Markov and Hidden Markov Models

COMP9418 — Advanced Topics in Statistical Machine Learning

Edwin V. Bonilla

School of Computer Science and Engineering
UNSW Sydney

September 6th, 2017

(Last Update: Tuesday 5th September, 2017 at 20:58)

Acknowledgments

- [Barber, BRML, 2012] Bayesian Reasoning and Machine Learning, David Barber, 2012
  `www.cs.ucl.ac.uk/staff/D.Barber/brml`

Aims (1)

This lecture will allow you to understand and apply some common probabilistic models for sequential data. In particular, following it you should be able to:

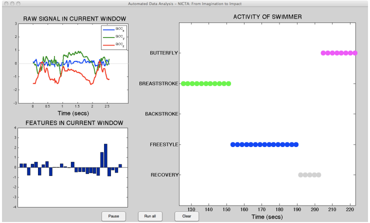
- Apply fully-observable Markov models to the analysis of sequential data.
- Carry out parameter estimation in first-order Markov chains and understand the complexity of this task for higher-order models.
- Carry out clustering of sequential data via mixtures of Markov chains.
- Understand how state space models capture long-term dependencies via the introduction of latent variables.

Aims (2)

- Distinguish the main inference problems, namely smoothing, filtering and prediction, that we can address with Hidden Markov Models (HMMs).
- Solve the above inferential problems along with the most-likely-hidden-path problem via efficient recursions in HMMs.
- Train HMMs for whole-sequence classification and time-dependent classification using a generative or a discriminative approach.

Dealing with Sequential Data (1)

Activity Recognition using Wearable Sensors



Observations may be correlated in time.

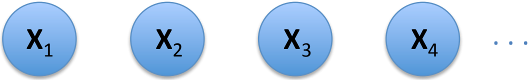
Dealing with Sequential Data (2)

- So far we have assumed iid data
  - Likelihood factorizes across observations
- This is unrealistic for many situations where data is inherently **sequential**
- Temporal data: financial forecasting, currency exchange rate, speech, sensor data, tracking
- Non-temporal data: sequence of characters in an english sentence, sequence of nucleotides in DNA
- We will assume stationary distributions, i.e. independence of time
  - But only the data evolves in time

How to Model Dependencies in Sequential Data?

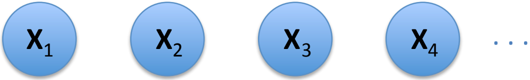
- Recent observations are more informative than historical ones
- Can have fully observed models where variables are linked through statistical dependencies
- Alternative, we can introduce latent variables (cf Gaussian mixtures)

- We can simply ignore the sequential nature of the data.



Simplest Approach First?

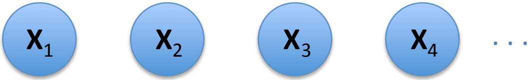
- We can simply ignore the sequential nature of the data.



- Example: Binary variable: rain/not rain a day

Simplest Approach First?

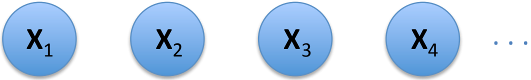
- We can simply ignore the sequential nature of the data.



- Example: Binary variable: rain/not rain a day
- Predicting rain/not rain tomorrow would simply account for frequencies

Simplest Approach First?

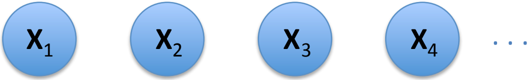
- We can simply ignore the sequential nature of the data.



- Example: Binary variable: rain/not rain a day
- Predicting rain/not rain tomorrow would simply account for frequencies
- Clearly historical observations (at least short term) are important!

Simplest Approach First?

- We can simply ignore the sequential nature of the data.



- Example: Binary variable: rain/not rain a day
- Predicting rain/not rain tomorrow would simply account for frequencies
- Clearly historical observations (at least short term) are important!
- But if you live in Scotland you do not need any model!

Outline

1. Markov Chains

2. Mixtures of Markov Chains

3. Hidden Markov Models

General Formulation for Fully Observable Models

Notation: We will denote $\mathbf{y}_{1:T} \stackrel{\text{def}}{=} \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T$

General Formulation for Fully Observable Models

Notation: We will denote $\mathbf{y}_{1:T} \stackrel{\text{def}}{=} \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T$

Without loss of generality we can express:

General Formulation for Fully Observable Models

Notation: We will denote $\mathbf{y}_{1:T} \stackrel{\text{def}}{=} \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T$

Without loss of generality we can express:

$$p(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1)$$ What are we using here?

General Formulation for Fully Observable Models

Notation: We will denote $\mathbf{y}_{1:T} \stackrel{\text{def}}{=} \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T$

Without loss of generality we can express:

$$p(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1)$$

What are we using here?

$$p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_3|\mathbf{x}_1, \mathbf{x}_2)$$

General Formulation for Fully Observable Models

Notation: We will denote $\mathbf{y}_{1:T} \stackrel{\text{def}}{=} \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T$

Without loss of generality we can express:

$$p(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1) \quad \text{What are we using here?}$$

$$p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_3|\mathbf{x}_1, \mathbf{x}_2)$$

$$p(\mathbf{x}_{1:T}) = p(\mathbf{x}_1)\prod_{t=2}^{T} p(\mathbf{x}_t|\mathbf{x}_{1:t-1})$$

General Formulation for Fully Observable Models

Notation: We will denote $\mathbf{y}_{1:T} \stackrel{\text{def}}{=} \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T$

Without loss of generality we can express:

$$p(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1) \quad \text{What are we using here?}$$

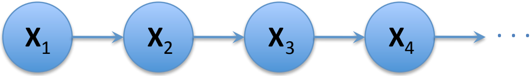
$$p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_3|\mathbf{x}_1, \mathbf{x}_2)$$

$$p(\mathbf{x}_{1:T}) = p(\mathbf{x}_1)\prod_{t=2}^{T} p(\mathbf{x}_t|\mathbf{x}_{1:t-1})$$

Our initial approach will be to drop some of the long-term dependencies in: $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}) = p(\mathbf{x}_t|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{t-1})$

First Order Markov Chain

- A Markov chain is defined on either discrete or continuous variables.
- In a first order Markov chain each observation only depends on its immediate past:



$$p(\mathbf{x}_{1:t}) = p(\mathbf{x}_1) \prod_{t=1}^{T} p(\mathbf{x}_t | \mathbf{x}_{t-1}).$$

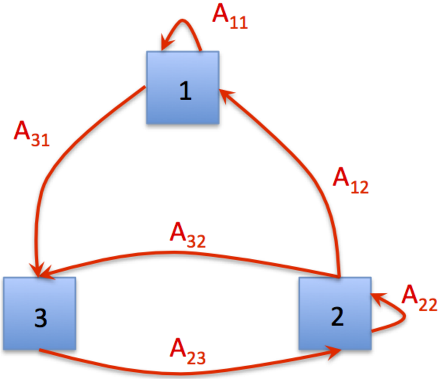
- If the chain is **stationary** $p(\mathbf{x}_t = \mathbf{s} | \mathbf{x}_{t-1} = \mathbf{s}') = f(\mathbf{s}, \mathbf{s}')$
    - Sometimes this is also called homogeneous

Transition Diagram

Consider a discrete state Markov chain with 3 states and define the **transition probabilities**: $A_{ij} = p(x_t = i | x_{t-1} = j)$:

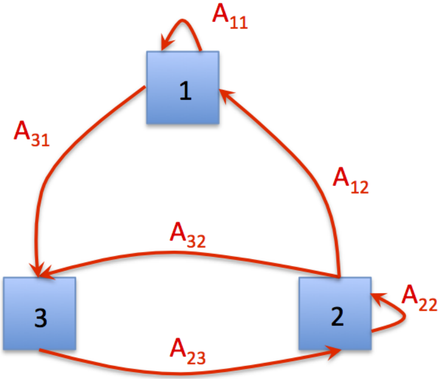
Transition Diagram

Consider a discrete state Markov chain with 3 states and define the **transition probabilities**: $A_{ij} = p(x_t = i | x_{t-1} = j)$:



Transition Diagram

Consider a discrete state Markov chain with 3 states and define the **transition probabilities**: $A_{ij} = p(x_t = i | x_{t-1} = j)$:



Missing link from i to j simply indicates that $A_{ji} = 0$.

Inference on a Markov Chain

Marginal Distribution

Given a discrete-state first-order Markov chain with K states, we are interested in the marginal:

$$p(x_t = i) = \sum_{j=1}^{K} \underbrace{p(x_t = i | x_{t-1} = j)}_{A_{ij}} p(x_{t-1} = j)$$

Given a discrete-state first-order Markov chain with K states, we are interested in the marginal:

$$p(x_t = i) = \sum_{j=1}^{K} \underbrace{p(x_t = i | x_{t-1} = j)}_{A_{ij}} p(x_{t-1} = j)$$

This defines the marginal recursively.

Inference on a Markov Chain

Marginal Distribution

Given a discrete-state first-order Markov chain with K states, we are interested in the marginal:

$$p(x_t = i) = \sum_{j=1}^{K} \underbrace{p(x_t = i | x_{t-1} = j)}_{A_{ij}} p(x_{t-1} = j)$$

This defines the marginal recursively.

