Hidden Markov Models
Terminology, Representation and Basic Problems
The next two weeks

Hidden Markov models (HMMs):

Wed 31/10: Terminology and basic algorithms.
Fri 2/11: Implementing the basic algorithms.
Wed 7/11: Implementing the basic algorithms, cont.
Selecting model parameters and training.
Fri 9/11: Selecting model parameters and training, cont.
Wed 14/11: Introduction to mandatory project.
Fri 16/11: Extensions and applications.

We use Chapter 13 from Bishop's book “Pattern Recognition and Machine Learning”. Rabiner's paper “A Tutorial on Hidden Markov Models [...]” might also be useful to read.

Blackboard and http://cs.au.dk/~cstorm/courses/ML_e18
What is machine learning?

Machine learning means different things to different people, and there is no general agreed upon core set of algorithms that must be learned.

For me, the core of machine learning is:

**Building a mathematical model** that captures some desired structure of the data that you are working on.

**Training the model** (i.e. set the parameters of the model) based on existing data to optimize it as well as we can.

**Making predictions** by using the model on new data.
A sequence of observations from a finite and discrete set, e.g. measurements of weather patterns, daily values of stocks, the composition of DNA or proteins, or ...

\[ X = x_1, x_2, \ldots, x_N \]

**Typical question/problem:** How likely is a given \( X \), i.e. \( p(X) \)?

We need a model that describes how to compute \( p(X) \)
Simple Models (1)

Observations are independent and identically distributed

\[ p(x_1, \ldots, x_N) = \prod_{n=1}^{N} p(x_n) \]

Too simplistic for realistic modelling of many phenomena
The $n$'th observation in a chain of observations is influenced only by the $n-1$'th observation, i.e.

$$p(x_n | x_1, \ldots, x_{n-1}) = p(x_n | x_{n-1})$$

The chain of observations is a 1st-order Markov chain, and the probability of a sequence of $N$ observations is

$$p(x_1, \ldots, x_N) = \prod_{n=1}^{N} p(x_n | x_1, \ldots, x_{n-1}) = p(x_1) \prod_{n=2}^{N} p(x_n | x_{n-1})$$
The model, i.e. \( p(x_n \mid x_{n-1}) \): A sequence of observations:

The \( n \)'th observation in a chain of observations is influenced only by the \( n-1 \)'th observation, i.e.

\[
p(x_n \mid x_1, \ldots, x_{n-1}) = p(x_n \mid x_{n-1})
\]

The chain of observations is a **1st-order Markov chain**, and the probability of a sequence of \( N \) observations is

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

What if the $n$'th observation in a chain of observations is influenced by a corresponding hidden variable?

If the hidden variables are discrete and form a Markov chain, then it is a hidden Markov model (HMM)

If the hidden variables are discrete and form a Markov chain, then it is a hidden Markov model (HMM)
Hidden Markov Models

What if the $n$'th observation in a chain of observations is influenced by a corresponding hidden variable?

If the hidden variables are discrete and form a Markov chain, then it is a hidden Markov model (HMM)
Hidden Markov Models

What if the \( n \)'th observation in a chain of observations is influenced by a corresponding hidden variable?

The joint distribution

\[
p(x_1, \ldots, x_N, z_1, \ldots, z_N) = p(z_1) \left[ \prod_{n=2}^{N} p(z_n | z_{n-1}) \right] \prod_{n=1}^{N} p(x_n | z_n)
\]

If the hidden variables are discrete and form a Markov chain, then it is a hidden Markov model (HMM)
Hidden Markov Models

What if the \( n \)th observation in a chain of observations is influenced by a corresponding hidden variable?

Hidden Markov Models

Transition probabilities

\[
p(x_1, \ldots, x_N, z_1, \ldots, z_N) = p(z_1) \prod_{n=2}^{N} p(z_n | z_{n-1}) \prod_{n=1}^{N} p(x_n | z_n)
\]

Emission probabilities

If the hidden variables are discrete and form a Markov chain, then it is a hidden Markov model (HMM)
**Transition probabilities**

**Notation:** In Bishop, the hidden variables $z_n$ are positional vectors, e.g. if $z_n = (0,0,1)$ then the model in step $n$ is in state $k=3$

