Machine Learning - Lecture 17

Sequential Data & Unifying Perspective I
13.07.2010

Bastian Leibe
RWTH Aachen
http://www.mmp.rwth-aachen.de

leibe@umic.rwth-aachen.de
Course Outline

• Fundamentals (2 weeks)
  - Bayes Decision Theory
  - Probability Density Estimation

• Discriminative Approaches (4 weeks)
  - Lin. Discriminants, SVMs, Boosting

• Generative Models (5 weeks)
  - Bayesian Networks + Applications
  - Markov Random Fields + Applications
  - Exact Inference
  - Approximate Inference

• Unifying Perspective (1 week)
  - HMMs, EM, CRFs, ...

B. Leibe
Topics of This Lecture

- Approximate Inference
  - Variational methods
  - Sampling approaches

- Sampling approaches
  - Sampling from a distribution
  - Ancestral Sampling
  - Rejection Sampling
  - Importance Sampling

- Markov Chain Monte Carlo
  - Markov Chains
  - Metropolis Algorithm
  - Metropolis-Hastings Algorithm
  - Gibbs Sampling
Topics of This Lecture

• Markov Chain Monte Carlo (cont’d)
  ➢ Markov Chains
  ➢ Metropolis Algorithm
  ➢ Metropolis-Hastings Algorithm
  ➢ Gibbs Sampling

• Models for Sequential Data
  ➢ Independence assumptions
  ➢ Markov models

• Hidden Markov Models (HMMs)
  ➢ Graphical Model view
  ➢ Forward-Backward Algorithm
  ➢ Viterbi Algorithm
  ➢ Baum-Welch Algorithm
  ➢ Unifying view
Recap: Sampling Idea

- **Objective:**
  - Evaluate expectation of a function $f(z)$ w.r.t. a probability distribution $p(z)$.
  
  $$\mathbb{E}[f] = \int f(z)p(z) \, dz$$

- **Sampling idea**
  - Draw $L$ independent samples $z^{(l)}$ with $l = 1, \ldots, L$ from $p(z)$.
  - This allows the expectation to be approximated by a finite sum
    $$\hat{f} = \frac{1}{L} \sum_{l=1}^{L} f(z^{(l)})$$
  - As long as the samples $z^{(l)}$ are drawn independently from $p(z)$, then
    $$\mathbb{E}[\hat{f}] = \mathbb{E}[f]$$

  ⇒ **Unbiased estimate, independent of the dimension of $z$!**
Recap: Sampling from a pdf

- In general, assume we are given the pdf $p(x)$ and the corresponding cumulative distribution:

$$F(x) = \int_{-\infty}^{x} p(z) dz$$

- To draw samples from this pdf, we can invert the cumulative distribution function:

$$u \sim Uniform(0, 1) \Rightarrow F^{-1}(u) \sim p(x)$$
Recap: Rejection Sampling

• Assumptions
  - Sampling directly from $p(z)$ is difficult.
  - But we can easily evaluate $p(z)$ (up to some norm. factor $Z_p$):
    $$p(z) = \frac{1}{Z_p} \hat{p}(z)$$

• Idea
  - We need some simpler distribution $q(z)$ (called proposal distribution) from which we can draw samples.
  - Choose a constant $k$ such that: $\forall z : kq(z) \geq \hat{p}(z)$

• Sampling procedure
  - Generate a number $z_0$ from $q(z)$.
  - Generate a number $u_0$ from the uniform distribution over $[0, kq(z_0)]$.
  - If $u_0 > \hat{p}(z_0)$ reject sample, otherwise accept.
Recap: Importance Sampling

- **Approach**
  - Approximate expectations directly (but does **not** enable to draw samples from $p(z)$ directly).
  - Goal:
    $$E[f] = \int f(z)p(z)dz$$

- **Idea**
  - Use a proposal distribution $q(z)$ from which it is easy to sample.
  - Express expectations in the form of a finite sum over samples $\{z^{(l)}\}$ drawn from $q(z)$.
    $$E[f] = \int f(z)p(z)dz = \int f(z)\frac{p(z)}{q(z)}q(z)dz$$
    $$\approx \frac{1}{L} \sum_{l=1}^{L} \frac{p(z^{(l)})}{q(z^{(l)})}f(z^{(l)})$$
    
    **Importance weights**

Image source: C.M. Bishop, 2006
Recap: MCMC - Markov Chain Monte Carlo

• Overview
  - Allows to sample from a large class of distributions.
  - Scales well with the dimensionality of the sample space.