Let $\mathbf{p}_1 = (p(x_1 = 1), p(x_1 = 2), \ldots, p(x_1 = K))^T$, then:

Inference on a Markov Chain

Marginal Distribution

Given a discrete-state first-order Markov chain with K states, we are interested in the marginal:

$$p(x_t = i) = \sum_{j=1}^{K} \underbrace{p(x_t = i | x_{t-1} = j)}_{A_{ij}} p(x_{t-1} = j)$$

This defines the marginal recursively.

Let $\mathbf{p}_1 = (p(x_1 = 1), p(x_1 = 2), \ldots, p(x_1 = K))^T$, then:

$$\mathbf{p}_2 = \mathbf{A}\mathbf{p}_1, \quad \mathbf{p}_3 = \mathbf{A}\mathbf{p}_2 = \mathbf{A}^2\mathbf{p}_1, \quad \ldots \quad \mathbf{p}_t = \mathbf{A}^{t-1}\mathbf{p}_1$$

Given a discrete-state first-order Markov chain with K states, we are interested in the marginal:

$$p(x_t = i) = \sum_{j=1}^{K} \underbrace{p(x_t = i | x_{t-1} = j)}_{A_{ij}} p(x_{t-1} = j)$$

This defines the marginal recursively.

Let $\mathbf{p}_1 = (p(x_1 = 1), p(x_1 = 2), \ldots, p(x_1 = K))^T$, then:

$$\mathbf{p}_2 = \mathbf{A}\mathbf{p}_1, \quad \mathbf{p}_3 = \mathbf{A}\mathbf{p}_2 = \mathbf{A}^2\mathbf{p}_1, \quad \ldots \quad \mathbf{p}_t = \mathbf{A}^{t-1}\mathbf{p}_1$$

Interpretation: The frequency that we visit a state at (time) step t given that we started from $p(x_1)$ and drew samples from the transition model.

Equilibrium Distribution of a Markov Chain

$$A = \begin{pmatrix} 0.9000 & 0.3000 \\ 0.1000 & 0.7000 \end{pmatrix} \quad A^5 = \begin{pmatrix} 0.7694 & 0.6917 \\ 0.2306 & 0.3083 \end{pmatrix}$$

$$A^{10} = \begin{pmatrix} 0.7515 & 0.7455 \\ 0.2485 & 0.2545 \end{pmatrix} \quad A^{20} = \begin{pmatrix} 0.7500 & 0.7500 \\ 0.2500 & 0.2500 \end{pmatrix}$$

Equilibrium Distribution of a Markov Chain

$$A = \begin{pmatrix} 0.9000 & 0.3000 \\ 0.1000 & 0.7000 \end{pmatrix} \quad A^5 = \begin{pmatrix} 0.7694 & 0.6917 \\ 0.2306 & 0.3083 \end{pmatrix}$$

$$A^{10} = \begin{pmatrix} 0.7515 & 0.7455 \\ 0.2485 & 0.2545 \end{pmatrix} \quad A^{20} = \begin{pmatrix} 0.7500 & 0.7500 \\ 0.2500 & 0.2500 \end{pmatrix}$$

Therefore, as $t \to \infty$, $\mathbf{p}_\infty$ is independent of the initial $\mathbf{p}_1$ and it is called the **equilibrium distribution**.

- The proportion of times we will visit the corresponding state (in the limiting case)

Equilibrium Distribution of a Markov Chain

$$A = \begin{pmatrix} 0.9000 & 0.3000 \\ 0.1000 & 0.7000 \end{pmatrix} \quad A^5 = \begin{pmatrix} 0.7694 & 0.6917 \\ 0.2306 & 0.3083 \end{pmatrix}$$

$$A^{10} = \begin{pmatrix} 0.7515 & 0.7455 \\ 0.2485 & 0.2545 \end{pmatrix} \quad A^{20} = \begin{pmatrix} 0.7500 & 0.7500 \\ 0.2500 & 0.2500 \end{pmatrix}$$

Therefore, as $t \to \infty$, $\mathbf{p}_\infty$ is independent of the initial $\mathbf{p}_1$ and it is called the **equilibrium distribution**.

- The proportion of times we will visit the corresponding state (in the limiting case)

Do all Markov chains have an equilibrium distribution?

Let us define the matrix $\mathbf{H}$ such that:

$H_{ij} = 1$ if website j hyperlinks website i and $H_{ij} = 0$ otherwise.

Let us define the matrix $\mathbf{H}$ such that:

$H_{ij} = 1$ if website j hyperlinks website i and $H_{ij} = 0$ otherwise.

Define a Markov chain with the following transitions:

$$A_{ij} = \frac{H_{ij}}{\sum_k H_{kj}}$$

Equilibrium Distribution of a Markov Chain

PageRank Example (From Barber, BRML, 2012)

Let us define the matrix $\mathbf{H}$ such that:

$H_{ij} = 1$ if website j hyperlinks website i and $H_{ij} = 0$ otherwise.

Define a Markov chain with the following transitions:

$$A_{ij} = \frac{H_{ij}}{\sum_k H_{kj}}$$

Equilibrium Distribution $p_\infty(i)$: The relative number of times we will visit website i if we follow the links at random $\rightarrow$ importance!

Equilibrium Distribution of a Markov Chain

PageRank Example (From Barber, BRML, 2012)

Let us define the matrix $\mathbf{H}$ such that:

$H_{ij} = 1$ if website j hyperlinks website i and $H_{ij} = 0$ otherwise.

Define a Markov chain with the following transitions:

$$A_{ij} = \frac{H_{ij}}{\sum_k H_{kj}}$$

Equilibrium Distribution $p_\infty(i)$: The relative number of times we will visit website i if we follow the links at random $\rightarrow$ importance!

An overly simplistic search engine:

1. For each website list the words associated with it
2. Make an "inverse" list of websites containing word w
3. Rank websites containing w according to equilibrium distribution

Learning the Parameters of a Markov Chain

Given a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, K\}$:

Learning the Parameters of a Markov Chain

Given a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, K\}$:

Goal Learn $\theta_0^i \stackrel{\text{def}}{=} p(x_1 = i)$ and $\theta_j^i \stackrel{\text{def}}{=} p(x_t = i | x_{t-1} = j)$ for $t > 1$.

Learning the Parameters of a Markov Chain

Given a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, K\}$:

Goal Learn $\theta_0^i \stackrel{\text{def}}{=} p(x_1 = i)$ and $\theta_j^i \stackrel{\text{def}}{=} p(x_t = i | x_{t-1} = j)$ for $t > 1$.

As usual, we write down the data likelihood:

$$p(\mathcal{D}|\theta) = \prod_{n=1}^{N} \prod_{i=1}^{K} (\theta_0^i)^{\mathbb{I}[x_1^n = i]} \prod_{t=2}^{T} \prod_{j=1}^{K} (\theta_j^i)^{\mathbb{I}[x_t^n = i, x_{t-1}^n = j]}.$$

Learning the Parameters of a Markov Chain

Given a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, K\}$:

Goal Learn $\theta_0^i \stackrel{\text{def}}{=} p(x_1 = i)$ and $\theta_j^i \stackrel{\text{def}}{=} p(x_t = i | x_{t-1} = j)$ for $t > 1$.

As usual, we write down the data likelihood:

$$p(\mathcal{D}|\theta) = \prod_{n=1}^{N} \prod_{i=1}^{K} (\theta_0^i)^{\mathbb{I}[x_1^n = i]} \prod_{t=2}^{T} \prod_{j=1}^{K} (\theta_j^i)^{\mathbb{I}[x_t^n = i, x_{t-1}^n = j]}.$$

Taking the log and optimizing wrt θ subject to $\sum_{i=1}^K \theta_j^i = 1$:

$$\theta_0^i = \frac{\sum_{n=1}^N \mathbb{I}[x_1^n = i]}{N} \quad \theta_j^i = \frac{\sum_{n=1}^N \sum_{t=2}^T \mathbb{I}[x_t^n = i, x_{t-1}^n = j]}{\sum_{n=1}^N \sum_{t=2}^T \mathbb{I}[x_{t-1}^n = j]}$$

Learning the Parameters of a Markov Chain

Given a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, K\}$:

Goal Learn $\theta_0^i \stackrel{\text{def}}{=} p(x_1 = i)$ and $\theta_j^i \stackrel{\text{def}}{=} p(x_t = i | x_{t-1} = j)$ for $t > 1$.

As usual, we write down the data likelihood:

$$p(\mathcal{D}|\theta) = \prod_{n=1}^{N} \prod_{i=1}^{K} (\theta_0^i)^{\mathbb{I}[x_1^n = i]} \prod_{t=2}^{T} \prod_{j=1}^{K} (\theta_j^i)^{\mathbb{I}[x_t^n = i, x_{t-1}^n = j]}.$$

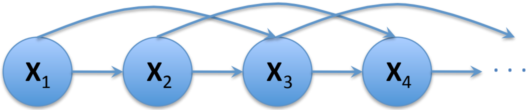
Taking the log and optimizing wrt θ subject to $\sum_{i=1}^K \theta_j^i = 1$:

$$\theta_0^i = \frac{\sum_{n=1}^N \mathbb{I}[x_1^n = i]}{N} \quad \theta_j^i = \frac{\sum_{n=1}^N \sum_{t=2}^T \mathbb{I}[x_t^n = i, x_{t-1}^n = j]}{\sum_{n=1}^N \sum_{t=2}^T \mathbb{I}[x_{t-1}^n = j]}$$

Simply counting occurrences and transitions!

