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
  • Dec. Trees, Random Forests, Model Sel.
• Generative Models (5 weeks)
  • Bayesian Networks + Applications
  • Markov Random Fields + Applications
  • Exact Inference
  • Approximate Inference
• Unifying Perspective (1 week)
  • HMMs, EM, CRFs, ...

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

Recap: Sampling Idea

• Objective:
  • Evaluate expectation of a function \( f(a) \) w.r.t. a probability distribution \( p(a) \).
  \[ \mathbb{E}[f] = \int f(a)p(a)\,da \]
• Sampling idea
  • Draw \( L \) independent samples \( a^{(l)} \) with \( l = 1, \ldots, L \) from \( p(a) \).
  • This allows the expectation to be approximated by a finite sum
  \[ f = \frac{1}{L} \sum_{\ell=1}^{L} f(a^{(\ell)}) \]
  • As long as the samples \( a^{(l)} \) are drawn independently from \( p(a) \), then
  \[ \mathbb{E}[f] = \mathbb{E}[f] \]
  ⇒ Unbiased estimate, independent of the dimension of \( a \)!

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

- Assumptions
  - Sampling directly from \( p(z) \) is difficult.
  - But we can easily evaluate \( p(z) \) up to some norm. factor \( Z_k \):
    \[
    p(z) = \frac{1}{Z_k} q(z)
    \]
- Idea
  - We need some simpler distribution \( q(z) \) (called proposal distribution) from which we can draw samples.
  - Choose a constant \( \tau \) such that:
    \[
    \forall \gamma : kq(\gamma) \geq p(\gamma)
    \]
- Sampling procedure
  - Generate a number \( z \), from \( q(z) \).
  - Generate a number \( u \), from the uniform distribution over \( [0,kq(z)] \).
  - If \( u < p(\gamma) \) accept sample, otherwise reject.

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 \( z^{(t)} \).
  - The proposal distribution depends on the current state: \( q(z | z^{(t)} ) \)
  - The sequence of samples forms a Markov chain \( z^{(t)}, z^{(t+1)}, \ldots \)
- 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.

Recap: Importance Sampling

- Approach
  - Approximate expectations directly (but does not enable to draw samples from \( p(z) \) directly).
  - Goal:
    \[
    \mathbb{E}[f] = \int f(z)p(z)dz
    \]
- 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 \( q(z) \):
    \[
    \mathbb{E}[f] = \frac{1}{L} \sum_{l=1}^{L} p(z^{(l)}) q(z^{(l)})
    \]
  - At each time step, we generate a candidate sample from the proposal distribution and accept the sample according to a criterion.
    \[
    u \sim q(z^{(l)})
    \]
    \[
    \]
Markov Chains

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

- Markov chains
  - First-order Markov chain:
    \[ p(z^{(n+1)} | z^{(1)}, \ldots, z^{(n)}) = p(z^{(n+1)} | z^{(n)}) \]
  - Marginal probability
    \[ p(z^{(n+1)}) = \sum_{z^{(n)}} p(z^{(n+1)} | z^{(n)}) p(z^{(n)}) \]

MCMC – Metropolis-Hastings Algorithm

- Metropolis-Hastings Algorithm
  - Generalization: Proposal distribution not required to be symmetric.
  - The new candidate sample $z'$ is accepted with probability
    \[ A(z' | z^*) = \min \left( 1, \frac{p(z') q(z^*)}{p(z^*) q(z')} \right) \]
  - where $i$ 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.

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 $x_i$ by a value drawn from the distribution $p(z_i | x_{-i})$.
    - This means we update one coordinate at a time.
  - Repeat procedure either by cycling through all variables or by choosing the next variable.

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:
      \[ T(z^{(n)}, z^{(n+1)}) = p(z^{(n+1)} | z^{(n)}) \]
    - Distribution $p(z)$ is invariant if:
      \[ p(z) = \sum_{z'} T(z', z) p(z') \]

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

MCMC – Metropolis-Hastings Algorithm

- Properties
  - We can show that $p(x)$ is an invariant distribution of the Markov chain defined by the Metropolis-Hastings algorithm.
  - We show detailed balance:
    \[ p(x) q(x' | x) A_i(x', x) = \min \left( p(x) q(x' | x), p(x') q(x | x') \right) \]
    \[ = \min \left( p(x) q(x' | x), p(x') q(x | x') \right) \]
    \[ = p(x') q(x | x') A_i(x', x) \]
### Gibbs Sampling

**Example**
- Assume distribution $p(z_1, z_2, z_3)$.
  - Replace $z_i^{(t)}$ with new value drawn from $z_i^{(t+1)} \sim p(z_i|z_{\overline{i}}^{(t)})$.
  - And so on...

**Properties**
- The factor that determines the acceptance probability in the Metropolis-Hastings algorithm is:
  $$A(z^*, z) = \frac{p(z^*)q(z|z^*)}{p(z|z^*)q(z^*|z)} = \frac{1}{p(z|z^*)q(z^*|z)}$$
- 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.

**References and Further Reading**
- Sampling methods for approximate inference are described in detail in Chapter 11 of Bishop’s book.
- 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](http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html))
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...

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

- First-order Markov chain:

- 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.
    - Number of parameters in the model:
      \[ K^{M-1}(K - 1) \]
      ⇒ 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)
  - Metropolis Chain
  - Metropolis-Hastings Algorithm
  - Gibbs Sampling
  - Baum-Welch Algorithm
  - Unifying view

- 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 \( k \)
  - 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 \( \phi \).

