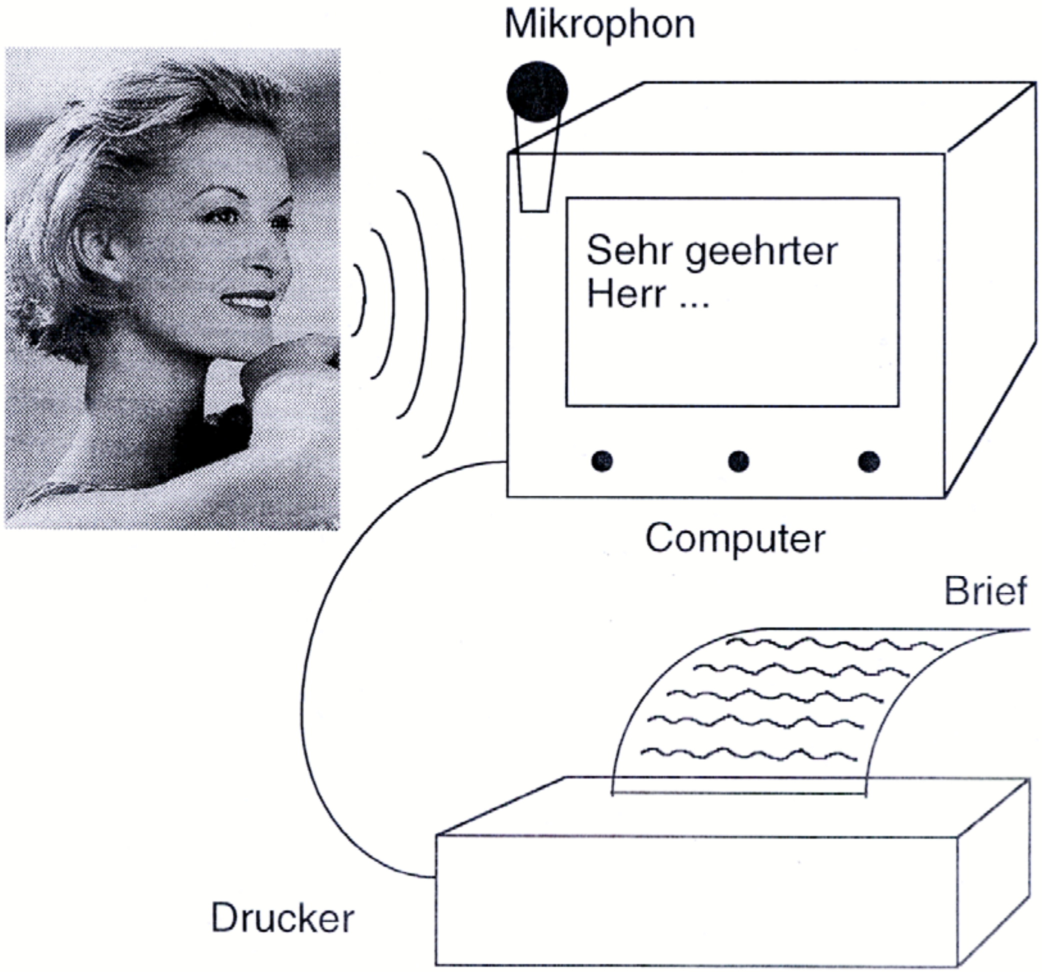
Kapitel 3: Spracherkennung

sprachgesteuerte Schreibmaschine

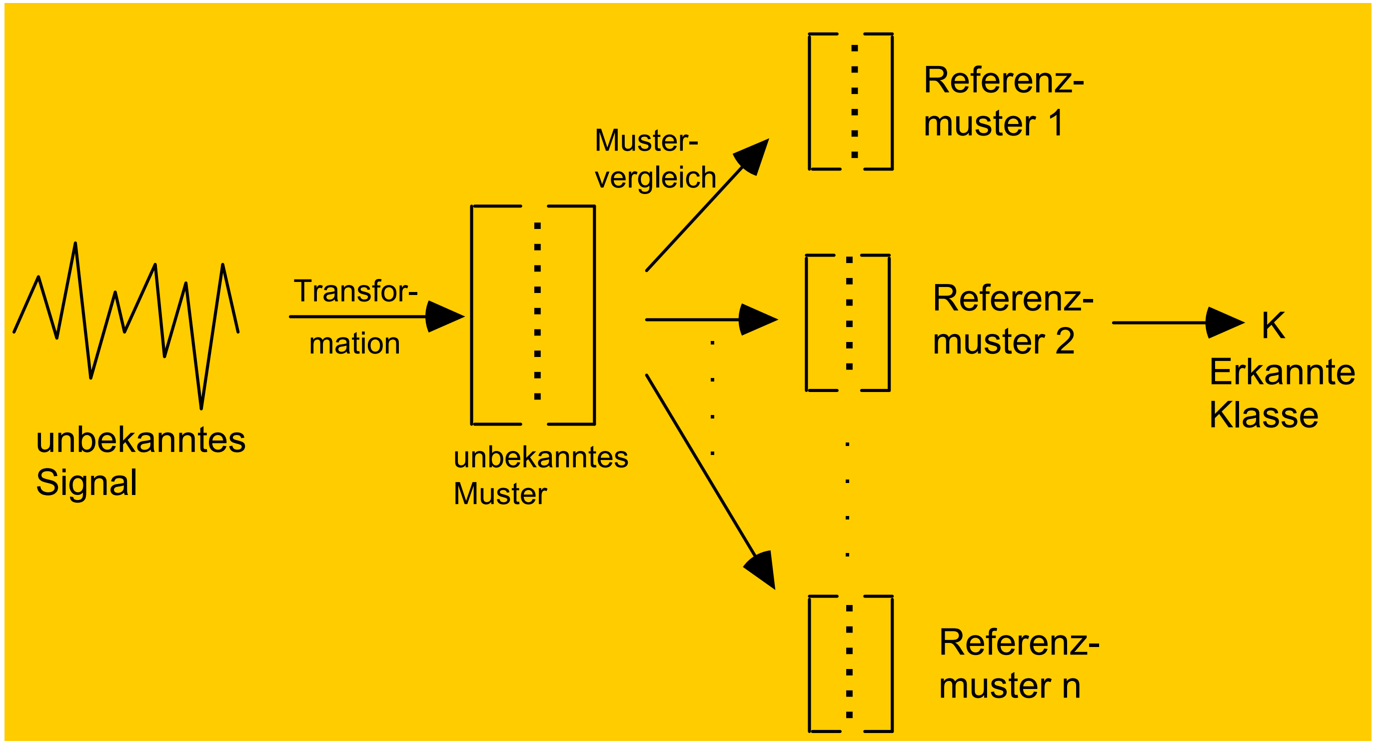