• Idea
  - We maintain a record of the current state \( z^{(\tau)} \)
  - The proposal distribution depends on the current state: \( q(z | z^{(\tau)}) \)
  - The sequence of samples forms a Markov chain \( z^{(1)}, z^{(2)},... \)

• Approach
  - At each time step, we generate a candidate sample from the proposal distribution and accept the sample according to a criterion.
  - Different variants of MCMC for different criteria.
MCMC - Metropolis Algorithm

- **Metropolis algorithm** [Metropolis et al., 1953]
  - Proposal distribution is symmetric: \( q(z_A|z_B) = q(z_B|z_A) \)
  - The new candidate sample \( z^* \) is accepted with probability
    \[
    A(z^*, z^{(\tau)}) = \min \left( 1, \frac{\tilde{p}(z^*)}{\tilde{p}(z^{(\tau)})} \right)
    \]

- **Implementation**
  - Choose random number \( u \) uniformly from unit interval \((0,1)\).
  - Accept sample if \( A(z^*, z^{(\tau)}) > u \).

- **Note**
  - New candidate samples always accepted if \( \tilde{p}(z^*) \geq \tilde{p}(z^{(\tau)}) \).
    - i.e. when new sample has higher probability than the previous one.
  - The algorithm sometimes accepts a state with lower probability.
**MCMC - Metropolis Algorithm**

- **Two cases**
  - If new sample is accepted: \( z^{(\tau+1)} = z^* \)
  - Otherwise: \( z^{(\tau+1)} = z^{(\tau)} \)

  This is in contrast to rejection sampling, where rejected samples are simply discarded.

  ⇒ Leads to multiple copies of the same sample!
MCMC - Metropolis Algorithm

• Property
  - When \( q(z_A | z_B) > 0 \) for all \( z \), the distribution of \( z^\tau \) tends to \( p(z) \) as \( \tau \to \infty \).

• Note
  - Sequence \( z^{(1)}, z^{(2)}, \ldots \) is not a set of independent samples from \( p(z) \), as successive samples are highly correlated.
  - We can obtain (largely) independent samples by just retaining every \( M^{\text{th}} \) sample.

• Example: Sampling from a Gaussian
  - Proposal: Gaussian with \( \sigma = 0.2 \).
  - Green: accepted samples
  - Red: rejected samples

Slide credit: Bernt Schiele

Image source: C.M. Bishop, 2006
Markov Chains

• Question
  - How can we show that $z^\tau$ tends to $p(z)$ as $\tau \to \infty$?

• Markov chains
  - First-order Markov chain:
    \[ p\left(z^{(m+1)}|z^{(1)}, \ldots, z^{(m)}\right) = p\left(z^{(m+1)}|z^{(m)}\right) \]
  - Marginal probability
    \[ p\left(z^{(m+1)}\right) = \sum_{z^{(m)}} p\left(z^{(m+1)}|z^{(m)}\right) p\left(z^{(m)}\right) \]
Markov Chains - Properties

• Invariant distribution
  - A distribution is said to be invariant (or stationary) w.r.t. a Markov chain if each step in the chain leaves that distribution invariant.
  - Transition probabilities:
    \[ T\left(z^{(m)}, z^{(m+1)}\right) = p\left(z^{(m+1)} | z^{(m)}\right) \]
  - Distribution \( p^*(z) \) is invariant if:
    \[ p^*(z) = \sum_{z'} T\left(z', z\right) p^*(z') \]

• Detailed balance
  - Sufficient (but not necessary) condition to ensure that a distribution is invariant:
    \[ p^*(z)T\left(z, z'\right) = p^*(z')T\left(z', z\right) \]
  - A Markov chain which respects detailed balance is reversible.