- **Graphical Model view**
  - Introduce latent variables \( z \) for the current system state.
  - The observed output \( x \) is conditioned on this state.
  - The state transition probabilities \( p(z_n | z_{n-1}) \) are given by the entries of \( A \):
    \[ p(z_{nk} = 1 | z_{n-1}, j = 1) = A_{jk} \]

- **Interpretation as a Generative Model**
  - Ancestral Sampling from an HMM
    - Choose initial latent variable \( x_1 \) according to \( \pi \)
    - Sample the corresponding observation \( x_1 \)
    - Choose state of variable \( x \) by sampling from \( p(x | z) \)

- **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 $\theta = (A, B, \pi)$
  - 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
    $p(X|\theta) = \sum_{z_1, \ldots, z_T} p(X, z_1, \ldots, z_T | \theta)$
  - Effort: $O(K^{2T})$

**Forward Pass**

- **More efficient procedure**
  - Joint probability of observing $x_1, \ldots, x_N$ and ending up in state $z_N$:
    $\alpha(n) = p(x_1, \ldots, x_n, z_n)$
    $= p(x_n | z_n) \sum_{z_{n-1}} p(z_{n-1}, \ldots, z_1, x_1 | \theta)$
    $= p(x_n | z_n) \sum_{z_{n-1}} p(z_{n-1} | z_{n-2}, \ldots, z_1, x_1 | \theta) p(z_{n-2}, \ldots, z_1, x_1 | \theta)$
    $= p(x_n | z_n) \sum_{z_{n-1}} p(z_{n-1} | z_{n-2}, \ldots, z_1, x_1 | \theta) \prod_{i=1}^{n-1} p(x_i | z_i)$
  - Computational effort: $O(K^2T)$

**Backward Pass**

- **Inverse procedure**
  - Joint probability of observing $x_N, \ldots, x_1$ starting from state $z_1$:
    $\beta(n) = p(x_N, \ldots, x_1 | z_1)$
    $= \sum_{z_{N-1}} p(x_N, \ldots, x_1 | z_N, z_{N-1}) p(z_{N-1} | z_1)$
    $= \sum_{z_{N-1}} p(x_N, \ldots, x_1 | z_N, z_{N-1}) p(z_{N-1} | z_1) p(z_N | z_{N-1}) p(z_{N-1} | \theta)$
    $= \sum_{z_{N-1}} p(x_N | z_N, z_{N-1}) p(z_{N-1} | z_1) p(z_N | z_{N-1}) p(z_{N-1} | \theta)$
    $= \sum_{z_{N-1}} p(x_N | z_N, z_{N-1}) p(z_{N-1} | z_1) p(z_{N-1} | \theta)$
  - Computational effort: $O(K^2T)$

**Forward-Backward Algorithm**

- **Initialization**
  - $\alpha(z_1) = p(x_1 | \alpha)$
  - $\beta(z_N) = 1$

- **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 $x_t$.
  - 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
  - Since we always condition on $x_{1:t}$, we can simplify this:
    \[ h(x_1) = p(x_1) \frac{p(x_1 | x_1)}{} \]
    \[ f_n(x_{n-1}, x_n) = p(x_{n-1} | x_n) \frac{p(x_{n-1} | x_n)}{} \]
  - New factors:
    \[ h(x_1) = p(x_1) \frac{p(x_1 | x_1)}{} \]
    \[ f_n(x_{n-1}, x_n) = p(x_{n-1} | x_n) \frac{p(x_{n-1} | x_n)}{} \]
  - Apply Sum-Product algorithm
    - Set $z_0$ as the root node and pass messages from the leaf node $h$ to the root.

Sum-Product Messages

\[ f_n(x_{n-1}, x_n) = \prod_{l \in \text{set}(f_n)} f_l(x_{n-1}, x_n) \]

\[ \mu_{x_{n-1} \rightarrow f_n(x_{n-1})} = \frac{f_n(x_{n-1}, x_n)}{\sum_{x_{n-1}} f_n(x_{n-1}, x_n)} \]

\[ \mu_f(x_n) = \prod_{l \in \text{set}(f_n)} \mu_{x_{n-1} \rightarrow f_n(x_{n-1})} \]

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

Comparison:
\[ \alpha(x_n) = p(x_n) \sum_{x_{n-1}} \alpha(x_{n-1}) \]

Sum-Product Messages

- Initialization
  \[ h(x_1) = p(x_1) \]
  \[ p(x_1 | x_1) \]
  \[ \alpha(x_1) = p(x_1 | x_1) \]
  ⇒ The results are equivalent!
  ⇒ Forward-backward algorithm is a special case of Sum-Product!

Machine Learning, Summer'10

Image source: C.M. Bishop, 2006
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 \( \theta = (A, \pi, \gamma) \)
  - sequence of observations \( X = \{ \mathbf{x}_1, \ldots, \mathbf{x}_T \} \)
  - Goal: Compute the most probable sequence of states

\[
\begin{align*}
\w(Z) &= \max_{Z} \mathbb{P}(X, Z) \\
      &= \max_{z_1, \ldots, z_T} \mathbb{P}(X_1, \ldots, X_T, z_1, \ldots, z_T)
\end{align*}
\]

- Wait... Doesn’t this also look familiar?
  - Yes! We can simply apply the Max-Sum algorithm here.

References and Further Reading

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