⇒ **AUFGABE:** **<u>AUTOMATISCHE SPRACHERKENNUNG MIT COMPUTERN</u>**

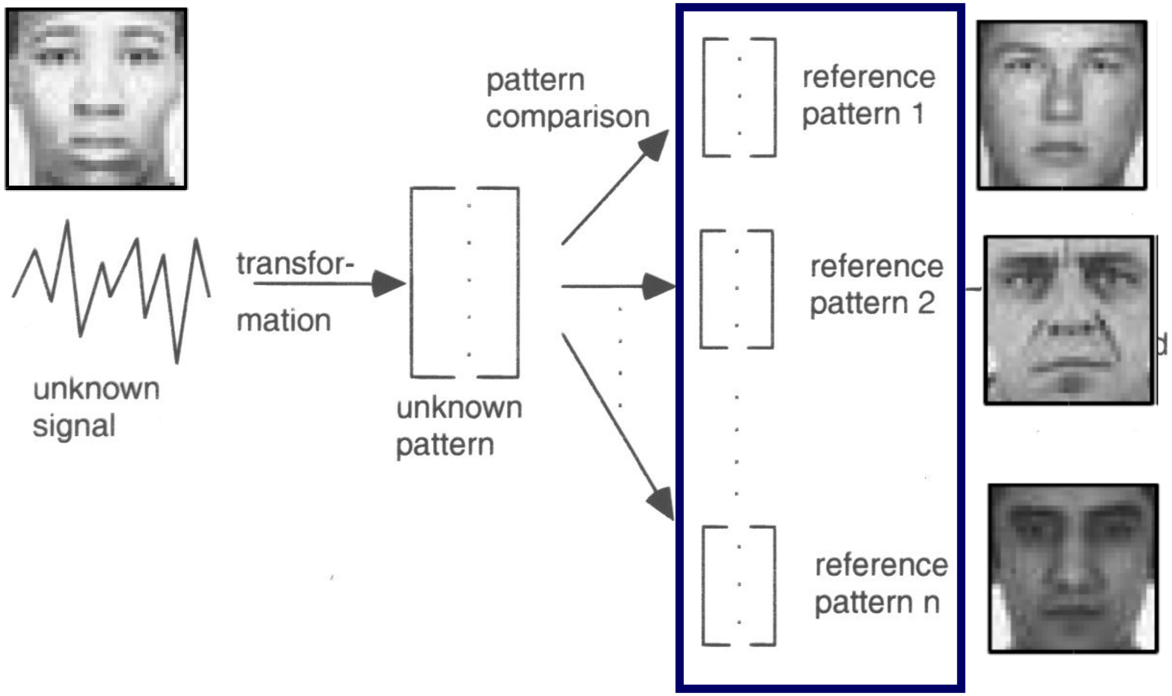
aktuelle Anwendungen der Spracherkennung