Slide credit: Bernt Schiele
MCMC - Metropolis-Hastings Algorithm

- **Metropolis-Hastings Algorithm**
  - Generalization: Proposal distribution not required to be symmetric.
  - The new candidate sample $z^*$ is accepted with probability
    \[
    A(z^*, z^{(\tau)}) = \min \left( 1, \frac{\tilde{p}(z^*) q_k(z^{(\tau)}|z^*)}{\tilde{p}(z^{(\tau)}) q_k(z^*|z^{(\tau)})} \right)
    \]
  - where $k$ labels the members of the set of possible transitions considered.

- **Note**
  - When the proposal distributions are symmetric, Metropolis-Hastings reduces to the standard Metropolis algorithm.
MCMC - Metropolis-Hastings Algorithm

• Properties
  - We can show that $p(z)$ is an invariant distribution of the Markov chain defined by the Metropolis-Hastings algorithm.
  - We show detailed balance:

$$p(z)q_k(z|z')A_k(z', z) = \min \{p(z)q_k(z'|z), p(z')q_k(z'|z)\}$$

$$= \min \{p(z')q_k(z'|z), p(z)q_k(z|z')\}$$

$$= p(z')q_k(z'|z)A_k(z, z')$$
MCMC - Metropolis-Hastings Algorithm

- Schematic illustration
  - For continuous state spaces, a common choice of proposal distribution is a Gaussian centered on the current state.
  - What should be the variance of the proposal distribution?
    - Large variance: rejection rate will be high for complex problems.
    - The scale $\rho$ of the proposal distribution should be as large as possible without incurring high rejection rates.
      - $\rho$ should be of the same order as the smallest length scale $\sigma_{\text{min}}$.
  - This causes the system to explore the distribution by means of a random walk.
    - Undesired behavior: number of steps to arrive at state that is independent of original state is of order $(\sigma_{\text{max}} / \sigma_{\text{min}})^2$.
    - Strong correlations can slow down the Metropolis algorithm!

B. Leibe
Gibbs Sampling

- **Approach**
  - MCMC-algorithm that is simple and widely applicable.
  - May be seen as a special case of Metropolis-Hastings.

- **Idea**
  - Sample variable-wise: replace $z_i$ by a value drawn from the distribution $p(z_i | z\setminus i)$.
    - This means we update one coordinate at a time.
  - Repeat procedure either by cycling through all variables or by choosing the next variable.
Gibbs Sampling

- **Example**
  - Assume distribution $p(z_1, z_2, z_3)$.
  - Replace $z_1^{(\tau)}$ with new value drawn from $z_1^{(\tau+1)} \sim p(z_1|z_2^{(\tau)}, z_3^{(\tau)})$.
  - Replace $z_2^{(\tau)}$ with new value drawn from $z_2^{(\tau+1)} \sim p(z_2|z_1^{(\tau+1)}, z_3^{(\tau)})$.
  - Replace $z_3^{(\tau)}$ with new value drawn from $z_3^{(\tau+1)} \sim p(z_3|z_1^{(\tau+1)}, z_2^{(\tau+1)})$.
  - And so on...
Gibbs Sampling

- Properties
  - The factor that determines the acceptance probability in the Metropolis-Hastings is determined by

\[
A(z^*, z) = \frac{p(z^*) q_k(z|z^*)}{p(z) q_k(z^*|z)} = \frac{p(z_k^*|z_{\setminus k}^*) p(z_{\setminus k}^*) p(z_k^*|z_{\setminus k})}{p(z_k|z_{\setminus k}) p(z_{\setminus k}) p(z_k|z_{\setminus k})} = 1
\]

  - I.e. we get an algorithm which always accepts!

- If you can compute (and sample from) the conditionals, you can apply Gibbs sampling.
  - The algorithm is completely parameter free.
  - Can also be applied to subsets of variables.
Gibbs Sampling

- **Example**
  - 20 iterations of Gibbs sampling on a bivariate Gaussian.

