Haladó Gépi Tanulás 6

Kornai András

HGT 2022/10/13

Óraterv

- Csoportbeszámolók
- Markov és Rejtett Markov (HMM) modellek folyt.
- Gaussian Mixtures (GMM)

How do HMMs work?

To define an HMM we need

- A set of model states $S = \{s_1, s_2, ..., s_n\}$
- Initial probabilities $p_1, p_2, ..., p_n$
- Transition probabilities $t_{i,j}$ $(1 \leq i, j \leq n)$
- Output probabilities $P(f \mid s_i)$

A single run

Begin in state s_i according to the p_i, and emit some signal f in accordance with $P(f \mid s_i)$. Move to some state s_j with probability $t_{i,j}$, continue

The probability of a particular sequence

$$P(f_1 f_2 \ldots f_k) =$$

$$\sum_{\pi \in S^k} p_{\pi(1)} \prod_{j<k} t_{\pi(j),\pi(j+1)} \prod_{j \leq k} P(f_j \mid \pi(j))$$

Brute force computation would require $n^k 2k$ multiplications, because of the Markov assumption this can be reduced to $n^2 2k$

Using the Markov assumption

The trick: instead of a single register accumulating the probabilities through various paths, maintain a separate register for each state.

Initialization: $R(i, 0) = p_i \ (1 \leq i \leq n)$

Iteration: $R(i, k+1) = \sum_{j=1}^{n} R(j, k) t_{j,i} P(f_{k+1} \mid s_i)$
(n multiplication for each of the n registers)

Termination: $P(f_1 f_2 ... f_K) = \sum_{j=1}^{n} R(j, K)$

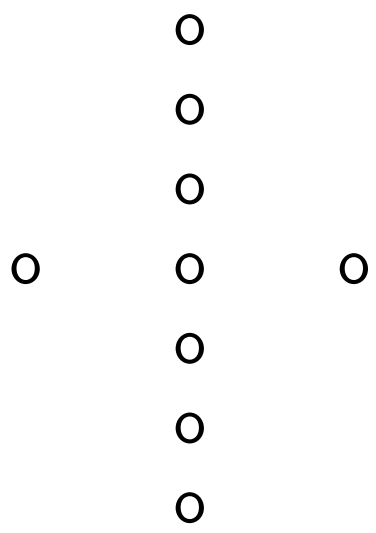
The Viterbi algorithm

Goal: find the best $\pi \in S^k$ given some $f_1 f_2 ... f_k$
Brute force: maintain the score along each of the n^k paths with some heuristic for pruning when this number gets too large.
Using the Markov assumption: for each state maintain only the best path that ends there, because the best one longer path will be an extension of one of these.

o o o o o o o
o o o o o o o
o o o o o o o
o o o o o o o
o o o o o o o

Isolated Character Recognition with HMMs

Build a model for each character, and find the best path through the resulting complex model.



(Same method works for small-vocabulary word-recognition)

Networks of networks are networks

1. word models built from character models
2. n-gram models built on character models
3. syntax models built from word n-grams

Increasing the Markovian parameter

The trick: instead of chains with memory going back to 2,3,... steps build 1-chains made out of compound states (pairs, triples,...)

A tale of two coins

- *Fake:* produces Head with $p = .9$, Tail with $p = .1$
- *Real:* produces Head with $p = .5$, Tail with $p = .5$
- *Coin swap is hard:* magician will use the same coin as in the preceding trial with $p = .99$