- **TELEKOMMUNIKATION**
  - => - AUSKUNFTSSYSTEME
  - - BUCHUNGSSYSTEME
  - => - AUTOMATISCHES CALL-ROUTING
  - => - TELEFONIER-HILFEN
  - - AUTOMATISCHE ÜBERSETZUNG
- **BÜROAUTOMATISIERUNG**
  - - DOKUMENTERSTELLUNG
  - => - BEFEHLSEINGABE FÜR PC
  - - CAD/CAE-SYSTEME
- **MEDIZIN**
  - => - RADIOLOGIE
  - - STEUERUNG VON MIKROSKOPEN
  - => - BEHINDERTENBEREICH
- **PRODUKTION**
  - => - QUALITÄTSKONTROLLE
  - => - LAGERSTEUERUNG
  - - WARENVERTEILUNG
  - - DATENERFASSUNG
  - - MASCHINENSTEUERUNG
  - - LEITWARTEN FÜR ANLAGEN
- **PRIVATER BEREICH**
  - - SCHNITTSTELLE FÜR MULTIMEDIA
  - - MULTIMODALE KOMMUNIKATION
  - => - UNTERHALTUNGSELEKTRONIK
  - => - KONSUMGÜTER, SPIELZEUGE
  - - HAUSHALT
  - - AUTO

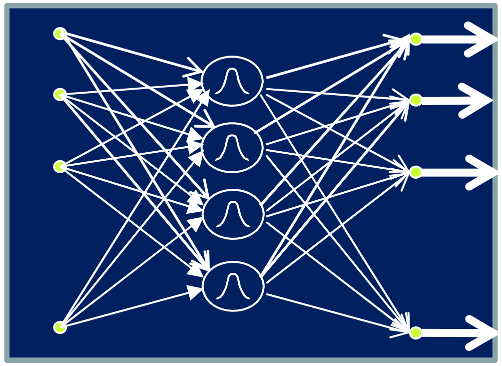
Generelle Vorgehensweise bei der Mustererkennung