**Transition probabilities:** If the hidden variables are discrete with $K$ states, the conditional distribution $p(z_n | z_{n-1})$ is a $K \times K$ table $A$, and the marginal distribution $p(z_1)$ describing the initial state is a $K$ vector $\pi$

The probability of going from state $j$ to state $k$ is:

$$A_{jk} \equiv p(z_{nk} = 1 | z_{n-1}, j = 1)$$

$$\sum_k A_{jk} = 1$$

The probability of state $k$ being the initial state is:

$$\pi_k \equiv p(z_{1k} = 1)$$

$$\sum_k \pi_k = 1$$
Transition probabilities:

If the hidden variables are discrete with $K$ states, the conditional distribution $p(z_n | z_{n-1})$ is a $K \times K$ table $A$, and the marginal distribution $p(z_1)$ describing the initial state is a $K$ vector $\pi$ ...

The probability of going from state $j$ to state $k$ is:

$$A_{jk} \equiv p(z_{nk} = 1 | z_{n-1}, j = 1)$$

$$\sum_k A_{jk} = 1$$

The probability of state $k$ being the initial state is:

$$\pi_k \equiv p(z_{1k} = 1)$$

$$\sum_k \pi_k = 1$$
The transition probabilities:

\[
p(z_n | z_{n-1}, A) = \prod_{k=1}^{K} \prod_{j=1}^{K} A_{jk}^{z_{n-1,j} z_{n,k}}
\]

\[
p(z_1 | \pi) = \prod_{k=1}^{K} \pi_{1k}^{z_{1,k}}
\]

The probability of going from state \(j\) to state \(k\) is:

\[A_{jk} \equiv p(z_{nk} = 1 | z_{n-1,j} = 1)\]

\[
\sum_k A_{jk} = 1
\]

The probability of state \(k\) being the initial state is:

\[\pi_k \equiv p(z_{1k} = 1)\]

\[
\sum_k \pi_k = 1
\]
**Emission probabilities:** The conditional distributions of the observed variables $p(x_n \mid z_n)$ from a specific state.

If the observed values $x_n$ are discrete (e.g. $D$ symbols), the emission probabilities $\Phi$ is a $K \times D$ table of probabilities which for each of the $K$ states specifies the probability of emitting each observable ...

$$p(x_n \mid z_n, \phi) = \prod_{k=1}^{K} p(x_n \mid \phi_k)^{z_{nk}}$$
Emission probabilities: The conditional distributions of the observed variables $p(x_n \mid z_n)$ from a specific state $z_n$.

If the observed values $x_n$ are discrete (e.g., $D$ symbols), the emission probabilities $\phi$ is a $K \times D$ table of probabilities which for each of the $K$ states specifies the probability of emitting each observable.

$$p(x_n \mid z_n, \phi) = \prod_{k=1}^{K} p(x_n \mid \phi_k)^{z_{nk}}$$
HMM joint probability distribution

\[ p(X, Z | \Theta) = p(z_1 | \pi) \prod_{n=2}^{N} p(z_n | z_{n-1}, A) \prod_{n=1}^{N} p(x_n | z_n, \phi) \]

Observables: \( X = \{x_1, \ldots, x_N\} \)
Latent states: \( Z = \{z_1, \ldots, z_N\} \)
Model parameters: \( \Theta = \{\pi, A, \phi\} \)

If \( A \) and \( \phi \) are the same for all \( n \) then the HMM is *homogeneous*.
Example – 2-state HMM

Observable: \{A, C, G, T\}, States: \{0,1\}

\[
\begin{array}{cccc}
A & \pi & \phi \\
0.95 & 0.05 & 1.00 & 0.00 \\
0.10 & 0.90 & & \\
\end{array}
\]
Example – 7-state HMM

Observable: \{A, C, G, T\}, States: \{0, 1, 2, 3, 4, 5, 6\}
A HMM generates a sequence of observables by moving from latent state to latent state according to the transition probabilities and emitting an observable (from a discrete set of observables, i.e. a finite alphabet) from each latent state visited according to the emission probabilities of the state ...