Second Order Markov Chains

We can consider more complex dependencies:



Now the current observation depends on the two previous time steps.

The parameterization for the transitions would be:

$$p(\mathbf{x}_t | \mathbf{x}_{t-2}, \mathbf{x}_{t-1}),$$

which for K-state discrete variables would correspond to $(K-1)(K^2)$ parameters.

How many parameters did we need for the first order Markov chain?

Higher Order Markov Models

- We can increase the flexibility of our models by considering M^{th} order Markov chains.

Higher Order Markov Models

- We can increase the flexibility of our models by considering M^{th} order Markov chains.
- The current observation depends on the previous M observations:

$$p(\mathbf{x}_n | \mathbf{x}_{n-M}, \ldots, \mathbf{x}_{n-1})$$

Higher Order Markov Models

- We can increase the flexibility of our models by considering M^{th} order Markov chains.
- The current observation depends on the previous M observations:

$$p(\mathbf{x}_n | \mathbf{x}_{n-M}, \ldots, \mathbf{x}_{n-1})$$

- We will need $(K-1)K^M$ parameters

Higher Order Markov Models

- We can increase the flexibility of our models by considering M^{th} order Markov chains.
- The current observation depends on the previous M observations:

$$p(\mathbf{x}_n|\mathbf{x}_{n-M}, \ldots, \mathbf{x}_{n-1})$$

- We will need $(K-1)K^M$ parameters
- The number of parameters grows exponentially with the order of the chain

Higher Order Markov Models

- We can increase the flexibility of our models by considering M^{th} order Markov chains.
- The current observation depends on the previous M observations:

$$p(\mathbf{x}_n | \mathbf{x}_{n-M}, \ldots, \mathbf{x}_{n-1})$$

- We will need $(K-1)K^M$ parameters
- The number of parameters grows exponentially with the order of the chain
- What can we do?

Higher Order Markov Models

- We can increase the flexibility of our models by considering M^{th} order Markov chains.
- The current observation depends on the previous M observations:

$$p(\mathbf{x}_n | \mathbf{x}_{n-M}, \ldots, \mathbf{x}_{n-1})$$

- We will need $(K-1)K^M$ parameters
- The number of parameters grows exponentially with the order of the chain
- What can we do?
  - We need to introduce long-term dependencies while avoiding an explosion in the number of parameters

Higher Order Markov Models

- We can increase the flexibility of our models by considering M^{th} order Markov chains.

- The current observation depends on the previous M observations:

$$p(\mathbf{x}_n | \mathbf{x}_{n-M}, \ldots, \mathbf{x}_{n-1})$$

- We will need $(K-1)K^M$ parameters
- The number of parameters grows exponentially with the order of the chain
- What can we do?
  - We need to introduce long-term dependencies while avoiding an explosion in the number of parameters
  - Our old trick of introducing latent variables

Higher Order Markov Models

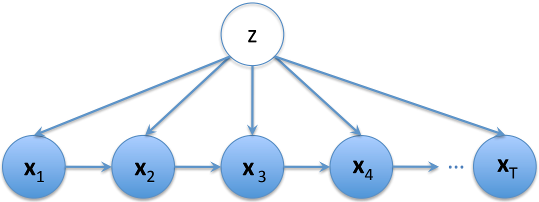
- We can increase the flexibility of our models by considering M^{th} order Markov chains.
- The current observation depends on the previous M observations:

$$p(\mathbf{x}_n | \mathbf{x}_{n-M}, \ldots, \mathbf{x}_{n-1})$$

- We will need $(K-1)K^M$ parameters
- The number of parameters grows exponentially with the order of the chain
- What can we do?
  - We need to introduce long-term dependencies while avoiding an explosion in the number of parameters
  - Our old trick of introducing latent variables
  - As e.g. in the mixture of Gaussians case

Mixtures of Markov Chains

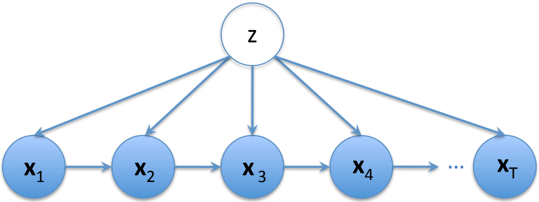
How do we **cluster** a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, K\}$?



- Mixture model with latent variable $z \in \{1, \ldots, M\}$.

Mixtures of Markov Chains

How do we **cluster** a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, K\}$?



- Mixture model with latent variable $z \in \{1, \ldots, M\}$.
- Markov chain conditioned on the latent variable z:

$$p(x_{1:T}|z) = p(x_1|z) \prod_{t=2}^{T} p(x_t|x_{t-1}, z)$$

The marginal distribution is given by:

$$p(\mathbf{x}) = \sum_{m=1}^{M} p(x_{1:T}|z = m)p(z = m).$$

The marginal distribution is given by:

$$p(\mathbf{x}) = \sum_{m=1}^{M} p(x_{1:T} | z = m) p(z = m).$$

Let $\theta = \{\pi_m, \gamma_m^j, \psi_m^{j,\ell} | m = 1, \ldots, M; j, \ell = 1, \ldots, K\}$ be the parameters of the model wit $\pi_m \stackrel{\text{def}}{=} p(z = m)$ and:

$$\gamma_m^j \stackrel{\text{def}}{=} p(x_1 = j | z = m) \quad \psi_m^{j,\ell} \stackrel{\text{def}}{=} p(x_t = j | x_{t-1} = \ell, z = m)$$

The marginal distribution is given by:

$$p(\mathbf{x}) = \sum_{m=1}^{M} p(x_{1:T}|z = m)p(z = m).$$

Let $\theta = \{\pi_m, \gamma_m^j, \psi_m^{j,\ell} | m = 1, \ldots, M; j, \ell = 1, \ldots, K\}$ be the parameters of the model wit $\pi_m \stackrel{\text{def}}{=} p(z = m)$ and:

$$\gamma_m^j \stackrel{\text{def}}{=} p(x_1 = j|z = m) \quad \psi_m^{j,\ell} \stackrel{\text{def}}{=} p(x_t = j|x_{t-1} = \ell, z = m)$$

As with the GMM, direct likelihood optimization is hard.

Instead we use EM obtaining the following updates:

Instead we use EM obtaining the following updates:

$$\pi_m = \frac{1}{N}\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})$$

$$\gamma_m^j = \frac{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\mathbb{I}[x_1^i = j]}{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})}$$

$$\psi_m^{j,\ell} = \frac{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\sum_{t=2}^{T}\mathbb{I}[x_t^i = j, x_{t-1}^i = \ell]}{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\sum_{t=2}^{T}\mathbb{I}[x_{t-1}^i = \ell]}.$$

Instead we use EM obtaining the following updates:

$$\pi_m = \frac{1}{N}\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})$$

$$\gamma_m^j = \frac{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\mathbb{I}[x_1^i = j]}{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})}$$

$$\psi_m^{j,\ell} = \frac{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\sum_{t=2}^{T}\mathbb{I}[x_t^i = j, x_{t-1}^i = \ell]}{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\sum_{t=2}^{T}\mathbb{I}[x_{t-1}^i = \ell]}.$$

These are similar updates to the single Markov chain's but weighted by the posterior $P(z^i = m|\mathbf{x}^i, \theta^{old})$.

Instead we use EM obtaining the following updates:

$$\pi_m = \frac{1}{N}\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})$$

$$\gamma_m^j = \frac{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\mathbb{I}[x_1^i = j]}{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})}$$

$$\psi_m^{j,\ell} = \frac{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\sum_{t=2}^{T}\mathbb{I}[x_t^i = j, x_{t-1}^i = \ell]}{\sum_{i=1}^{N} P(z^i = m|\mathbf{x}^i, \theta^{old})\sum_{t=2}^{T}\mathbb{I}[x_{t-1}^i = \ell]}.$$

These are similar updates to the single Markov chain's but weighted by the posterior $P(z^i = m|\mathbf{x}^i, \theta^{old})$.

The posterior $P(z^i = m|\mathbf{x}^i, \theta^{old})$ can be computed from the above updates straightforwardly (E step). How?