- **Note:** strong correlations can slow down Gibbs sampling.
Summary: Approximate Inference

• Exact Bayesian Inference often intractable.

• Rejection and Importance Sampling
  ➢ Generate independent samples.
  ➢ Impractical in high-dimensional state spaces.

• Markov Chain Monte Carlo (MCMC)
  ➢ Simple & effective (even though typically computationally expensive).
  ➢ Scales well with the dimensionality of the state space.
  ➢ Issues of convergence have to be considered carefully.

• Gibbs Sampling
  ➢ Used extensively in practice.
  ➢ Parameter free
  ➢ Requires sampling conditional distributions.
References and Further Reading

• Sampling methods for approximate inference are described in detail in Chapter 11 of Bishop’s book.

  Christopher M. Bishop  
  Pattern Recognition and Machine Learning  
  Springer, 2006

• Another good introduction to Monte Carlo methods can be found in Chapter 29 of MacKay’s book (also available online: http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html)
Topics of This Lecture

- Markov Chain Monte Carlo (cont’d)
  - Markov Chains
  - Metropolis Algorithm
  - Metropolis-Hastings Algorithm
  - Gibbs Sampling

- Models for Sequential Data
  - Independence assumptions
  - Markov models

- Hidden Markov Models (HMMs)
  - Graphical Model view
  - Forward-Backward Algorithm
  - Viterbi Algorithm
  - Baum-Welch Algorithm
  - Unifying view
Sequential Data

- Many real-world problems involve sequential data
  - Speech recognition
  - Visual object tracking
  - Robot planning
  - DNA sequencing
  - Financial forecasting
  - ...

- In the following, we will look at sequential problems from a Graphical Models perspective...

B. Leibe
Models for Sequential Data

- Simplest model
  - Treat all observations as independent (i.i.d.)
  - Corresponding graphical model:

- What can we infer from such a model?
  ⇒ Only relative frequencies of certain events.
  ⇒ Such a model is of limited use in practice.
  ⇒ In practice, the data often exhibits trends that help prediction!
Markov Models

- Markov assumption
  - Each observation only depends on the most recent previous observation:
  \[
p(x_1, \ldots, x_N) = \prod_{n=1}^{N} p(x_n | x_1, \ldots, x_{n-1}) = p(x_1) \prod_{n=2}^{N} p(x_n | x_{n-1})
  \]

- First-order Markov chain:
Markov Models

- Still quite restrictive...
  - We often want to model longer-term trends over several successive sequential observations.

- Generalization: Second-order Markov chain

\[
p(x_1, \ldots, x_N) = p(x_1)p(x_2|x_1) \prod_{n=3}^{N} p(x_n|x_{n-1}, x_{n-2})
\]
Markov Models

- We can generalize this further
  - $M$th order Markov chains
  - However, this does not scale well.

  - Suppose all $x_n$ can take on $K$ possible values.
  - #Parameters in the model:
    \[ K^{M-1}(K - 1) \]
    \[ \Rightarrow \text{Exponential complexity!} \]

- Goal
  - We want a model that is not as limited by the Markov assumption
  - But that can be specified by few parameters
  - We can achieve that by introducing a state space model.
Topics of This Lecture

- Markov Chain Monte Carlo (cont’d)
  - Markov Chains
  - Metropolis Algorithm
  - Metropolis-Hastings Algorithm
  - Gibbs Sampling

- Models for Sequential Data
  - Independence assumptions
  - Markov models

- Hidden Markov Models (HMMs)
  - Graphical Model view
  - Forward-Backward Algorithm
  - Viterbi Algorithm
  - Baum-Welch Algorithm
  - Unifying view
Hidden Markov Models (HMMs)

- **Traditional view**
  - The system is at each time in a certain state $k$
    \[ k \in \{1, 2, 3\} \]
  - The (Markovian) state transition probabilities are given by the matrix $A$
    \[
    A = \begin{bmatrix}
    A_{11} & A_{12} & A_{13} \\
    A_{21} & A_{22} & A_{23} \\
    A_{31} & A_{32} & A_{33}
    \end{bmatrix}
    \]
  - We cannot observe the states directly, they are hidden.
  - We just know the initial probability distribution over states $\pi$.
  - Each state produces a characteristic output, given by a probability distribution over output symbols $\phi$. 