Model $M$:

A run follows a sequence of states:

And emits a sequence of symbols:
Computing $P(X,Z)$

$$p(X, Z|\Theta) = p(z_1|\pi) \left[ \prod_{n=2}^{N} p(z_n|z_{n-1}, A) \right] \prod_{n=1}^{N} p(x_n|z_n, \phi)$$

```python
def joint_prob(x, z):
    ""
    Returns the joint probability of x and z
    ""
    p = init_prob[z[0]] * emit_prob[z[0]][x[0]]
    for i in range(1, len(x)):
        p = p * trans_prob[z[i-1]][z[i]] * emit_prob[z[i]][x[i]]
    return p
```
Computing \( P(X,Z) \)

\[
p(X, Z|\Theta) = p(z_1|\pi) \prod_{n=2}^{N} p(z_n|z_{n-1}, A) \prod_{n=1}^{N} p(x_n|z_n, \phi)
\]

```python
def joint_prob(x, z):
    """Returns the joint probability of x and z"""
    p = init_prob[z[0]] * emit_prob[z[0]][x[0]]
    for i in range(1, len(x)):
        p = p * trans_prob[z[i-1]][z[i]] * emit_prob[z[i]][x[i]]
    return p
```

```
$ python hmm_jointprob.py hmm-7-state.txt test_seq100.txt
> seq100
p(x,z) = 1.8619524290102162e-65

$ python hmm_jointprob.py hmm-7-state.txt test_seq200.txt
> seq200
p(x,z) = 1.6175774997005771e-122

$ python hmm_jointprob.py hmm-7-state.txt test_seq300.txt
> seq300
p(x,z) = 3.0675430597843052e-183

$ python hmm_jointprob.py hmm-7-state.txt test_seq400.txt
> seq400
p(x,z) = 4.860704144302979e-247

$ python hmm_jointprob.py hmm-7-state.txt test_seq500.txt
> seq500
p(x,z) = 5.258724342206735e-306

$ python hmm_jointprob.py hmm-7-state.txt test_seq600.txt
> seq600
p(x,z) = 0.0
```
Computing $P(X,Z)$

$$p(X, Z | \Theta) = p(z_1 | \pi) \prod_{n=2}^{N} p(z_n | z_{n-1}, A) \prod_{n=1}^{N} p(x_n | z_n, \phi)$$

```python
$ python hmm_jointprob.py hmm-7-state.txt test_seq100.txt
> seq100
p(x,z) =  1.8619524290102162e-65

$ python hmm_jointprob.py hmm-7-state.txt test_seq200.txt
> seq200
p(x,z) =  1.6175774997005771e-122

$ python hmm_jointprob.py hmm-7-state.txt test_seq300.txt
> seq300
p(x,z) =  3.0675430597843052e-183

$ python hmm_jointprob.py hmm-7-state.txt test_seq400.txt
> seq400
p(x,z) =  4.860704144302979e-247

$ python hmm_jointprob.py hmm-7-state.txt test_seq500.txt
> seq500
p(x,z) =  5.258724342206735e-306

$ python hmm_jointprob.py hmm-7-state.txt test_seq600.txt
> seq600
p(x,z) =  0.0
```

Should be >0 by construction of $X$ and $Z$
Representing numbers

A floating point number $n$ is represented as $n = f \times 2^e$ cf. the IEEE-754 standard which specify the range of $f$ and $e$

See e.g. Appendix B in Tanenbaum's Structured Computer Organization for further details.
The problem – Too small numbers

For the simple HMM, the joint-probability $p(X, Z)$ is

$$p(X, Z) = 1 \cdot \prod_{n=2}^{N} 1 \cdot \prod_{n=1}^{N} \frac{1}{2} = \left(\frac{1}{2}\right)^n = 2^{-n}$$

If $n > 467$ then $2^{-n}$ is smaller than $10^{-324}$, i.e. cannot be represented
The problem – Too small numbers

For the simple HMM, the joint-probability \( p(X,Z) \) is

\[
p(X,Z) = 1 \cdot \prod_{n=2}^{N} 1 \cdot \prod_{n=1}^{N} \frac{1}{2} = \left(\frac{1}{2}\right)^n = 2^{-n}
\]

If \( n > 467 \) then \( 2^{-n} \) is smaller than \( 10^{-324} \), i.e. cannot be represented

No problem representing

\[
\log p(X,Z) = -n
\]

as the decimal range is approx \(-10^{308}\) to \(10^{308}\)
Solution: Compute $\log P(X, Z)$

$$p(X, Z|\Theta) = p(z_1|\pi) \left[ \prod_{n=2}^{N} p(z_n|z_{n-1}, A) \right] \prod_{n=1}^{N} p(x_n|z_n, \phi)$$

