Universität Potsdam
Institut für Informatik
Lehrstuhl Maschinelles Lernen

Hidden-Markov-Modelle

Tobias Scheffer
Peter Haider
Paul Prasse
Uwe Dick

Hidden-Markov-Modelle: Wozu?

- Spracherkennung:
  - Akustisches Modell.
- Geschriebene Sprache:
  - Part-of-Speech-Tagging,
  - Informationsextraktion.
- Biologie:
  - Finden von Genen in der DNA.

Markov-Prozesse

- $X_1, ..., X_n$: Zufallsvariablen.
- Allgemein gilt: $P(X_1,..., X_n) = P(X_1)\prod_{i=2}^{n} P(X_i \mid X_{i-1},..., X_1)$
- Zufallsvariablen bilden eine Markovkette, gdw:
  $P(X_1,..., X_n) = P(X_1)\prod_{i=2}^{n} P(X_i \mid X_{i-1})$
- Jede Variable X_i nur von Vorgänger $X_{i\text{-}1}$ abhängig.

- Markov-Modell:
  Probabilistischer endlicher Automat, Folge der Zustände ist Markov-Kette.

- (Andrei Markov, 1856-1922)

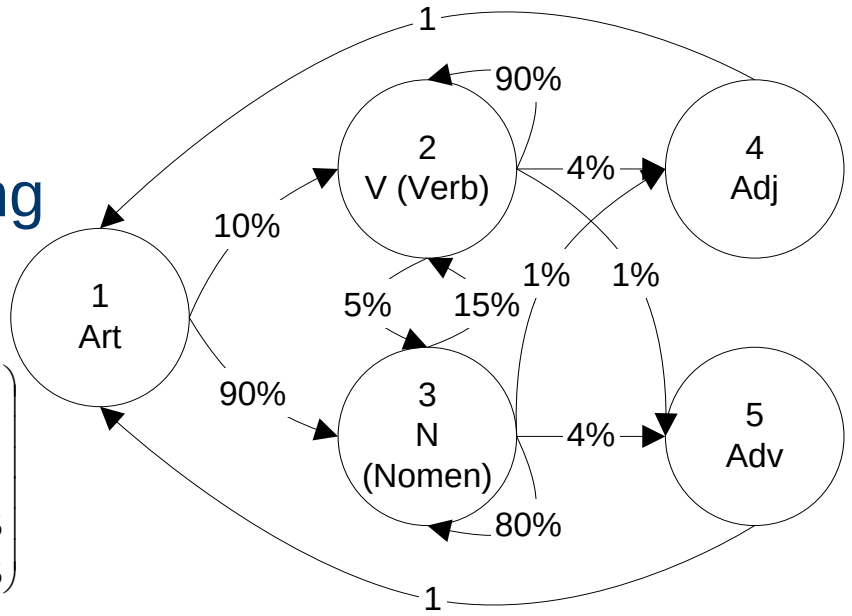
Markov-Modell

- Zustände 1,..., N (Folge der Zustände ist Markov-Kette),
- Transitionswahrscheinlichkeiten a_{ij}; $\sum_{j=1}^{N} a_{ij} = 1$
- Startwahrscheinlichkeiten π_i.

- Zustand zur Zeit t: q_t.

- Z.B. Wortartenerkennung („Part-of-Speech Tagging“)

