$$\pi_s = \frac{\text{count}(\text{start} \rightarrow s)}{n}$$

Transition probabilities

$$A_{s',s} = \frac{\text{count}\,(s \rightarrow s')}{\text{count}\,(s)}$$

Emission probabilities

$$B_{s,x} = \frac{\text{count}\begin{pmatrix} s \\ \downarrow \\ x \end{pmatrix}}{\text{count}\,(s)}$$

Supervised learning details

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi, A, B} P(D|\pi, A, B) = \max_{\pi, A, B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

(π, A, B) can be estimated separately just by counting

– Makes learning simple and fast

Initial probabilities

$$\pi_s = \frac{\text{count}(\text{start} \rightarrow s)}{n}$$

Transition probabilities

$$A_{s',s} = \frac{\text{count}\,(s \rightarrow s')}{\text{count}\,(s)}$$

Emission probabilities

$$B_{s,x} = \frac{\text{count}\begin{pmatrix} s \\ \downarrow \\ x \end{pmatrix}}{\text{count}\,(s)}$$

Exercise: Derive these using derivatives of the log likelihood. Requires Lagrangian multipliers.

Priors and smoothing

- Maximum likelihood estimation works best with lots of annotated data
  - Never the case
- Priors inject information about the probability distributions
  - Dirichlet priors for multinomial distributions
- Effectively additive smoothing
  - Add small constants to the counts

Hidden Markov Models summary

- Predicting sequences
  - As many output states as observations
- Markov assumption helps decompose the score
- Several algorithmic questions
  - Most likely state
  - Learning parameters
    - Supervised, Unsupervised
  - Probability of an observation sequence
    - Sum over all assignments to states, replace max with sum in Viterbi
  - Probability of state for each observation
    - Sum over all assignments to all other states

Questions?

Outline

- Sequence models
- Hidden Markov models
  - Inference with HMM
  - Learning
- Conditional Models and Local Classifiers
- Global models
  - Conditional Random Fields
  - Structured Perceptron for sequences