Goal: Efficient ways of modelling long-term dependencies

1. Introduce a **latent variable** $\mathbf{z}_t$ for each observation

Goal: Efficient ways of modelling long-term dependencies

1. Introduce a **latent variable** $\mathbf{z}_t$ for each observation
   - These latent variables may be of different type and dimensionality to the observed ones

Goal: Efficient ways of modelling long-term dependencies

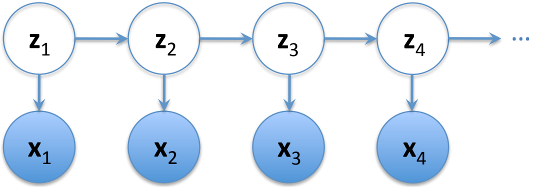
1. Introduce a **latent variable** $\mathbf{z}_t$ for each observation
   - These latent variables may be of different type and dimensionality to the observed ones
2. Make the latent variables form a Markov chain

Goal: Efficient ways of modelling long-term dependencies

1. Introduce a **latent variable** $\mathbf{z}_t$ for each observation
   - These latent variables may be of different type and dimensionality to the observed ones
2. Make the latent variables form a Markov chain
3. Draw observations from these latent variables

Goal: Efficient ways of modelling long-term dependencies

1. Introduce a **latent variable** $\mathbf{z}_t$ for each observation
   - These latent variables may be of different type and dimensionality to the observed ones
2. Make the latent variables form a Markov chain
3. Draw observations from these latent variables

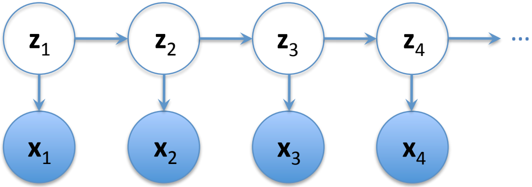


State Space Models

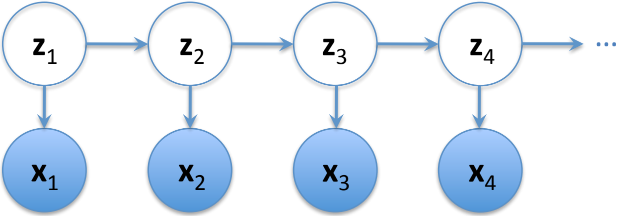
Introducing Complex Dependencies through Latent Variables

Goal: Efficient ways of modelling long-term dependencies

1. Introduce a **latent variable** $\mathbf{z}_t$ for each observation
   - These latent variables may be of different type and dimensionality to the observed ones
2. Make the latent variables form a Markov chain
3. Draw observations from these latent variables



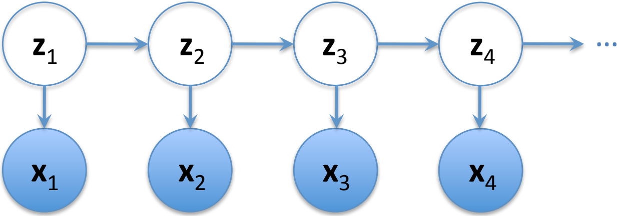
This is known as a state space model.



Joint distribution:

$$p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}) = p(\mathbf{z}_1)p(\mathbf{x}_1|\mathbf{z}_1)\prod_{t=2}^{T} p(\mathbf{z}_t|\mathbf{z}_{t-1})p(\mathbf{x}_t|\mathbf{z}_t)$$

State Space Models
Properties



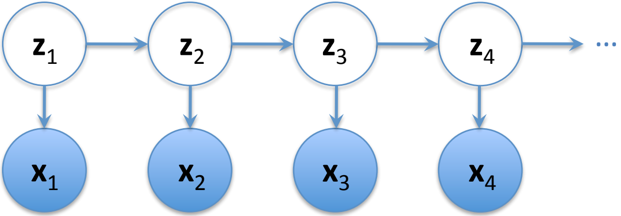
Joint distribution:

$$p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}) = p(\mathbf{z}_1) p(\mathbf{x}_1 | \mathbf{z}_1) \prod_{t=2}^{T} p(\mathbf{z}_t | \mathbf{z}_{t-1}) p(\mathbf{x}_t | \mathbf{z}_t)$$

Predictions for $\mathbf{x}_t$ depend on all the previous observations!

State Space Models

Properties



Joint distribution:

$$p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}) = p(\mathbf{z}_1)p(\mathbf{x}_1|\mathbf{z}_1)\prod_{t=2}^{T} p(\mathbf{z}_t|\mathbf{z}_{t-1})p(\mathbf{x}_t|\mathbf{z}_t)$$

Predictions for $\mathbf{x}_t$ depend on all the previous observations!

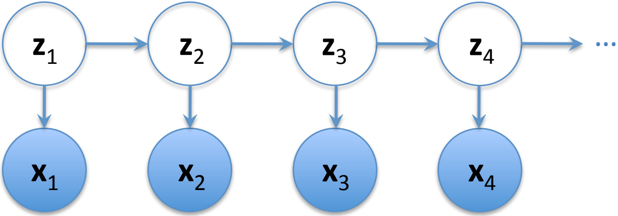
- Can show this using *d-separation*

Hidden Markov models: latent variables are discrete

Linear dynamical systems: latent and visible variables are Gaussian

Hidden Markov Models

Discrete Latent Variables



- Each time slice corresponds to a mixture distribution
- The selection of the mixture component at t depends on the selection of the mixture component at $t-1$
- Widely used in speech recognition, natural language, analysis of biological sequences, etc

Transition Distribution: Assuming $z_t \in \{1, \ldots, K\}$ then:

$$A_{ij} = p(z_t = i | z_{t-1} = j)$$

and an **initial distribution**: $\pi_i = p(z_1 = i)$. So we have a table of $K \times K$ probabilities.

Transition Distribution: Assuming $z_t \in \{1, \ldots, K\}$ then:

$$A_{ij} = p(z_t = i | z_{t-1} = j)$$

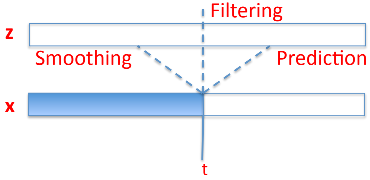
and an **initial distribution**: $\pi_i = p(z_1 = i)$. So we have a table of $K \times K$ probabilities.

Emission Distribution:

- Discrete states $x_t \in \{1, 2, \ldots S\}$: We have the $S \times K$ emission matrix:

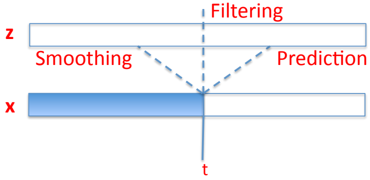
$$B_{ij} = p(x_t = i | z_t = j)$$

- Continuous x_t: z_t selects one of K possible distributions $p(x_t | z_t)$, e.g. a Gaussian: $p(x_t | z_t) = \mathcal{N}(\mathbf{x}_t | \boldsymbol{\mu}_{z_t}, \boldsymbol{\Sigma}_{z_t})$



- Filtering: Inferring the present $p(z_t|x_{1:t})$
- Smoothing: Inferring the past: $p(z_u|x_{1:t})$, $u < t$
- Prediction: Inferring the future: $p(z_s|x_{1:t})$, $s > t$

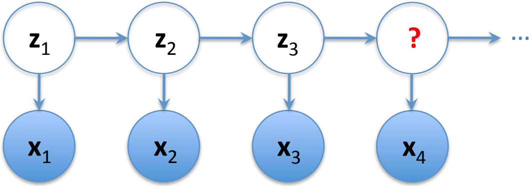
- Likelihood: $p(x_{1:T})$
- Most likely hidden path (Viterbi alignment): $\text{argmax}_{z_{1:T}}\, p(z_{1:T}|x_{1:T})$



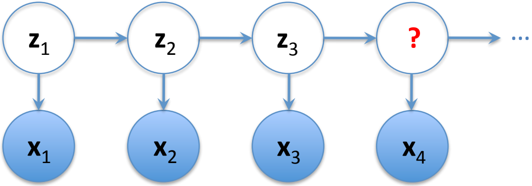
- Filtering: Inferring the present $p(z_t|x_{1:t})$
- Smoothing: Inferring the past: $p(z_u|x_{1:t})$, $u < t$
- Prediction: Inferring the future: $p(z_s|x_{1:t})$, $s > t$
- Likelihood: $p(x_{1:T})$
- Most likely hidden path (Viterbi alignment): $\text{argmax}_{z_{1:T}}\, p(z_{1:T}|x_{1:T})$

We can use any standard inference method in *graphical models* to solve this problems, e.g. using the Junction Tree Algorithm.

- Instead, we will derive recursions directly.

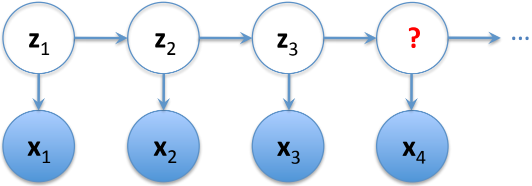


We can find $p(z_t|x_{1:t})$ by considering $p(z_t, x_{1:t})$ and normalizing accordingly.