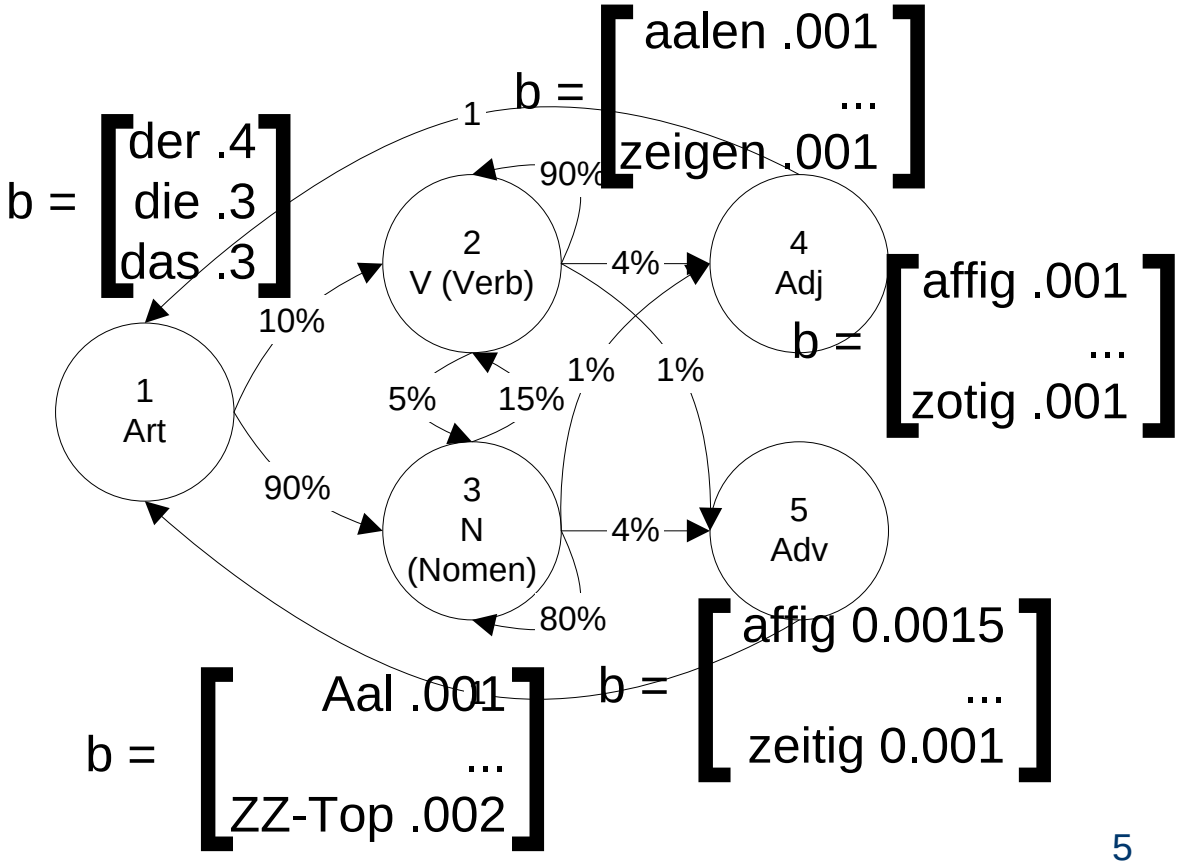
$$\pi = \begin{pmatrix} 0.4 \\ 0.2 \\ 0.3 \\ 0.05 \\ 0.05 \end{pmatrix}$$



- P(Artikel, Nomen, Nomen, Verb)?

Hidden-Markov-Modell

- Folge der Zustände ist nicht sichtbar.
- Statt dessen: Zustände emittieren Beobachtungen O_t (mit Wahrscheinlichkeit $b_i(O_t)$).



Hidden-Markov-Modell, Definitionen

- Zustände $\Omega = \{1, \ldots, N\}$ q_t: Zustand zur Zeit t.
- Übergangswahrscheinlichkeiten $A = \{a_{ij}\}$
- Startwahrscheinlichkeiten $\pi = \{\pi_i = P(q_1 = i)\}$
- Beobachtungswahrscheinlichkeiten
  $B = \{b_i(O_t) = P(O_t \mid q_t = i)\}$

- HMM definiert durch Parameter $\lambda = (A, B, \pi)$

Markov-Annahmen

- Markov-Annahme für Zustandsfolgen:

$$P(q_t \mid q_{t-1},\ldots,q_1) = P(q_t \mid q_{t-1})$$

- Markov-Annahme für Beobachtungen:

$$P(O_t \mid q_t, q_{t-1},\ldots,q_1) = P(O_t \mid q_t)$$

Drei Basisprobleme

- Problem 1: Likelihood einer Beobachtungsfolge:
  - „Wie gut passt ein Modell zu einer Beobachtungsfolge?“
  - Berechne $P(O_1,...,O_T \mid \lambda)$
- Problem 2: Optimale Zustandskette finden:
  - „Welche Zustandskette hat die Beobachtung am wahrscheinlichsten erzeugt?“
  - Berechne $\arg\max_{q_1,...,q_T} P(q_1,...,q_T \mid O_1,...,O_T,\lambda)$
- Problem 3: Lernproblem
  - „Gegeben viele Beobachtungsfolgen, finde die Parameter des HMMs!“
  - Berechne $\arg\max_{\lambda} P(\{(O_1,...,O_T),...\} \mid \lambda)$

„Wie gut passt ein Modell zur Beobachtungsfolge?“

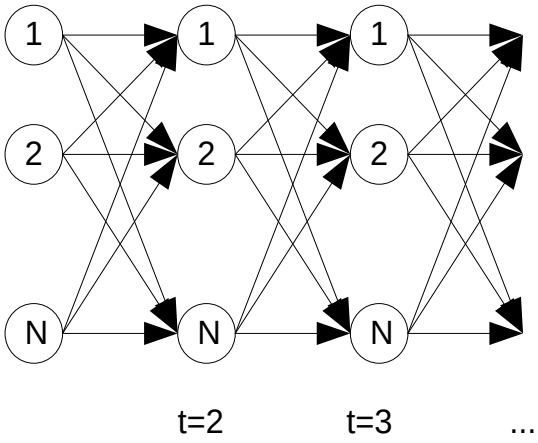
- $$P(O_1,\ldots,O_T \mid \lambda) = \sum_{alle(q_1,\ldots,q_T)} P(O_1,\ldots,O_T \mid q_1,\ldots,q_T,\lambda) P(q_1,\ldots,q_T \mid \lambda)$$

  $$= \sum_{alle(q_1,\ldots,q_T)} b_{q_1}(O_1)..b_{q_T}(O_T)\pi_{q_1} a_{q_1 q_2}\ldots a_{q_{T-1} q_T}$$

- # Summanden = N^T.
- Auswertung exponentiell in der Länge der Eingabe.
- Bei Auswertung werden dieselben Wahrscheinlichkeiten wiederholt berechnet
- Gesucht: polynomieller Algorithmus.
- Dynamische Programmierung: Zwischenergebnisse speichern.