B. Leibe
Hidden Markov Models (HMMs)

- **HMMs**
  - Widely used in speech recognition, natural language modelling, handwriting recognition, biological sequence analysis (DNA, proteins), financial forecasting,...
  - Really changed the field...

- Often used in special forms
  - E.g., *left-to-right HMM*

  \[
  A = \begin{bmatrix}
  A_{11} & A_{12} & A_{13} \\
  0 & A_{22} & A_{23} \\
  0 & 0 & A_{33}
  \end{bmatrix}
  \]

- How can we encode them as graphical models?
Hidden Markov Models (HMMs)

- **Graphical Model view**
  - Introduce latent variables $z$ for the current system state.
  - The observed output $x$ is conditioned on this state.

  ![Graphical Model Diagram]

  - The state transition probabilities $p(z_n | z_{n-1})$ are given by the entries of $A$:

    $$p(z_{nk} = 1 | z_{n-1}, j = 1) = A_{jk}$$
Hidden Markov Models (HMMs)

- State transitions
**Interpretation as a Generative Model**

- **Ancestral Sampling from an HMM**
  - Choose initial latent variable $z_1$ according to $\pi_k$
  - Sample the corresponding observation $x_1$
  - Choose state of variable $z_2$ by sampling from $p(z_2 | z_1)$
  - ...

B. Leibe
Three Main Tasks in HMMs

1. Likelihood Estimation
   - Given: an observation sequence and a model
   - What is the likelihood of this sequence given the model?

   “Forward-backward algorithm”

2. Finding the most probable state sequence
   - Given: an observation sequence and a model
   - What is the most likely sequence of states?

   “Viterbi algorithm”

3. Learning HMMs
   - Given: several observation sequences
   - How can we learn/adapt the model parameters?

   “Baum-Welch algorithm”
1. Likelihood Estimation in HMMs

- Problem definition
  - Given: HMM model $\theta = (A, \phi, \pi)$
    - sequence of observations $X = \{x_1, \ldots, x_T\}$
  - Goal: Compute the likelihood $p(X | \theta)$

- Problem: We don’t know the state sequence
  - Naïve approach: marginalize over all possible sequences
    $$ p(X|\theta) = \sum_{\{z_1, \ldots, z_T\}} p(X|z_1, \ldots, z_T, \theta)p(z_1, \ldots, z_T|\theta) $$
  - Effort: $O(K^TT)$
1. Likelihood Estimation in HMMs

- More efficient procedure
  - Make use of the factorization property

\[
p(X|\theta) = \sum_{z_n} p(x_1, \ldots, x_N, z_n) \\
= \sum_{z_n} p(x_1, \ldots, x_N|z_n)p(z_n) \\
= \sum_{z_n} p(x_1, \ldots, x_n|z_n)p(x_{n+1}, \ldots, x_N|z_n)p(z_n) \\
= \sum_{z_n} p(x_1, \ldots, x_n, z_n)p(x_{n+1}, \ldots, x_N|z_n) \\
= \sum_{z_n} \alpha(z_n)\beta(z_n)
\]

- Divide into “past” factor \(\alpha\) and “future” factor \(\beta\)
Forward Pass

- More efficient procedure
  - Joint probability of observing \( x_1, \ldots, x_n \) and ending up in state \( z_n \):
    \[
    \alpha(z_n) = p(x_1, \ldots, x_n|z_n)p(z_n)
    = p(x_n|z_n)p(x_1, \ldots, x_{n-1}|z_n)p(z_n)
    = p(x_n|z_n) \sum_{z_{n-1}} p(x_1, \ldots, x_{n-1}, z_{n-1}|z_n)p(z_n)
    = p(x_n|z_n) \sum_{z_{n-1}} p(z_{n}|z_{n-1})p(x_1, \ldots, x_{n-1}|z_{n-1})p(z_{n-1})
    = p(x_n|z_n) \sum_{z_{n-1}} p(z_{n}|z_{n-1})\alpha(z_{n-1})
    \]
  - Computational effort: \( \mathcal{O}(K^2T) \)