Beispiel Gesichtserkennung

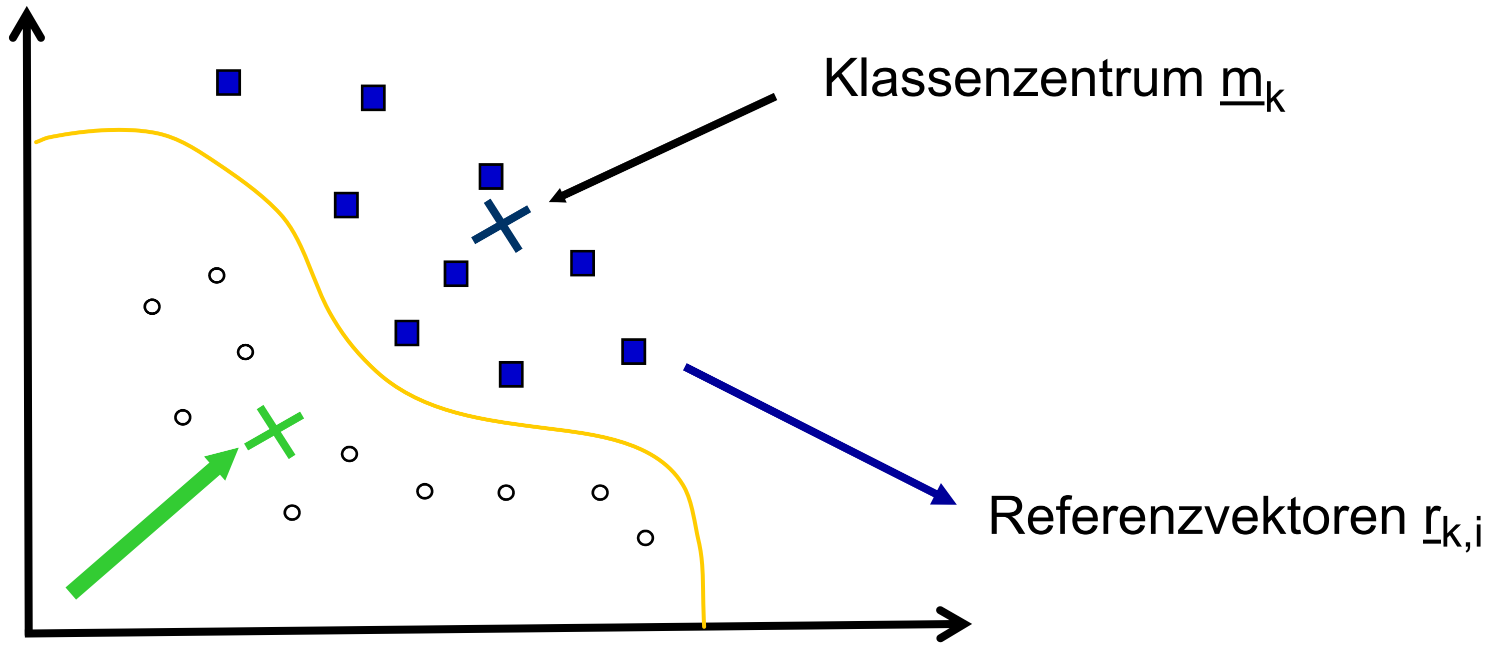


- ➔ Machine Learning
- ➔ Deep Learning
- ➔ Big Data



Klassifikator, z.B.: Neural Network, Support Vector Machine, Hidden Markov Model

Abstandsklassifizierung



Abstandsberechnung

Vektor $\underline{x}$: Unbekannter, zu klassifizierender Mustervektor
Vektoren $\underline{r}_{k,i}$: i-ter Referenzvektor für die k-te Klasse

Klassenzentrum:
$$\mathbf{m}_k = \frac{1}{M_k} \cdot \sum_{i=1}^{M_k} \mathbf{r}_{k,i}$$

Abstandsformel:

$$d_k(\mathbf{x}, \mathbf{m}_k) = (\mathbf{x} - \mathbf{m}_k)^{\top} \cdot \mathbf{W}_k \cdot (\mathbf{x} - \mathbf{m}_k)$$

Minimaler Abstand:

$$k_x = \underset{k}{\operatorname{argmin}}\, d_k(\mathbf{x}, \mathbf{m}_k)$$

Wahl der Gewichtungsmatrix

Einfachster Fall: Euklidischer Abstand

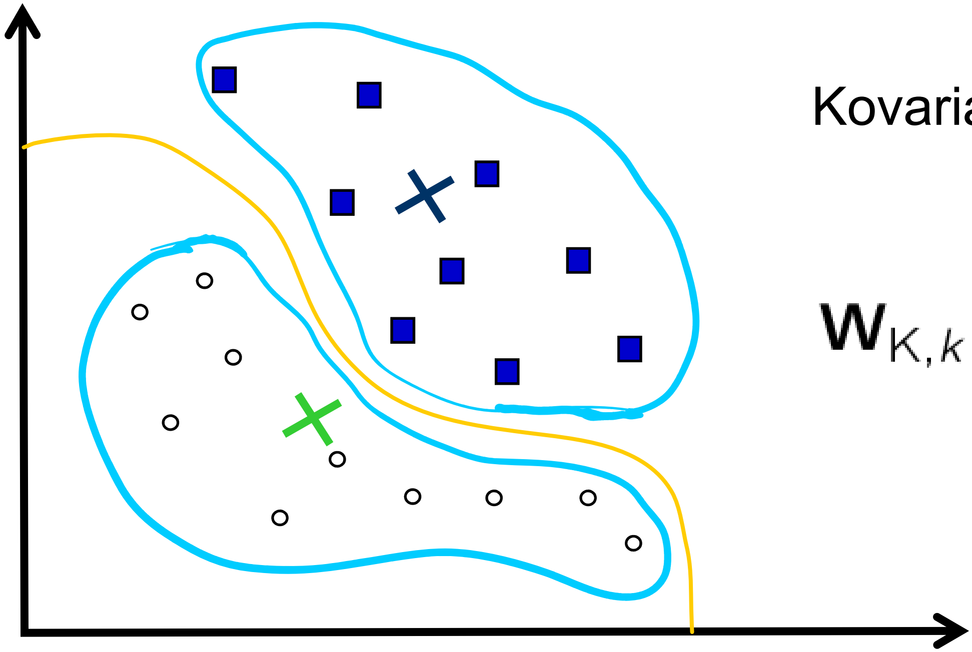
$$\mathbf{W}_k = \mathbf{W} = \mathbf{W}^{-1} = \mathbf{1}$$