Trellis

- Trellis: Array über Zustände x Zeit.
- Hilft der Vorstellung, muss man aber nicht implementieren.



- Rekursive Hilfsvariablen
  - $\alpha_t(i) = P(O_1,...,O_t, q_t = i \mid \lambda)$
  - $\beta_t(i) = P(O_{t+1},...,O_T \mid q_t = i, \lambda)$
  - $\gamma_t(i) = P(q_t = i \mid O_1,...,O_T, \lambda)$

„Forward“

- Wahrscheinlichkeit einer initialen Beobachtungsfolge und eines Zustands:

$$\alpha_t(i) = P(O_1,...,O_t, q_t = i \mid \lambda)$$

- Theorem:

$$\alpha_1(i) = \pi_i b_i(O_1)$$

$$\alpha_{t+1}(j) = \left(\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right) b_j(O_{t+1})$$

- Nach Theorem kann α durch dynamische Programmierung berechnet werden:
  - Initialisiere $\alpha_1(i)$.
  - Für t von 2 bis T: berechne $\alpha_t(i)$ unter Verwendung der schon bestimmten $\alpha_{t-1}(i)$.

„Forward“: Beweis

- Induktionsverankerung:

$$\alpha_1(i) = P(O_1, q_1 = i \mid \lambda)$$
$$= P(q_1 = i \mid \lambda) P(O_1 \mid q_1 = i, \lambda) = \pi_i b_i(O_1)$$

- Induktionsschritt t $\rightarrow$ t+1

$$\alpha_{t+1}(j) = P(O_1, \ldots, O_{t+1}, q_{t+1} = j \mid \lambda) = \sum_{i=1}^{N} P(O_1, \ldots, O_{t+1}, q_t = i, q_{t+1} = j \mid \lambda)$$

$$= \sum_{i=1}^{N} P(O_1, \ldots, O_t, q_t = i \mid \lambda) P(q_{t+1} = j \mid q_t = i, O_1, \ldots, O_t, \lambda)$$

$$P(O_{t+1} \mid q_{t+1} = j, q_t = i, O_1, \ldots, O_t, \lambda)$$

$$= \sum_{i=1}^{N} P(O_1, \ldots, O_t, q_t = i \mid \lambda) P(q_{t+1} = j \mid q_t = i, \lambda) P(O_{t+1} \mid q_{t+1} = j, \lambda)$$

$$= \left(\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right) b_j(O_{t+1})$$

„Forward“: Termination

- $P(O_1,...,O_T \mid \lambda) = \sum_i \alpha_T(i)$
- Beweis:

$$P(O_1,...,O_T \mid \lambda) = \sum_i P(O_1,...,O_T, q_T = i \mid \lambda) = \sum_i \alpha_T(i)$$

Problem 1 gelöst

- Problem 1 ist gelöst, nämlich das Lösen von

$$P(O_1,...,O_T \mid \lambda)$$

kann nun effizient durchgeführt werden.

„Welches Modell passt am besten?“

- Bsp: Worterkennung. Ein HMM für jedes Wort, das erkannt werden soll.
- Gegeben: Sprachsignal (Beobachtungssequenz), gesucht: Welches der Wörter wurde gesagt?
- $$\arg\max_k P(\lambda_k \mid O_1,...,O_T)$$
  $$= \arg\max P(O_1,...,O_T \mid \lambda_k) P(\lambda_k)$$
- Likelihood durch Forward-Algorithmus, A-Priori-Wahrscheinlichkeit durch Abzählen der Worthäufigkeit in der Trainingsmenge.

Problem 2: Was ist die optimale Zustandskette?

- Beispiele:
  - Part-of-Speech-Tagging, ein Zustand pro Part-of-Speech,
  - Gensequenzanalyse, Zustände entsprechen Tags, mit denen das Genom annotiert werden soll.
- Möglichkeit 1: Welcher einzelne Zustand zur Zeit t passt am besten zur Beobachtungsfolge?
  - $\arg\max_i \gamma_t(i) = \arg\max_i P(q_t = i \mid O_1,...,O_T, \lambda)$
  - Bestimmung durch Forward-Backward-Algorithmus

„Backward“

- $\beta_t(i) = P(O_{t+1},...,O_T \mid q_t = i, \lambda)$
- Theorem:

$$\beta_T(i) = 1$$

$$\beta_t(i) = \left(\sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \right)$$

- Nach dem Theorem kann β durch dynamische Programmierung bestimmt werden:
  - Initialisiere $\beta_T(i)=1$.
  - Für t von T-1 bis 1: bestimme $\beta_t(i)$ unter Verwendung der $\beta_{t+1}(j)$.