We can find $p(z_t|x_{1:t})$ by considering $p(z_t, x_{1:t})$ and normalizing accordingly.

$$p(z_t, x_{1:t}) = \sum_{z_{t-1}} p(z_t, z_{t-1}, x_{1:t-1}, x_t) \quad \text{Def. marginal prob.}$$



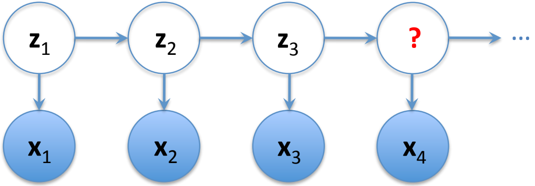
We can find $p(z_t|x_{1:t})$ by considering $p(z_t, x_{1:t})$ and normalizing accordingly.

$$p(z_t, x_{1:t}) = \sum_{z_{t-1}} p(z_t, z_{t-1}, x_{1:t-1}, x_t) \quad \text{Def. marginal prob.}$$

$$= p(x_t|z_t) \sum_{z_{t-1}} p(z_t|z_{t-1}) p(z_{t-1}, x_{1:t-1}) \quad \text{used model independencies}$$

Hidden Markov Models

Filtering (1)

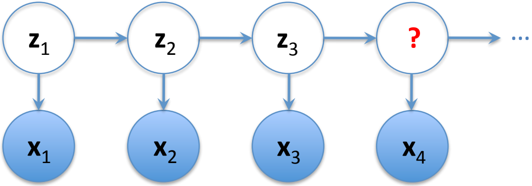


We can find $p(z_t|x_{1:t})$ by considering $p(z_t, x_{1:t})$ and normalizing accordingly.

$$p(z_t, x_{1:t}) = \sum_{z_{t-1}} p(z_t, z_{t-1}, x_{1:t-1}, x_t) \quad \text{Def. marginal prob.}$$

$$= p(x_t|z_t) \sum_{z_{t-1}} p(z_t|z_{t-1}) p(z_{t-1}, x_{1:t-1}) \quad \text{used model independencies}$$

$$\alpha(z_t) = p(x_t|z_t) \sum_{z_{t-1}} p(z_t|z_{t-1}) \alpha(z_{t-1}), \quad t > 1$$



We can find $p(z_t|x_{1:t})$ by considering $p(z_t, x_{1:t})$ and normalizing accordingly.

$$p(z_t, x_{1:t}) = \sum_{z_{t-1}} p(z_t, z_{t-1}, x_{1:t-1}, x_t) \quad \text{Def. marginal prob.}$$

$$= p(x_t|z_t) \sum_{z_{t-1}} p(z_t|z_{t-1}) p(z_{t-1}, x_{1:t-1}) \quad \text{used model independencies}$$

$$\alpha(z_t) = p(x_t|z_t) \sum_{z_{t-1}} p(z_t|z_{t-1}) \alpha(z_{t-1}), \quad t > 1$$

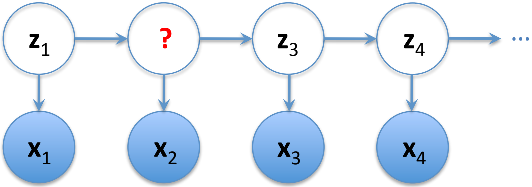
and $\alpha(z_1) = p(z_1)p(x_1|z_1)$

$$\alpha(z_t) = \underbrace{p(x_t|z_t)}_{\text{New evidence}} \underbrace{\sum_{z_{t-1}} \underbrace{p(z_t|z_{t-1})}_{\text{Dynamics}} \alpha(z_{t-1})}_{\text{New prior}}$$

- Filtered distribution propagated forward through the dynamics to reveal a new "prior" at time t
- This distribution is modulated by the observation x_t to incorporate the new evidence

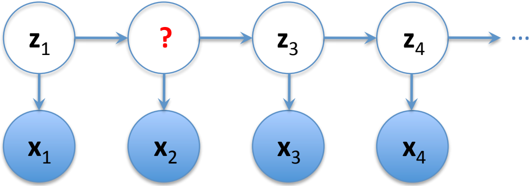
Hidden Markov Models

Smoothing: $p(z_t|x_{1:T})$



Hidden Markov Models

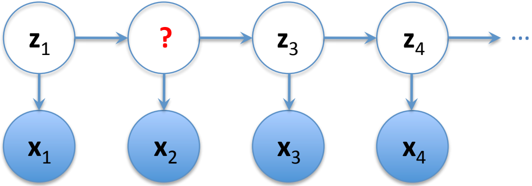
Smoothing: $p(z_t|x_{1:T})$



$$p(z_t, x_{1:T}) = p(z_t, x_{1:t}, x_{t+1:T})$$

Hidden Markov Models

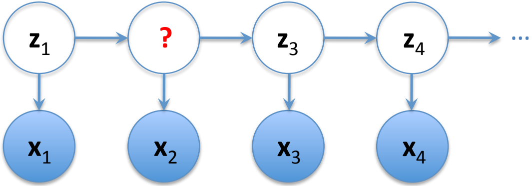
Smoothing: $p(z_t|x_{1:T})$



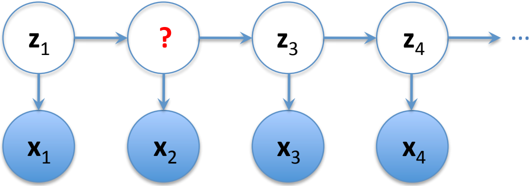
$$p(z_t, x_{1:T}) = p(z_t, x_{1:t}, x_{t+1:T})$$
$$= p(z_t, x_{1:t})p(x_{t+1:T}|z_t, x_{1:t}) \quad \text{product rule}$$

Hidden Markov Models

Smoothing: $p(z_t|x_{1:T})$



$$
\begin{aligned}
p(z_t, x_{1:T}) &= p(z_t, x_{1:t}, x_{t+1:T}) \\
&= p(z_t, x_{1:t}) p(x_{t+1:T}|z_t, x_{1:t}) \qquad \text{product rule} \\
&= \underbrace{p(z_t, x_{1:t})}_{\alpha(z_t)} \underbrace{p(x_{t+1:T}|z_t)}_{\beta(z_t)} \qquad \text{model independencies}
\end{aligned}
$$

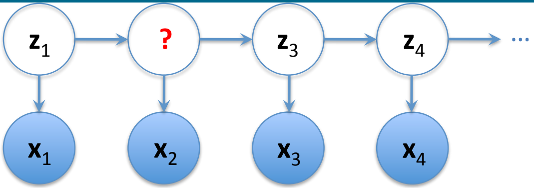


$$
\begin{aligned}
p(z_t, x_{1:T}) &= p(z_t, x_{1:t}, x_{t+1:T}) \\
&= p(z_t, x_{1:t}) p(x_{t+1:T}|z_t, x_{1:t}) \qquad \text{product rule} \\
&= \underbrace{p(z_t, x_{1:t})}_{\alpha(z_t)} \underbrace{p(x_{t+1:T}|z_t)}_{\beta(z_t)} \qquad \text{model independencies}
\end{aligned}
$$

Recursive form for $\alpha(z_t)$ as in filtering. what is $\beta(z_t)$?

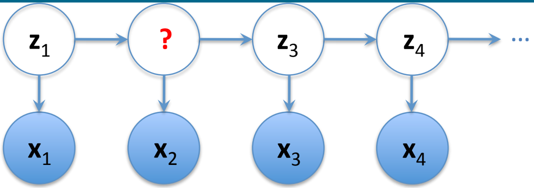
Hidden Markov Models

Smoothing: β recursion



Hidden Markov Models

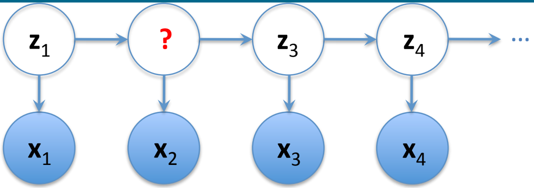
Smoothing: β recursion



$$p(x_{t:T} | z_{t-1}) = \sum_{z_t} p(x_t, x_{t+1:T}, z_t | z_{t-1})$$

Hidden Markov Models

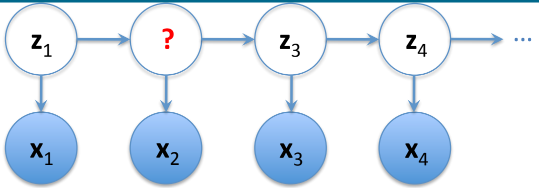
Smoothing: β recursion



$$
\begin{aligned}
p(x_{t:T}|z_{t-1}) &= \sum_{z_t} p(x_t, x_{t+1:T}, z_t | z_{t-1}) \\
&= \sum_{z_t} p(x_t|z_t) \underbrace{p(x_{t+1:T}|z_t)}_{\beta(z_t)} p(z_t|z_{t-1}) \qquad \text{model independencies}
\end{aligned}
$$

Hidden Markov Models

Smoothing: β recursion



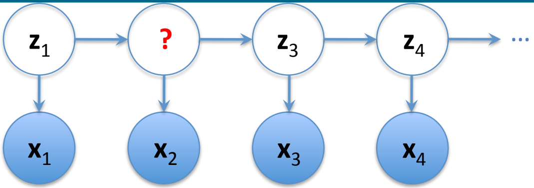
$$p(x_{t:T}|z_{t-1}) = \sum_{z_t} p(x_t, x_{t+1:T}, z_t|z_{t-1})$$

$$= \sum_{z_t} p(x_t|z_t) \underbrace{p(x_{t+1:T}|z_t)}_{\beta(z_t)} p(z_t|z_{t-1}) \quad \text{model independencies}$$

$$\beta(z_{t-1}) = \sum_{z} p(x_t|z_t) p(z_t|z_{t-1}) \beta(z_t) \quad \text{backward recursion!}$$