Use $\log (XY) = \log X + \log Y$, and define $\log 0$ to be $-\infty$.

$$\log p(X, Z|\Theta) = \log p(z_1|\pi) + \sum_{n=2}^{N} \log p(z_n|z_{n-1}, A) + \sum_{n=1}^{N} \log p(x_n|z_n, \phi)$$
Solution: Compute log $P(X,Z)$

$$
\log p(X, Z|\Theta) = \log p(z_1|\pi) + \sum_{n=2}^{N} \log p(z_n|z_{n-1}, A) + \sum_{n=1}^{N} \log p(x_n|z_n, \phi)
$$

```python
def log_joint_prob(self, x, z):
    
    Returns the log transformed joint probability of x and z

    logp = log(init_prob[z[0]]) + log(emit_prob[z[0]][x[0]])
    for i in range(1, len(x)):
        logp = logp + log(trans_prob[z[i-1]][z[i]]) + log(emit_prob[z[i]][x[i]])
    return logp
```
Solution: Compute \( \log P(X,Z) \)

\[
\log p(X, Z|\Theta) = \log p(z_1|\pi) + \sum_{n=2}^{N} \log p(z_n|z_{n-1}, A) + \sum_{n=1}^{N} \log p(x_n|z_n, \phi)
\]

```python
def log_joint_prob(self, x, z):
    """
    Returns the log transformed joint probability of x and z
    """
    logp = log(init_prob[z[0]]) + log(emit_prob[z[0]][x[0]])
    for i in range(1, len(x)):
        logp = logp + log(trans_prob[z[i-1]][z[i]]) + log(emit_prob[z[i]][x[i]])
    return logp
```

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq100.txt
> seq100
log p(x,z) = -149.04640541441395

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq200.txt
> seq200
log p(x,z) = -280.43445168576596

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq300.txt
> seq300
log p(x,z) = -420.25219508298494

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq400.txt
> seq400
log p(x,z) = -567.1573346564519

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq500.txt
> seq500
log p(x,z) = -702.9311499793356

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq600.txt
> seq600
log p(x,z) = -842.0056730984585
Using HMMs

- Determine the likelihood of a sequence of observations.
- Find a plausible underlying explanation (or decoding) of a sequence of observations.
Using HMMs

- Determine the likelihood of a sequence of observations.

- Find a plausible underlying explanation (or decoding) of a sequence of observations.

\[
p(X|\Theta) = \sum_Z p(X, Z|\Theta)
\]
Using HMMs

- Determine the likelihood of a sequence of observations.
- Find a plausible underlying explanation (or decoding) of a sequence of observations.

\[ p(X | \Theta) = \sum_Z p(X, Z | \Theta) \]

The sum has \( K^N \) terms, but it turns out that it can be computed in \( O(K^2N) \) time, but first we will consider decoding.
Decoding using HMMs

Given a HMM $\Theta$ and a sequence of observations $X = x_1, \ldots, x_N$, find a plausible explanation, i.e. a sequence $Z^* = z^*_1, \ldots, z^*_N$ of values of the hidden variable.
Decoding using HMMs

Given a HMM $\Theta$ and a sequence of observations $X = x_1, \ldots, x_N$, find a plausible explanation, i.e. a sequence $Z^* = z_1^*, \ldots, z_N^*$ of values of the hidden variable.

Viterbi decoding

$Z^*$ is the overall most likely explanation of $X$:

$$Z^* = \arg \max_Z p(X, Z | \Theta)$$
Decoding using HMMs

Given a HMM $\Theta$ and a sequence of observations $X = x_1, \ldots, x_N$, find a plausible explanation, i.e. a sequence $Z^* = z^*_1, \ldots, z^*_N$ of values of the hidden variable.

**Viterbi decoding**

$Z^*$ is the overall most likely explanation of $X$:

$$Z^* = \arg \max_Z p(X, Z|\Theta)$$

**Posterior decoding**

$z^*_n$ is the most likely state to be in the $n$'th step:

$$z^*_n = \arg \max_{z_n} p(z_n|x_1, \ldots, x_N)$$
Summary

- Terminology of hidden Markov models (HMMs)
- **Viterbi- and Posterior decoding** for finding a plausible underlying explanation (sequence of hidden states) of a sequence of observation
- **Next**: Algorithms for computing the Viterbi and Posterior decodings efficiently