„Backward“: Beweis

- Induktionsschritt t+1 $\rightarrow$ t

$$\beta_t(i) = P(O_{t+1},\ldots,O_T \mid q_t = i,\lambda)$$

$$= \sum_j P(O_{t+1},\ldots,O_T, q_{t+1} = j \mid q_t = i,\lambda)$$

$$= \sum_j P(O_{t+1},\ldots,O_T \mid q_{t+1} = j, q_t = i,\lambda) P(q_{t+1} = j \mid q_t = i,\lambda)$$

$$= \sum_j P(O_{t+2},\ldots,O_T \mid q_{t+1} = j, q_t = i,\lambda) P(q_{t+1} = j \mid q_t = i,\lambda)$$

$$P(O_{t+1} \mid q_{t+1} = j, q_t = i,\lambda)$$

$$= \sum_j P(O_{t+2},\ldots,O_T \mid q_{t+1} = j,\lambda) P(q_{t+1} = j \mid q_t = i,\lambda) P(O_{t+1} \mid q_{t+1} = j,\lambda)$$

$$= \sum_j \beta_{t+1}(j) a_{ij} b_j(O_{t+1})$$

„Forward Backward“: Wahrscheinlichkeit eines Zustandes

- $\gamma_t(i) = P(q_t = i \mid O_1, \ldots, O_T, \lambda)$
- P(Zustand i zur Zeit t | Beobachtungssequenz)
- $$\gamma_t(i) = P(q_t = i \mid O_1, \ldots, O_T, \lambda)$$
$$= \frac{P(q_t = i, O_1, \ldots, O_T \mid \lambda)}{P(O_1, \ldots, O_T \mid \lambda)}$$
$$= \frac{P(q_t = i, O_1, \ldots, O_t, O_{t+1}, \ldots, O_T \mid \lambda)}{P(O_1, \ldots, O_T \mid \lambda)}$$
$$= \frac{P(q_t = i, O_1, \ldots, O_t \mid \lambda) P(O_{t+1}, \ldots, O_T \mid q_t = i, \lambda)}{P(O_1, \ldots, O_T \mid \lambda)}$$
$$= \frac{\alpha_t(i)\beta_t(i)}{P(O_1, \ldots, O_T \mid \lambda)}$$

Forward-Backward-Algorithmus

- (Forward)
- Initialisiere $\alpha_1(i) = \pi_i b_i(O_1)$ (alle Zustände i)
- Für t von 1 bis T-1
  - Berechne (für alle j) $\alpha_{t+1}(j) = \left(\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right) b_j(O_{t+1})$
- Berechne $P(O_1,...,O_T \mid \lambda) = \sum_i \alpha_T(i)$
- (Backward)
- Initialisiere $\beta_T(i) = 1$
- Für t von T-1 bis 1
  - Berechne (für alle i) $\beta_i(t) = \left(\sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \right)$
  - Berechne (für alle i) $\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O_1,...,O_T \mid \lambda)}$

Forward-Backward-Algorithmus

- Läuft mit quadratischem Aufwand: $O(TN^2)$
- Berechnet $\gamma_t(i) = P(q_t = i \mid O_1,...,O_T,\lambda)$
  und $P(O_1,...,O_T \mid \lambda)$

Problem 2: Was ist die optimale Zustandskette?

- Möglichkeit 1: Welcher einzelne Zustand zur Zeit t passt am besten zur Beobachtungsfolge?
  - $\arg\max_i \gamma_t(i) = \arg\max_i P(q_t = i \mid O_1,...,O_T,\lambda)$
  - Bestimmung durch Forward-Backward-Algorithmus
- Möglichkeit 2: Welche komplette Zustandsfolge passt am besten zur Beobachtungsfolge?
  - $\arg\max_{(q_1,...,q_T)} P(q_1,...,q_T \mid O_1,...,O_T,\lambda)$
  - Bestimmen mit Viterbi-Algorithmus

Viterbi-Algorithmus, Theorem

- $\delta_t(i) = \max_{q_1,\ldots,q_{t-1}} P(q_1,\ldots,q_{t-1},q_t = i,O_1,\ldots,O_t \mid \lambda)$
- Theorem:

  $\delta_1(i) = \pi_i b_i(O_1)$

  $\delta_{t+1}(j) = \left(\max_i \delta_t(i) a_{ij}\right) b_j(O_{t+1})$
- Beweis:

$$\delta_{t+1}(j) = \max_{q_1,\ldots,q_t} P(q_1,\ldots,q_t,q_{t+1} = j,O_1,\ldots,O_{t+1} \mid \lambda)$$
$$= \max_{q_1,\ldots,q_t} P(q_1,\ldots,q_t,O_1,\ldots,O_t \mid \lambda) P(q_{t+1} = j \mid q_t,\ldots) P(O_{t+1} \mid q_{t+1} = j,\ldots)$$
$$= \left(\max_i \left(\max_{q_1,\ldots,q_{t-1}} P(q_1,\ldots,q_t = i,O_1,\ldots,O_t \mid \lambda)\right) a_{ij}\right) b_j(O_{t+1})$$
$$= \left(\max_i \delta_t(i) a_{ij}\right) b_j(O_{t+1})$$