Hidden Markov Models

Smoothing: β recursion



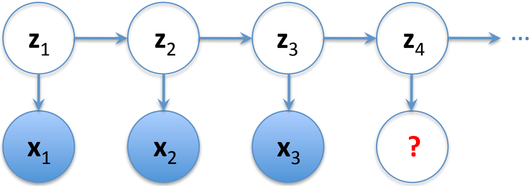
$$p(x_{t:T}|z_{t-1}) = \sum_{z_t} p(x_t, x_{t+1:T}, z_t|z_{t-1})$$

$$= \sum_{z_t} p(x_t|z_t)\underbrace{p(x_{t+1:T}|z_t)}_{\beta(z_t)}p(z_t|z_{t-1}) \qquad \text{model independencies}$$

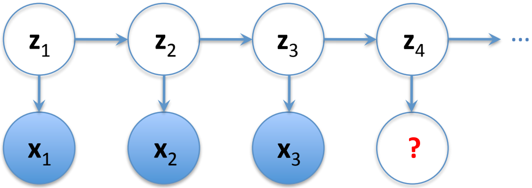
$$\beta(z_{t-1}) = \sum_{z} p(x_t|z_t)p(z_t|z_{t-1})\beta(z_t) \qquad \text{backward recursion!}$$

$\beta(z_T) = 1$. **forward-backward algorithm** (or $\alpha - \beta$ recursions)

Recursions can be performed in parallel



$\mathbf{z}_1$
$\mathbf{z}_2$
$\mathbf{z}_3$
$\mathbf{z}_4$
…
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
?

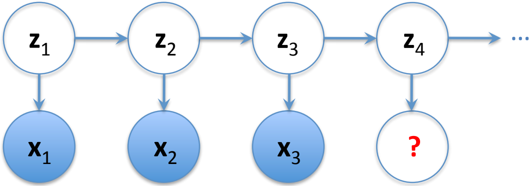


One-step Ahead Prediction:

$$p(x_{t+1}|x_{1:t}) = \sum_{z_t, z_{t+1}} p(x_{t+1}|z_{t+1}) p(z_{t+1}|z_t) \underbrace{p(z_t|x_{1:t})}_{\text{filtering}}$$

Hidden Markov Models

Prediction and Likelihood



One-step Ahead Prediction:

$$p(x_{t+1}|x_{1:t}) = \sum_{z_t, z_{t+1}} p(x_{t+1}|z_{t+1}) p(z_{t+1}|z_t) \underbrace{p(z_t|x_{1:t})}_{\text{filtering}}$$

Likelihood Computation:

$$p(x_{1:T}) = \sum_{z_T} p(z_T, x_{1:T}) = \sum_{z_T} \alpha(z_T)$$

It requires only forward computation (filtering).

The most likely hidden path of $p(z_{1:T}|x_{1:T})$ is the same as the most likely state of $p(z_{1:T}, x_{1:T})$.

The most likely hidden path of $p(z_{1:T}|x_{1:T})$ is the same as the most likely state of $p(z_{1:T}, x_{1:T})$.

Consider maximizing over z_T:

$$
\begin{aligned}
&\max_{z_T} \prod_{t=1}^{T} p(x_t|z_t)p(z_t|z_{t-1}) \\
&= \left(\prod_{t=1}^{T-1} p(x_t|z_t)p(z_t|z_{t-1}) \right) \underbrace{\max_{z_T} p(x_T|z_T)p(z_T|z_{T-1})}_{\mu(z_{T-1})}
\end{aligned}
$$

The most likely hidden path of $p(z_{1:T}|x_{1:T})$ is the same as the most likely state of $p(z_{1:T}, x_{1:T})$.

Consider maximizing over z_T:

$$\max_{z_T} \prod_{t=1}^{T} p(x_t|z_t)p(z_t|z_{t-1})$$
$$= \left(\prod_{t=1}^{T-1} p(x_t|z_t)p(z_t|z_{t-1})\right) \underbrace{\max_{z_T} p(x_T|z_T)p(z_T|z_{T-1})}_{\mu(z_{T-1})}$$

Hence we can define:

$$\mu(z_{t-1}) = \max_{z_t} p(x_t|z_t)p(z_t|z_{t-1})\mu(z_t)$$

for $2 \leqslant t \leqslant T$ and with $\mu(z_T) = 1$.

The information propagated backwards regarding maximizing over $z_2, \dots, z_T$ is contained in $\mu(z_1)$. Therefore:

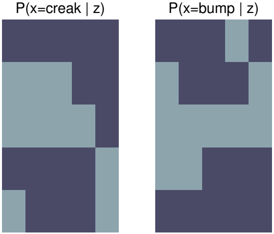
$$z_1^* = \operatorname*{argmax}_{z_1} p(x_1|z_1)p(z_1)\mu(z_1)$$

Then we can compute the others by backtracking:

$$z_t^* = \operatorname*{argmax}_{z_t} p(x_t|z_t)p(z_t|z_{t-1}^*)\mu(z_t)$$

This is a special case of the *max-product* algorithm (in graphical models) and is called **Viterbi Algorithm**.

- You are in bed but have a "mental" partition of the floor as a 5×5 grid
- You know probability of "creak" and "bump" given a position



- Burglar can move only one grid square (left, right, forward, backwards) at time t
- You observe a series of creak/bump information
- Where is the burglar?

Burglar Example (Reproduced from Barber, BRML, 2011): Setting

- Location of the burglar is hidden. Discrete variable $z \in \{1, \ldots, 25\}$
- Absence/presence of creaks and bumps are visible.
- Assume independence and create a new 4-state visible variable using $p(x|z) = p(x^{\text{creak}}|z)p(x^{\text{bump}}|z)$
- How to specify the dynamics?

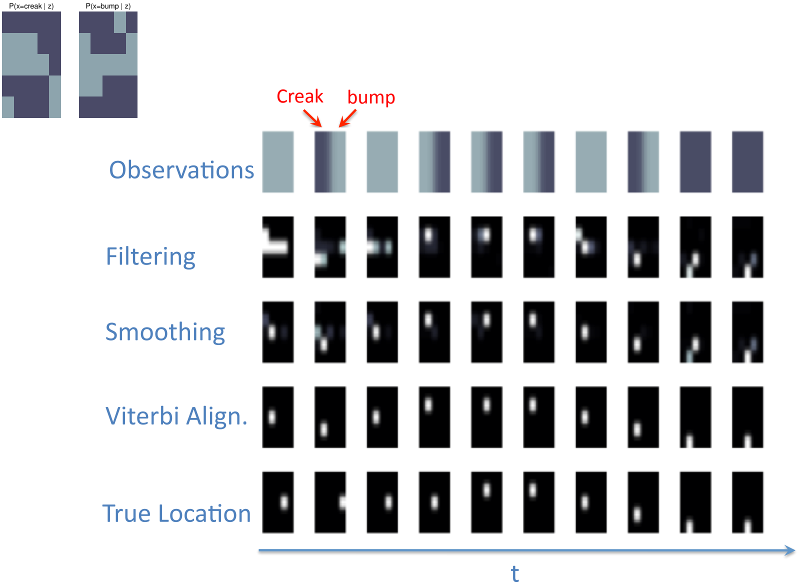
- Location of the burglar is hidden. Discrete variable $z \in \{1, \ldots, 25\}$
- Absence/presence of creaks and bumps are visible.
- Assume independence and create a new 4-state visible variable using $p(x|z) = p(x^{\text{creak}}|z)p(x^{\text{bump}}|z)$
- How to specify the dynamics?

Inference questions:

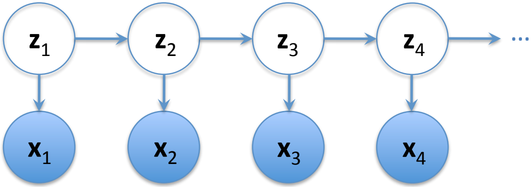
- Where might the burglar be at time t?
- Where could the burglar have been?
  - Important info for the police
- Single best guess for sequence of burglar's positions

HMM Classical Inference Problems

Burglar Example (Reproduced from Barber, BRML, 2011)



Recall our HMM model is given by:



The joint distribution:

$$p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}) = p(\mathbf{z}_1) p(\mathbf{x}_1|\mathbf{z}_1) \prod_{t=2}^{T} p(\mathbf{z}_t|\mathbf{z}_{t-1}) p(\mathbf{x}_t|\mathbf{z}_t)$$

is parameterized by: $A_{ij} = p(z_t = i|z_{t-1} = j)$, $\pi_i = p(z_1 = i)$ and (assuming discrete observations) $B_{ij} = p(x_t = i|z_t = j)$.

How do we learn these parameters from data?