Auch denkbar: Gewichtung mit Diagonalmatrix

$$W = \begin{pmatrix} w_{11} & 0 & 0 & 0 \\ 0 & w_{22} & 0 & 0 \\ 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & w_{NN} \end{pmatrix}$$

oder Gewichtung mit vollbesetzter Matrix …..

Mahalanobis-Abstand



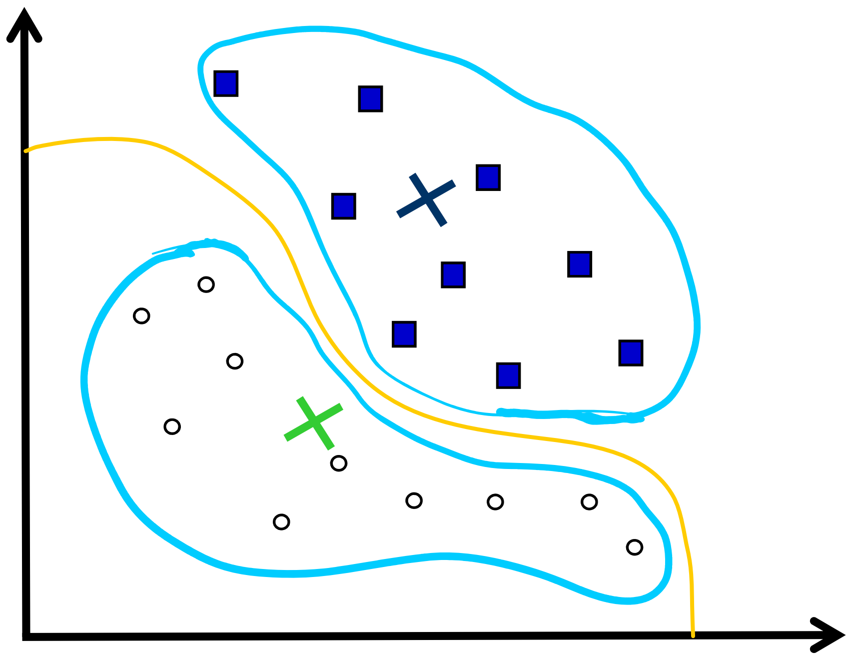
Kovarianzmatrix der k-ten Klasse:

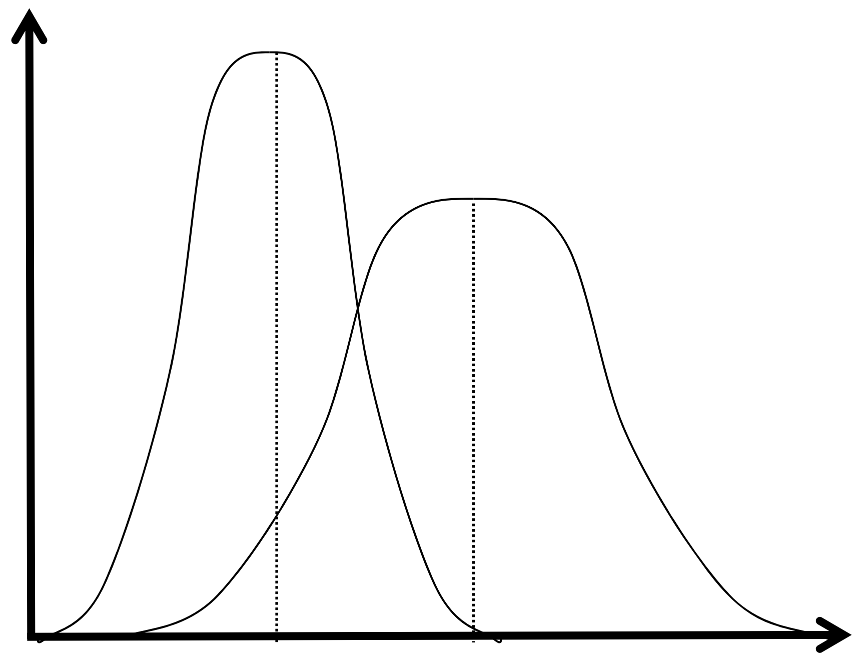
$$\mathbf{W}_{K,k} = \frac{1}{M_k} \sum_{i=1}^{M_k} \mathbf{r}_{k,i} \cdot \mathbf{r}_{k,i}^{\mathsf{T}} \quad - \mathbf{m}_k \cdot \mathbf{m}_k^{\mathsf{T}}$$