- Vorgängerzustand auf wahrscheinlichstem Pfad: $\psi_t(j)$

Viterbi-Algorithmus

- Initialisierung: $\delta_1(i) = \pi_i b_i(O_1)$
- Initialisierung: $\psi_1(i) = 0$
- Für t von 1 bis T-1 und j von 1 bis N:
  - $\delta_{t+1}(j) = \left(\max_i \delta_t(i) a_{ij}\right) b_j(O_{t+1})$
  - $\psi_{t+1}(j) = \left(\arg\max_i \delta_t(i) a_{ij}\right)$
- Termination $q_T^* = \arg\max_i \delta_T(i)$
- Für t von T-1 bis 1
  - $q_t^* = \psi_{t+1}(q_{t+1}^*)$

- Ausgabe der Zustandsfolge $q_1^*, \ldots, q_T^*$

Problem 3: Lernproblem

- Gegeben: Sammlung von Beobachtungsfolgen.
- Gesucht HMM-Parameter λ.
- Sichtbare Zustände
  - Z.B. Part-of-Speech-Tagging: Jede Beobachtung ist mit dem zugehörigen Zustand markiert.
  - Schätzen der Parameter durch Zählen der Starthäufigkeiten, Transitionen, Beobachtungen.
- Unsichtbare Zustände
  - Z.B. Worterkennung: Nur Sprachsignal gegeben, Zustandsfolgen sind unbekannt.
  - Lernen der Parameter durch Baum-Welch-Algorithmus.

Sichtbare Zustände

- Trainingsmenge

$$S = \left\langle (O^{(1)}, Q^{(1)}), ..., (O^{(m)}, Q^{(m)}) \right\rangle$$

$$O^{(k)} = O_1^{(k)}, ..., O_{T_k}^{(k)}$$

$$Q^{(k)} = q_1^{(k)}, ..., q_{T_k}^{(k)}$$

- Schätze π_i = (#Beispielsequenzen k mit $q_1^{(k)} = i$)/m
- Schätze a_{ij} = (#Stellen mit $q_t^{(k)} = i, q_{t+1}^{(k)} = j$) / (#Stellen mit $q_t^{(k)} = i$)
- Schätze $b_i(O)$ = (#Stellen mit $q_t^{(k)} = i, O_t^{(k)} = O$) / (#Stellen mit $q_1^{(k)} = i$)

Unsichtbare Zustände

- Trainingsmenge

$$S = \left\langle O^{(1)}, \ldots, O^{(m)} \right\rangle$$

$$O^{(k)} = O_1^{(k)}, \ldots, O_{T_k}^{(k)}$$

- Zustände unbekannt
- Forward-Backward kann Zustandswahrscheinlichkeiten berechnen, braucht dafür aber Modell,
- Können Modell schätzen (letzte Folie), brauchen dafür aber Zustandswahrscheinlichkeiten.

Baum-Welch-Algorithmus

- Wenn die Zustände der Beobachtungen bekannt wären, könnte man die Parameter durch Abzählen der Häufigkeiten in Trainingsmenge schätzen.
- Instanz des EM-Algorithmus.
- Beginne mit zufälligen Parametern und iteriere zwei Schritte bis zur Konvergenz
  - Berechne die Zustände durch Forward-Backward-Algorithmus auf Grundlage des aktuellen Modells
  - Schätze die Parameter des Modells auf Grundlage berechneter Zustände.

Baum-Welch-Algorithmus

- Hilfsvariable: $\xi_t(i,j) = P(q_t = i, q_{t+1} = j \mid O_1,...,O_T,\lambda)$
- Berechnung:

$$P(q_t = i, q_{t+1} = j \mid O_1,...,O_T,\lambda)$$

$$= \frac{P(q_t = i, q_{t+1} = j, O_1,...,O_T \mid \lambda)}{P(O_1,...,O_T \mid \lambda)}$$

$$= \frac{1}{P(O_1,...,O_T \mid \lambda)} P(O_1,...,O_t, q_t = i \mid \lambda) P(q_{t+1} = j \mid q_t = i, O_1,...,O_T,\lambda)$$

$$\times P(O_{t+1} \mid q_{t+1} = j, q_t = i, O_1,...,O_t,\lambda) P(O_{t+2},...,O_T \mid q_{t+1} = j, q_t = i, O_1,...,O_{t+1},\lambda)$$

$$= \frac{1}{P(O_1,...,O_T \mid \lambda)} P(O_1,...,O_t, q_t = i \mid \lambda) P(q_{t+1} = j \mid q_t = i,\lambda)$$

$$\times P(O_{t+1} \mid q_{t+1} = j,\lambda) P(O_{t+2},...,O_T \mid q_{t+1} = j,\lambda)$$

$$= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O_1,...,O_T \mid \lambda)}$$