B. Leibe

Image source: C.M. Bishop, 2006
Backward Pass

- Inverse procedure
  - Joint probability of observing $x_{n+1}, \ldots, x_N$ starting from state $z_n$:
    \[
    \beta(z_n) = p(x_{n+1}, \ldots, x_N | z_n) = \sum_{z_{n+1}} p(x_{n+1}, \ldots, x_N, z_{n+1} | z_n)
    \]
    \[
    = \sum_{z_{n+1}} p(x_{n+1}, \ldots, x_N | z_{n+1}) p(z_{n+1} | z_n)
    \]
    \[
    = \sum_{z_{n+1}} p(x_{n+1} | z_{n+1}) p(x_{n+2}, \ldots, x_N | z_{n+1}) p(z_{n+1} | z_n)
    \]
    \[
    = \sum_{z_{n+1}} p(x_{n+1} | z_{n+1}) \beta(z_{n+1}) p(z_{n+1} | z_n)
    \]
  - Computational effort: $O(K^2 T)$
Forward-Backward Algorithm

• Initialization

\[ \alpha(z_1) = \pi p(x_1 | \phi) \]
\[ \beta(z_N) = 1 \]

• Evaluate the likelihood
  
  ➢ Now, we can compute

\[ p(X | \theta) = \sum_{z_n} \alpha(z_n) \beta(z_n) \]

  ➢ In particular

\[ p(X | \theta) = \sum_{z_N} \alpha(z_N) \]
Interpreting the Result

- Computing the likelihood
  - Obtaining $p(X)$ means taking the joint distribution $p(X, Z)$ and summing over all possible values of $Z$.
  - This means summing over all possible paths through the lattice diagram.
  - There are exponentially many such paths.
  - By expressing the likelihood as $p(X) = \sum_{z_N} \alpha(z_N)$ we have reduced the cost to a linear number of paths by swapping the order of sums and multiplications.
- But wait... haven’t we seen this trick before?
1. Likelihood Estimation in HMMs Revisited

- Graphical Model View
  - We have seen that we can write the HMM as a graphical model with latent variables $z_n$:

  ![Graphical Model](image)

  - What we now want to compute are the marginals of $X$.

  $$p(X) = \sum_Z p(X, Z)$$

  ⇒ Use the **Sum-Product algorithm**!
Sum-Product Algorithm for HMMs

- Preparation
  - Begin by transforming the HMM into a factor graph

\[
\begin{align*}
\chi(z_1) &= p(z_1) \\
\psi_n(z_{n-1}, z_n) &= p(z_n | z_{n-1}) \\
g_n(x_n, z_n) &= p(x_n | z_n)
\end{align*}
\]

- factors: $\chi(z_1) = p(z_1)$, $\psi_n(z_{n-1}, z_n) = p(z_n | z_{n-1})$, $g_n(x_n, z_n) = p(x_n | z_n)$
Sum-Product Algorithm for HMMs

- Preparation
  - Since we always condition on $x_1, \ldots, x_N$, we can simplify this:

$$
\begin{align*}
    h(z_1) &= p(z_1) p(x_1 | z_1) \\
    f_n(z_{n-1}, z_n) &= p(z_n | z_{n-1}) p(x_n | z_n)
\end{align*}
$$

- New factors:
  - Set $z_N$ as the root node and pass messages from the leaf node $h$ to the root.