Given a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, S\}$ (assume we know the number of hidden states K)

Goal: learn $\theta = \{\mathbf{A}, \boldsymbol{\pi}, \mathbf{B}\}$

Given a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \ldots, S\}$ (assume we know the number of hidden states K)

Goal: learn $\theta = \{\mathbf{A}, \boldsymbol{\pi}, \mathbf{B}\}$

We can try direct log likelihood maximization:

$$\mathcal{L}(\theta) = \sum_{n=1}^{N} \log \sum_{\mathbf{z}} p(\mathbf{x}_{1:T}^n, z_{1:T}^n | \theta)$$

Given a set of observation sequences $\mathcal{D} = \{x_{1:T}^n\}_{n=1}^N$ where $x_t \in \{1, \dots, S\}$ (assume we know the number of hidden states K)

Goal: learn $\theta = \{\mathbf{A}, \boldsymbol{\pi}, \mathbf{B}\}$

We can try direct log likelihood maximization:

$$\mathcal{L}(\theta) = \sum_{n=1}^{N} \log \sum_{\mathbf{z}} p(\mathbf{x}_{1:T}^n, z_{1:T}^n | \theta)$$

- The distribution does not fully factorize over $t = 1, \dots, T$
- Need to marginalize over latent variables directly
- We would like to get "closed-form" updates

Given a set of observation sequences $\mathcal{D} = \{x^n_{1:T}\}_{n=1}^N$ where $x_t \in \{1, \ldots, S\}$ (assume we know the number of hidden states K)

Goal: learn $\theta = \{\mathbf{A}, \boldsymbol{\pi}, \mathbf{B}\}$

We can try direct log likelihood maximization:

$$\mathcal{L}(\theta) = \sum_{n=1}^{N} \log \sum_{\mathbf{z}} p(\mathbf{x}^n_{1:T}, z^n_{1:T} | \theta)$$

- The distribution does not fully factorize over $t = 1, \ldots, T$
- Need to marginalize over latent variables directly
- We would like to get "closed-form" updates

What can we do?

As in GMMs, we can use the complete data log-likelihood:

$$\mathcal{L}^{\text{comp}}(\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}^n, \mathbf{z}^n | \theta),$$

and iterate:

1. compute its **expectation** over the posterior $\langle \mathcal{L}^{\text{comp}} \rangle_{p(\mathbf{Z}|\mathbf{X},\theta^{\text{old}})}$
2. **Maximize** this expectation wrt θ

In the M-step we need to maximize the objective function:

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \left\langle \sum_{n=1}^{N} \log p(\mathbf{x}^n, \mathbf{z}^n | \boldsymbol{\theta}) \right\rangle_{p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}})}$$

In the M-step we need to maximize the objective function:

$$
\begin{aligned}
\mathcal{Q}(\theta, \theta^{\text{old}}) &= \left\langle \sum_{n=1}^{N} \log p(\mathbf{x}^n, \mathbf{z}^n | \theta) \right\rangle_{p(\mathbf{Z}|\mathbf{X}, \theta^{\text{old}})} \\
&= \sum_{n=1}^{N} \left\langle \log p(z_1^n | \theta) \right\rangle_{p(z_1^n | x_{1:T}^n, \theta^{\text{old}})} \\
&+ \sum_{n=1}^{N} \sum_{t=2}^{T} \left\langle \log p(z_t^n | z_{t-1}^n, \theta) \right\rangle_{p(z_{t-1}^n, z_t^n | x_{1:T}^n, \theta^{\text{old}})} \\
&+ \sum_{n=1}^{N} \sum_{t=1}^{T} \left\langle \log p(x_t^n | z_t^n, \theta) \right\rangle_{p(z_t^n | x_{1:T}^n, \theta^{\text{old}})}
\end{aligned}
$$

In the M-step we need to maximize the objective function:

$$
\begin{aligned}
\mathcal{Q}(\theta, \theta^{\text{old}}) &= \left\langle \sum_{n=1}^{N} \log p(\mathbf{x}^n, \mathbf{z}^n | \theta) \right\rangle_{p(\mathbf{Z}|\mathbf{X}, \theta^{\text{old}})} \\
&= \sum_{n=1}^{N} \langle \log p(z_1^n | \theta) \rangle_{p(z_1^n | x_{1:T}^n, \theta^{\text{old}})} \\
&+ \sum_{n=1}^{N} \sum_{t=2}^{T} \left\langle \log p(z_t^n | z_{t-1}^n, \theta) \right\rangle_{p(z_{t-1}^n, z_t^n | x_{1:T}^n, \theta^{\text{old}})} \\
&+ \sum_{n=1}^{N} \sum_{t=1}^{T} \langle \log p(x_t^n | z_t^n, \theta) \rangle_{p(z_t^n | x_{1:T}^n, \theta^{\text{old}})}
\end{aligned}
$$

wrt $\theta = \{\mathbf{A}, \pi, \mathbf{B}\}$ subject to the usual normalization constraints.

Performing the corresponding derivatives we get the following updates:

$$\pi_i = p(z_1 = i) = \frac{1}{N}\sum_{n=1}^{N} p(z_1^n = i | x_{1:T}^n, \theta^{\text{old}})$$

$$A_{ij} = p(z_t = i | z_{t-1} = j) \propto \sum_{n=1}^{N}\sum_{t=2}^{T} p(z_t^n = i, z_{t-1}^n = j | x_{1:T}^n, \theta^{\text{old}})$$

$$B_{sj} = p(x_t = s | z_t = j) \propto \sum_{n=1}^{N}\sum_{t=1}^{T} \mathbb{I}[x_t^n = s] p(z_t^n = j | x_{1:T}^n, \theta^{\text{old}})$$

Interpretation?

how to deal with different-length sequences?

In the E-step, based on the old parameters, we need to update the distributions:

$$p(z_1 = i | x_{1:T}, \theta^{\text{old}}) \quad \text{and} \quad p(z_t = j | x_{1:T}, \theta^{\text{old}})$$

What classical inference problem are we addressing there?

What about $p(z_t = i, z_{t-1} = j | x_{1:T}, \theta^{\text{old}})$?

- This is a pairwise marginal, which can be shown to be:

$$p(z_t, z_{t-1} | x_{1:T}) \propto \alpha(z_{t-1}) p(x_t | z_t) p(z_t | z_{t-1}) \beta(z_t)$$

In the E-step, based on the old parameters, we need to update the distributions:

$$p(z_1 = i | x_{1:T}, \theta^{\text{old}}) \quad \text{and} \quad p(z_t = j | x_{1:T}, \theta^{\text{old}})$$

What classical inference problem are we addressing there?

What about $p(z_t = i, z_{t-1} = j | x_{1:T}, \theta^{\text{old}})$?

- This is a pairwise marginal, which can be shown to be:

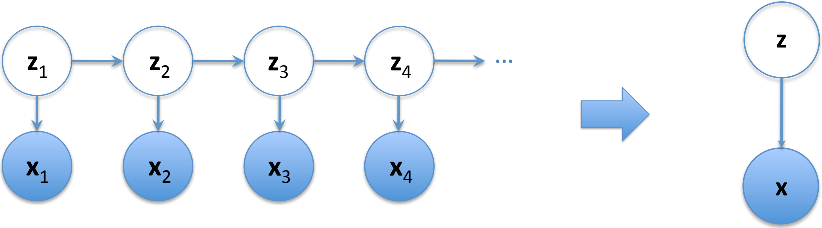
$$p(z_t, z_{t-1} | x_{1:T}) \propto \alpha(z_{t-1}) p(x_t | z_t) p(z_t | z_{t-1}) \beta(z_t)$$

This is an interesting example where learning requires the computation of non-straightforward marginals (inference)

Repeat Until convergence (e.g. using data likelihood):

1. Initialize parameters $\theta^{\text{old}} = \{\mathbf{A}, \boldsymbol{\pi}, \mathbf{B}\}$
2. Run forward-backward recursions to compute corresponding posteriors $p(z_1 = i | x_{1:T}, \theta^{\text{old}})$, $p(z_t = j | x_{1:T}, \theta^{\text{old}})$, $p(z_t = i, z_{t-1} = j | x_{1:T}, \theta^{\text{old}})$
3. Update parameters $\mathbf{A}$, $\boldsymbol{\pi}$, $\mathbf{B}$ using these posteriors accordingly
4. Evaluate likelihood as convergence criterion

- EM is plagued with local optima and a good parameter initialization is needed
- This is specially critical for the emission distribution
- Can initialize $p(x_t|z_t)$ with a simple (non-temporal) mixture model, i.e. $p(x) = \sum_z p(z)p(x|z)$



Except for the specific derivation of learning the emission matrix, everything applies to continuous observations.

- We need $p(\mathbf{x}_t|z_t)$ mapping the discrete state to a continuous distribution over outputs.

Except for the specific derivation of learning the emission matrix, everything applies to continuous observations.

- We need $p(\mathbf{x}_t|z_t)$ mapping the discrete state to a continuous distribution over outputs.
- Gaussian: $p(x_t|z_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_{z_t}, \boldsymbol{\Sigma}_{z_t})$

Except for the specific derivation of learning the emission matrix, everything applies to continuous observations.

- We need $p(\mathbf{x}_t|z_t)$ mapping the discrete state to a continuous distribution over outputs.
- Gaussian: $p(x_t|z_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_{z_t}, \boldsymbol{\Sigma}_{z_t})$
- $\boldsymbol{\mu}_{z_t}$, and $\boldsymbol{\Sigma}_{z_t}$ updates are obtained by weighting the sample mean and covariance wrt the posterior $p(z_t|x_{1:T}, \boldsymbol{\theta}^{\text{old}})$

Except for the specific derivation of learning the emission matrix, everything applies to continuous observations.