Baum-Welch-Algorithmus

- Trainingsmenge $S = \langle O^{(1)}, \ldots, O^{(m)} \rangle; O^{(k)} = O_1^{(k)}, \ldots, O_{T_k}^{(k)}$
- Zustände unbekannt

1. Initialisiere λ zufällig.
2. Wiederhole bis Konvergenz: Für alle k von 1 bis m
   - Berechne die α, β, γ durch Forward-Backward
   - Für i und j von 1 bis N, berechne $\xi_t(i,j)$
   - Schätze $\pi_i^{(k)} = \gamma_1(i)$
   - Schätze $a_{ij}^{(k)} = \sum_t \xi_t(i,j) \Big/ \sum_t \gamma_t(i)$
   - Schätze $b_i^{(k)}(O) = \sum_{t:O_t=O} \gamma_t(i) \Big/ \sum_t \gamma_t(i)$
3. Mittle Schätzer für λ über m Beispiele und Wiederhole ab Schritt 2.

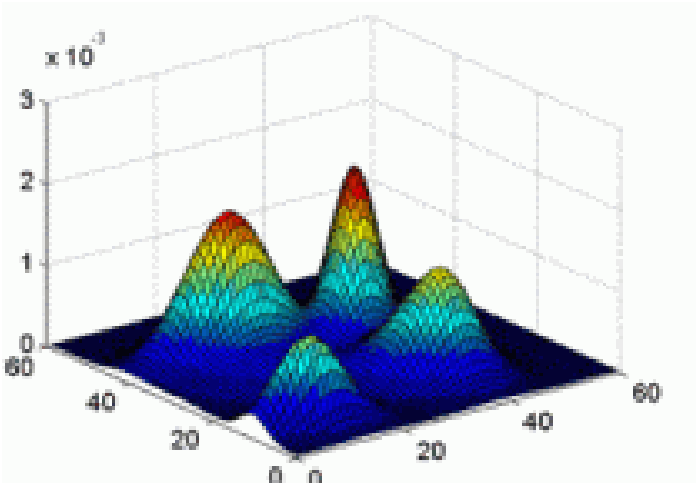
Problem 3 ist gelöst

- Problem 3 wird vom Baum-Welch Algorithmus gelöst.

Kontinuierliche HMM

- Bisher: endlich viele Beobachtungen O,$b_i(O)$ diskrete Verteilung
- Sprachverarbeitung: Beobachtung kontinuierliches Signal, kontinuierliche Dichtefunktion $b_i(X)$.
- $b_i(X)$ Überlagerung von M multivariaten Gaußverteilungen.

$$b_i(x) = \sum_{k=1}^{M} c_{ik} N[\mu_{ik}, \Sigma_{ik}](x) = \sum_{k=1}^{M} c_{ik} b_{ik}(x)$$



Kontinuierliche HMM

- Bisher: $b_i^{(k)}(O) = \sum_{t:O_t=O} \gamma_t(i) \Big/ \sum_t \gamma_t(i)$
- Stattdessen: Maximum-Likelihood-Schätzer der kontinuierlichen Verteilung.
- Likelihood: $b_i(x) = \sum_{k=1}^{M} c_{ik} N[\mu_{ik}, \Sigma_{ik}](x)$
- Problem: sowohl die Zugehörigkeit c_{ik} der Beobachtungen als auch die M Verteilungen N[...] sind unbekannt.
- Schätzen einer multivariaten Gaußschen Mischverteilung wird durch EM-Algorithmus gelöst.
- In jeder Baum-Welch-Iteration und für jeden der N Zustände wird der EM-Algorithmus zur Schätzung der $b_i(X)$ gestartet.

Schätzen der Gaußschen Mischverteilung

- Hier: Vereinfachte Darstellung. Nur univariate Mischverteilung.
- Vereinfachte Darstellung: x_1 bis x_m seien die Beobachtungen, die im aktuellen Zustand i erzeugt wurden (tatsächlich wird jede Beobachtung O_t mit $\gamma_t(i)$ im Zustand i erzeugt).

$$b_{ik}(x) = P(x \mid Modell_k) = P(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i; \qquad \sigma = \sqrt{\frac{\sum_{i=1}^{m}(x_i - \mu)^2}{m(m-1)}}$$