B. Leibe
Sum-Product Messages

\[
\begin{align*}
\mu_{z_{n-1} \rightarrow f_n}(z_{n-1}) &= \mu_{f_{n-1} \rightarrow z_{n-1}}(z_{n-1}) \\
\mu_{f_n \rightarrow z_n}(z_n) &= \sum_{z_{n-1}} f_n(z_{n-1}, z_n) \mu_{z_{n-1} \rightarrow f_n}(z_{n-1})
\end{align*}
\]

Message definitions:

\[
\begin{align*}
\mu_{f_s \rightarrow x}(x) &= \sum_{X_s} f_s(x_s) \prod_{m \in \text{ne}(f_s) \setminus x} \mu_{x_m \rightarrow f_s}(x_m) \\
\mu_{x_m \rightarrow f_s}(x_m) &= \prod_{l \in \text{ne}(x_m) \setminus f_s} \mu_{f_l \rightarrow x_m}(x_m)
\end{align*}
\]

\[
\begin{align*}
\mu_{x \rightarrow f}(x) &= 1 & \mu_{f \rightarrow x}(x) &= f(x)
\end{align*}
\]
Sum-Product Messages

\[
f_n(z_{n-1}, z_n) = p(z_n|z_{n-1})p(x_n|z_n)
\]

\[
\mu_{z_{n-1} \rightarrow f_n(z_{n-1})} = \mu_{f_{n-1} \rightarrow z_{n-1}}(z_{n-1})
\]

\[
\mu_{f_n \rightarrow z_n}(z_n) = \sum_{z_{n-1}} f_n(z_{n-1}, z_n)\mu_{z_{n-1} \rightarrow f_n}(z_{n-1})
\]

\[
= \sum_{z_{n-1}} f_n(z_{n-1}, z_n)\mu_{f_n \rightarrow z_{n-1}}(z_{n-1})
\]

\[
= p(x_n|z_n) \sum_{z_{n-1}} p(z_n|z_{n-1})\mu_{f_n \rightarrow z_{n-1}}(z_{n-1})
\]

**Comparison:**

\[
\alpha(z_n) = p(x_n|z_n) \sum_{z_{n-1}} p(z_n|z_{n-1})\alpha(z_{n-1})
\]
Sum-Product Messages

• Initialization

\[ \mu_{h \rightarrow z_1}(z_1) = h(z_1) = p(z_1)p(x_1|z_1) \]

\[ \alpha(z_1) = \pi p(x_1|\phi) = p(z_1)p(x_1|z_1) \]

⇒ The results are equivalent!

⇒ Forward-backward algorithm is a special case of Sum-Product!
Three Main Tasks in HMMs

1. Likelihood Estimation
   - Given: an observation sequence and a model
   - What is the likelihood of this sequence given the model?
     "Forward-backward algorithm"
     ⇒ special case of Sum-Product!

2. Finding the most probable state sequence
   - Given: an observation sequence and a model
   - What is the most likely sequence of states?
     "Viterbi algorithm"

3. Learning HMMs
   - Given: several observation sequences
   - How can we learn/adapt the model parameters?
     "Baum-Welch algorithm"
2. Finding the Most Probable State Sequence

- **Problem definition**
  - Given: HMM model $\theta = (A, \phi, \pi)$
  - Sequence of observations $X = \{x_1, \ldots, x_T\}$
  - Goal: Compute the most probable sequence of states
    
    $$w(Z) = \max_Z p(X, Z)$$
    
    $$= \max_{z_1, \ldots, z_N} p(x_1, \ldots, x_N, z_1, \ldots, z_N)$$

- **Wait... Doesn’t this also look familiar?**
  - Yes! We can simply apply the *Max-Sum algorithm* here.
Three Main Tasks in HMMs

1. Likelihood Estimation
   - Given: an observation sequence and a model
   - What is the likelihood of this sequence given the model?
     "Forward-backward algorithm"
     \[ \Rightarrow \text{special case of Sum-Product!} \]

2. Finding the most probable state sequence
   - Given: an observation sequence and a model
   - What is the most likely sequence of states?
     "Viterbi algorithm"
     \[ \Rightarrow \text{special case of Max-Sum!} \]

3. Learning HMMs
   - Given: several observation sequences
   - How can we learn/adapt the model parameters?
     "Baum-Welch algorithm"
     application of EM!
     \[ \Rightarrow \text{next lecture} \]
References and Further Reading

- HMMs and their interpretation as graphical models are described in detail in Chapter 13 of Bishop’s book.

Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006