- We need $p(\mathbf{x}_t|z_t)$ mapping the discrete state to a continuous distribution over outputs.
- Gaussian: $p(x_t|z_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_{z_t}, \boldsymbol{\Sigma}_{z_t})$
- $\boldsymbol{\mu}_{z_t}$, and $\boldsymbol{\Sigma}_{z_t}$ updates are obtained by weighting the sample mean and covariance wrt the posterior $p(z_t|x_{1:T}, \boldsymbol{\theta}^{\text{old}})$
  - cf updates for B_{sj}

Except for the specific derivation of learning the emission matrix, everything applies to continuous observations.

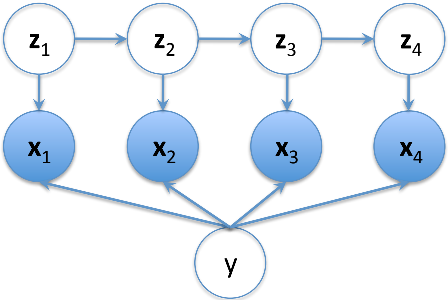
- We need $p(\mathbf{x}_t|z_t)$ mapping the discrete state to a continuous distribution over outputs.
- Gaussian: $p(x_t|z_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_{z_t}, \boldsymbol{\Sigma}_{z_t})$
- $\boldsymbol{\mu}_{z_t}$, and $\boldsymbol{\Sigma}_{z_t}$ updates are obtained by weighting the sample mean and covariance wrt the posterior $p(z_t|x_{1:T}, \boldsymbol{\theta}^{\text{old}})$
  - cf updates for B_{sj}
- The **HMM-GMM** model uses a mixture of Gaussians as the emission distribution

Except for the specific derivation of learning the emission matrix, everything applies to continuous observations.

- We need $p(\mathbf{x}_t|z_t)$ mapping the discrete state to a continuous distribution over outputs.
- Gaussian: $p(x_t|z_t) = \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_{z_t}, \boldsymbol{\Sigma}_{z_t})$
- $\boldsymbol{\mu}_{z_t}$, and $\boldsymbol{\Sigma}_{z_t}$ updates are obtained by weighting the sample mean and covariance wrt the posterior $p(z_t|x_{1:T}, \boldsymbol{\theta}^{\text{old}})$
  - cf updates for B_{sj}
- The **HMM-GMM** model uses a mixture of Gaussians as the emission distribution
  - Popular in tracking and speech recognition

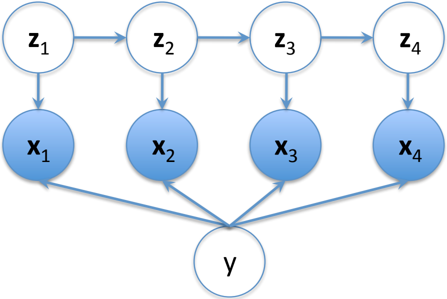
Sequence Classification with HMMs

We want to classify complete sequences based on labeled data $\mathcal{D} = \{(x^n_{1:T_n}, y^n)\}_{n=1}^N$, e.g. $y \in \{\text{swimming, no swimming}\}$



Sequence Classification with HMMs

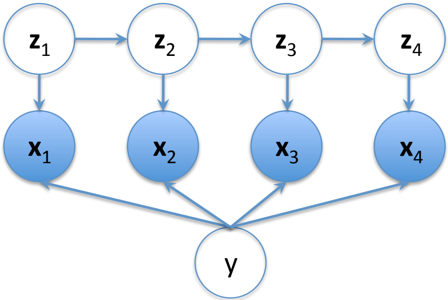
We want to classify complete sequences based on labeled data $\mathcal{D} = \{(x^n_{1:T_n}, y^n)\}_{n=1}^N$, e.g. $y \in \{\text{swimming, no swimming}\}$



We can simply model each conditional likelihood $p(\mathbf{x}_{1:T}|y)$ with an HMM use Bayes rule for classification (*i.e. class conditional approach*)

Sequence Classification with HMMs

We want to classify complete sequences based on labeled data $\mathcal{D} = \{(x^n_{1:T_n}, y^n)\}_{n=1}^N$, e.g. $y \in \{\text{swimming}, \text{no swimming}\}$

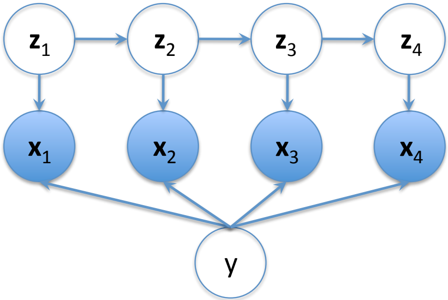


We can simply model each conditional likelihood $p(\mathbf{x}_{1:T}|y)$ with an HMM use Bayes rule for classification (*i.e. class conditional approach*)

- However this is inherently **generative**, any problems?

Sequence Classification with HMMs

We want to classify complete sequences based on labeled data $\mathcal{D} = \{(x^n_{1:T_n}, y^n)\}_{n=1}^{N}$, e.g. $y \in \{\text{swimming, no swimming}\}$



We can simply model each conditional likelihood $p(\mathbf{x}_{1:T}|y)$ with an HMM use Bayes rule for classification (*i.e. class conditional approach*)

- However this is inherently **generative**, any problems?
- In many applications, it is customary to train C HMMs in a **discriminative** way

Discriminative Training of HMMs for Sequence Classification

- Define a new single discriminative model using the C HMMs:

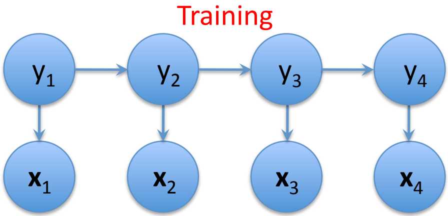
$$p(y|\mathbf{x}_{1:T}) = \frac{p(\mathbf{x}_{1:T}|y)}{\sum_{y'=1}^{C} p(\mathbf{x}_{1:T}|y')p(y')}$$

- Then maximise the likelihood of the classes and corresponding observations $\mathbf{x}_{1:T}$:

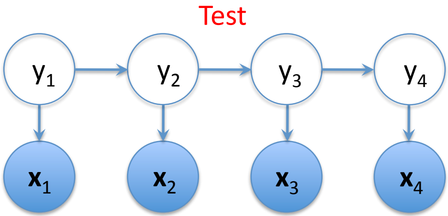
$$\mathcal{L} = \sum_{n=1}^{N} \log p(y^{(n)}|\mathbf{x}_{1:T}^{(n)})$$

- EM-style updates no longer possible
- Learning via gradient-based optimization

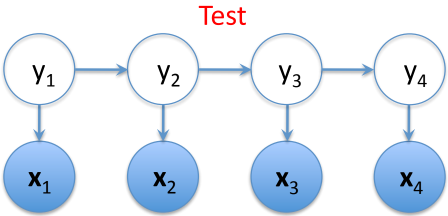
We can also be given time-dependent labels $\mathcal{D} = \{(\mathbf{x}^n_{1:T_n}, y^n_{1:T_n})\}_{n=1}^N$



We can also be given time-dependent labels $\mathcal{D} = \{(\mathbf{x}^n_{1:T_n}, y^n_{1:T_n})\}_{n=1}^N$
Our task is to find the most likely label for a **new** sequence:

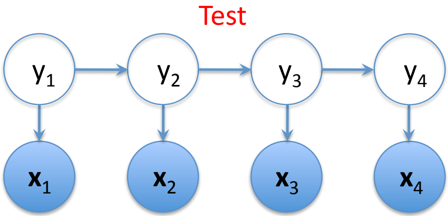


We can also be given time-dependent labels $\mathcal{D} = \{(\mathbf{x}^n_{1:T_n}, y^n_{1:T_n})\}_{n=1}^N$
Our task is to find the most likely label for a **new** sequence:



How would you train this model? How would you make predictions?

We can also be given time-dependent labels $\mathcal{D} = \{(\mathbf{x}^n_{1:T_n}, y^n_{1:T_n})\}_{n=1}^N$
Our task is to find the most likely label for a **new** sequence:



How would you train this model? How would you make predictions?
Alternatively, we can be **discriminative** by realizing that:
$p(x_t|y_t) \propto \tilde{p}(y_t|x_t)\tilde{p}(x_t)$

- Learn transitions and discriminative model $\tilde{p}(y_t|x_t)$ separately and do Viterbi decoding afterwards.

We have seen that standard HMMs are inherently generative.

- We model the joint $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}|\mathbf{y})p(\mathbf{y})$
- We make predictions $p(\mathbf{y}|\mathbf{x})$ using Bayes' rule

If we lack prior information and only care about discriminating between patterns given a set of features:

- We can model $p(\mathbf{y}|\mathbf{x})$ directly.
- This will avoid making unrealistic assumptions about the density of $\mathbf{x}$

Such an approach is adopted by Conditional Random Fields

Summary and Conclusions

- Modelling dependencies in sequential data is essential
- Difficult trade-off flexibility vs complexity in fully observable models
- Use trick of introducing latent variables
- Hidden Markov Models are an elegant way of modelling long-term dependencies in observations
- Reading: Barber (BRML, 2017) Ch 23 (except sec 23.4)