ML-Hypothese für Normalverteilung

- ML-Hypothese für Normalverteilung
- $\mu_{ML} = \arg\min_{\mu} \sum_{i=1}^{m} (x_i - \mu)^2$

$$= \frac{1}{m} \sum_{i=1}^{m} x_i$$

$$\sigma_{ML} = \sqrt{\frac{\sum_{i=1}^{m} (x_i - \mu)^2}{m(m-1)}}$$

$$h_{ML} = \arg\max_h P(D \mid h)$$

$$= \arg\max_h \prod_{i=1}^{m} p(x_i \mid h)$$

$$= \arg\max_{(\mu,\sigma)} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x_j - \mu}{\sigma}\right)^2}$$

$$= \arg\max_{(\mu,\sigma)} \ln \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x_i - \mu}{\sigma}\right)^2}$$

$$= \arg\max_{(\mu,\sigma)} \sum_{i=1}^{m} \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2}\left(\frac{x_i - \mu}{\sigma}\right)$$

ML-Hypothese für mehrere normalverteilte Cluster

- Z_J: Clusterzuordnung: $z_{IJ} = 1$, wenn x_i aus Cluster j stammt.

$$\mu_j = \frac{\sum_{i=1}^{m} z_{ij} x_i}{\sum_{i=1}^{m} z_{ij}}$$

- Wenn x_i mit einer Wahrscheinlichkeit von $E(z_{IJ})$ aus Cluster j stammt:

$$\mu_j = \frac{\sum_{i=1}^{m} E[z_{ij}] x_i}{\sum_{i=1}^{m} E[z_{ij}]}$$

Probabilistische Clusterzuordnung

- Wahrscheinlichkeit dafür, dass x_i aus Cluster j stammt:
-
$$E[z_{ij}] = P(Modell_j \mid x_i)$$
$$= \frac{P(x_i \mid Modell_j) P(Modell_j)}{\sum_k P(x_i \mid Modell_k) P(Modell_k)}$$
- Bei gleich wahrscheinlichen Clustern:

$$E[z_{ij}] = P(Modell_j \mid x_i)$$
$$= \frac{P(x_i \mid Modell_j)}{\sum_k P(x_i \mid Modell_k)}$$

EM-Algorithmus (intuitiv)

- Beginne mit zufälligen Werten für alle
- Wiederhole
  - E-Schritt: Für alle Datenpunkte i:

$$E[z_{ij}] = \frac{P(x_i \mid \mu_j, \sigma_j) P(j)}{\sum_k P(x_i \mid \mu_k, \sigma_k) P(k)}$$

  - M-Schritt: Schätze alle μ_i, σ_i:

$$\mu_j = \frac{\sum_{i=1}^{m} E[z_{ij}] x_i}{\sum_{i=1}^{m} E[z_{ij}]}; \sigma_j = \frac{\sqrt{\sum_{i=1}^{m} E[z_{ij}](x_i - \mu)^2}}{\sum_{i=1}^{m} E[z_{ij}]}$$

Baum-Welch: Kommentare

- Bei der Initialisierung darf keine Wahrscheinlichkeit auf 0 gesetzt werden (lokales Optimum).
- HMM-Größe (z.B.)
  - „Tetrahydrocannabinol“: 24 Zustände
  - „a“: 3 Zustände.
  - 3 bis ca. 64 Gaußkurven in jedem Zustand.
- Training für akustische Modellierung erfordert Sprachdaten, die auf Wortebene „getaggt“ sind.

Skalierung

- Forward-Backward und Viterbi multiplizieren viele Wahrscheinlichkeiten auf, numerisch kommt dabei schnell 0 heraus.
- Mit Log-Wahrscheinlichkeiten arbeiten, statt mit Wahrscheinlichkeiten.
- Forward-Backward- und Viterbi lassen sich entsprechend umformulieren.
- (Aus Multiplikationen werden Additionen)

Hidden-Markov-Modelle: Erweiterungen

- Hierarchische HMM: Knoten können Unterknoten haben, für Part-of-Speech-Tagging interessant (z.B. kommt ein Verb in einer Verbphrase vor).
- Conditional Random Fields und Hidden-Markov-Support-Vektor-Maschinen:
  - Diskriminative (statt generativer) Sequenzmodelle.
  - Modellieren P(q | O) statt P(q, O).
- Mehrdimensionale HMM: Markov Random Fields (Bildverarbeitung).
- Probabilistische, kontextfreie Grammatiken: CFG statt endlichen Automaten.

Diskriminative Modelle

- Optimierungskriterium beim Lernen von HMMs:
  - Bei ML-Schätzer: $P((O^{(1)},Q^{(1)}),...,(O^{(m)},Q^{(m)})\,|\,\lambda)$
  - Bei MAP-Schätzer: $P(\lambda\,|\,(O^{(1)},Q^{(1)}),...,(O^{(m)},Q^{(m)}))$
    $= P((O^{(1)},Q^{(1)}),...,(O^{(m)},Q^{(m)})\,|\,\lambda)\,p(\lambda)$
- Bei vielen Anwendungen eigentlich gewünscht: korrekte Zustände gegeben Beobachtungen.
  - $P(Q^{(1)},...,Q^{(m)}\,|\,O^{(1)},...,O^{(m)},\lambda)$
  - Lässt sich mit diskriminativen Modellen optimieren.

Diskriminative Modelle: HM-SVM

- „Hidden-Markov-Support-Vector-Machine“
- Vorhersage:
  - $q_1^* ... q_T^* = \arg\max_{Q_1 ... Q_T} f(Q_1 ... Q_T, O_1, ..., O_t)$

    $= \arg\max_{Q_1 ... Q_T} \mathbf{w}^T \Phi(Q_1 ... Q_T, O_1, ..., O_t)$

- Φ wird so konstruiert, dass es dieselben Informationen enthält, die auch HMM verarbeitet:

$$\Phi(O,Q) = \begin{pmatrix} \#[q_t = 1][O_t = o_1] \\ ... \\ \#[q_t = 1][O_t = o_M] \\ ... \\ \#[q_t = N][O_t = o_1] \\ ... \\ \#[q_t = N][O_t = o_M] \\ \#[q_t = 1][q_{t+1} = 1] \\ ... \\ \#[q_t = N][q_{t+1} = N] \end{pmatrix}$$

Vorkommen von Beobachtung o_1 im Zustand 1

Vorkommen von Beobachtung o_M im Zustand *N*

Vorkommen von Übergängen von Zustand *N* nach Zustand *N*.

Diskriminative Modelle: HM-SVM

- Φ wird so konstruiert, dass es dieselben Informationen enthält, die auch HMM verarbeitet:

$$\Phi(O,Q) = \begin{pmatrix} \#[q_t = 1][O_t = o_1] \\ \dots \\ \#[q_t = 1][O_t = o_M] \\ \dots \\ \#[q_t = N][O_t = o_1] \\ \dots \\ \#[q_t = N][O_t = o_M] \\ \#[q_t = 1][q_{t+1} = 1] \\ \dots \\ \#[q_t = N][q_{t+1} = N] \end{pmatrix}$$

Vorkommen von Beobachtung o_1 im Zustand 1

Vorkommen von Beobachtung o_M im Zustand *N*

Vorkommen von Übergängen von Zustand *N* nach Zustand *N*.

- Jeder Eintrag des Parametervektors w entspricht einem einzelnen Parameter des HMM (Übergangs- bzw. Beobachtungswahrscheinlichkeit)

Diskriminative Modelle: HM-SVM

- Statt einstellen der Parameter durch Abzählen:
  - diskriminatives Optimierungskriterium
- Optimierungskriterium: richtige Zustandsfolge für jede Beobachtungsfolge mit maximalem Abstand (Margin):

$$\min_{w} ||w||^2 + \sum_{k} \left[\max\{0, 1 - w^T \Phi(O^{(k)}, Q^{(k)}) + \max_{\bar{Q} \neq Q^{(k)}} w^T \Phi(O^{(k)}, \bar{Q}) \} \right]$$

HMM vs. HM-SVM

- Entscheidungsfunktionen HMM vs HM-SVM:
  - $f(Q,O)$ vs. $\log P(Q,O)$
  - Gleiche Form der linearen Entscheidungsfunktion.
- Optimierungskriterien:
  - $P(\lambda \mid Q^{(1..m)}, O^{(1..m)}) = P(Q,^{(1..m)} O^{(1..m)} \mid \lambda) p(\lambda)$
  - Vs. max Margin $\min_w ||w||^2 + \sum_k \left[\max\{0, 1 - w^T \Phi(O^{(k)}, Q^{(k)}) + \max_{\bar{Q} \neq Q^{(k)}} w^T \Phi(O^{(k)}, \bar{Q})\} \right]$
  - = Generativ vs. diskriminativ.
  - Generativ = Einstellen der Parameter, so dass Wahrscheinlichkeit von Zuständen und Beobachtungen maximal
  - Diskriminativ = Einstellen der Parameter, so dass möglichst die richtigen Zustände vorhergesagt werden, gegeben die Beobachtungen

Diskriminative Modelle: Conditional Random Fields

- Modell von $P(Q|O)$ statt $P(Q,O)$
  - $P(Q|O) \propto \exp\left[\mathbf{w}^T \Phi(Q,O)\right]$
- Optimierungskriterium:
  - $\arg\max_{\mathbf{w}} P(Q^{(1)},...,Q^{(m)} | O^{(1)},...,O^{(m)}, \mathbf{w}) p(\mathbf{w})$

    $= \arg\max_{\mathbf{w}} \log P(Q^{(1)},...,Q^{(m)} | O^{(1)},...,O^{(m)}, \mathbf{w}) - \Omega(\mathbf{w})$

    $= \arg\max_{\mathbf{w}} \sum_i \mathbf{w}^T \Phi(Q^{(i)}, O^{(i)}) - \Omega(\mathbf{w})$
- HM-SVM vs. CRF:
  - HM-SVM: Maximiere Abstand zu falschen Zustandssequenzen
  - CRF: Entscheidungsfunktionswert für richtige Zustandssequenz möglichst hoch.

Fragen?