Automatische Berechnung der Gewichtungsmatrix:

$$d_k(\mathbf{x}, \mathbf{m}_k) = (\mathbf{x} - \mathbf{m}_k)^{\mathsf{T}} \cdot \mathbf{W}_{K,k}{}^{-1} \cdot (\mathbf{x} - \mathbf{m}_k)$$

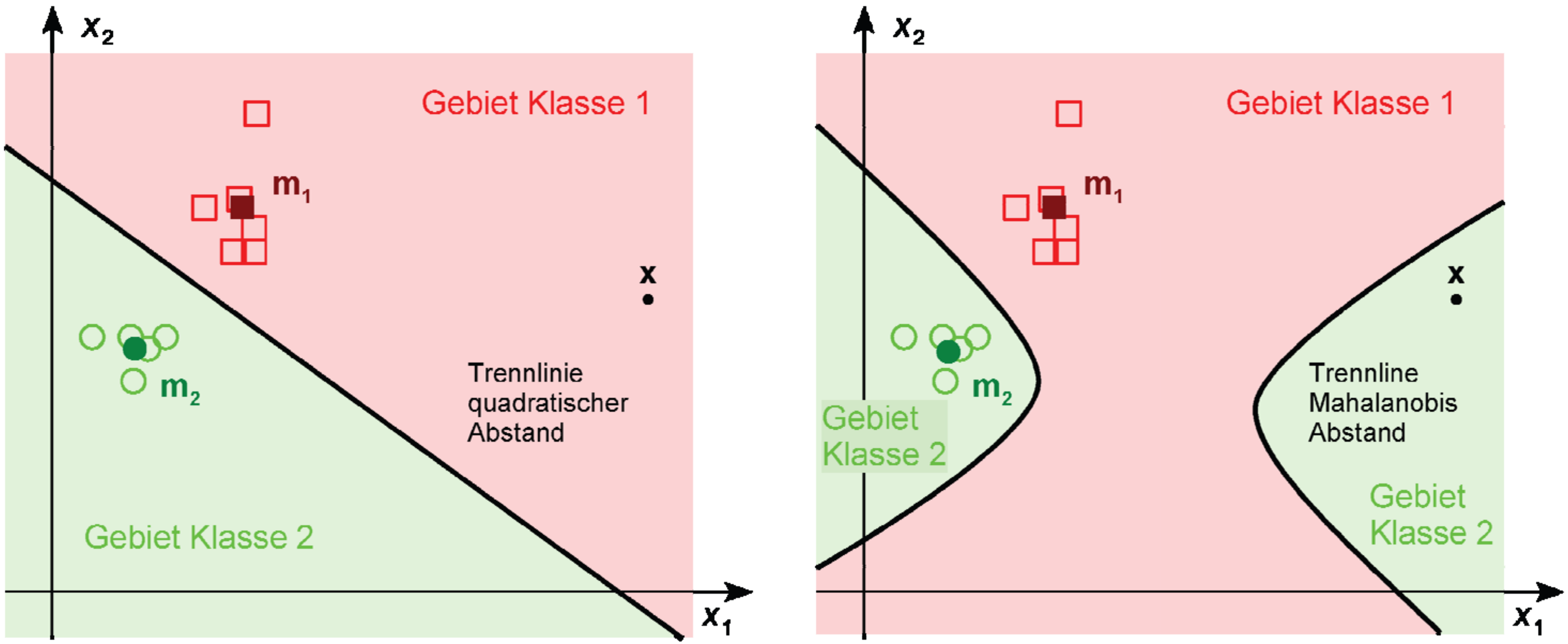
Probabilistische Interpretation



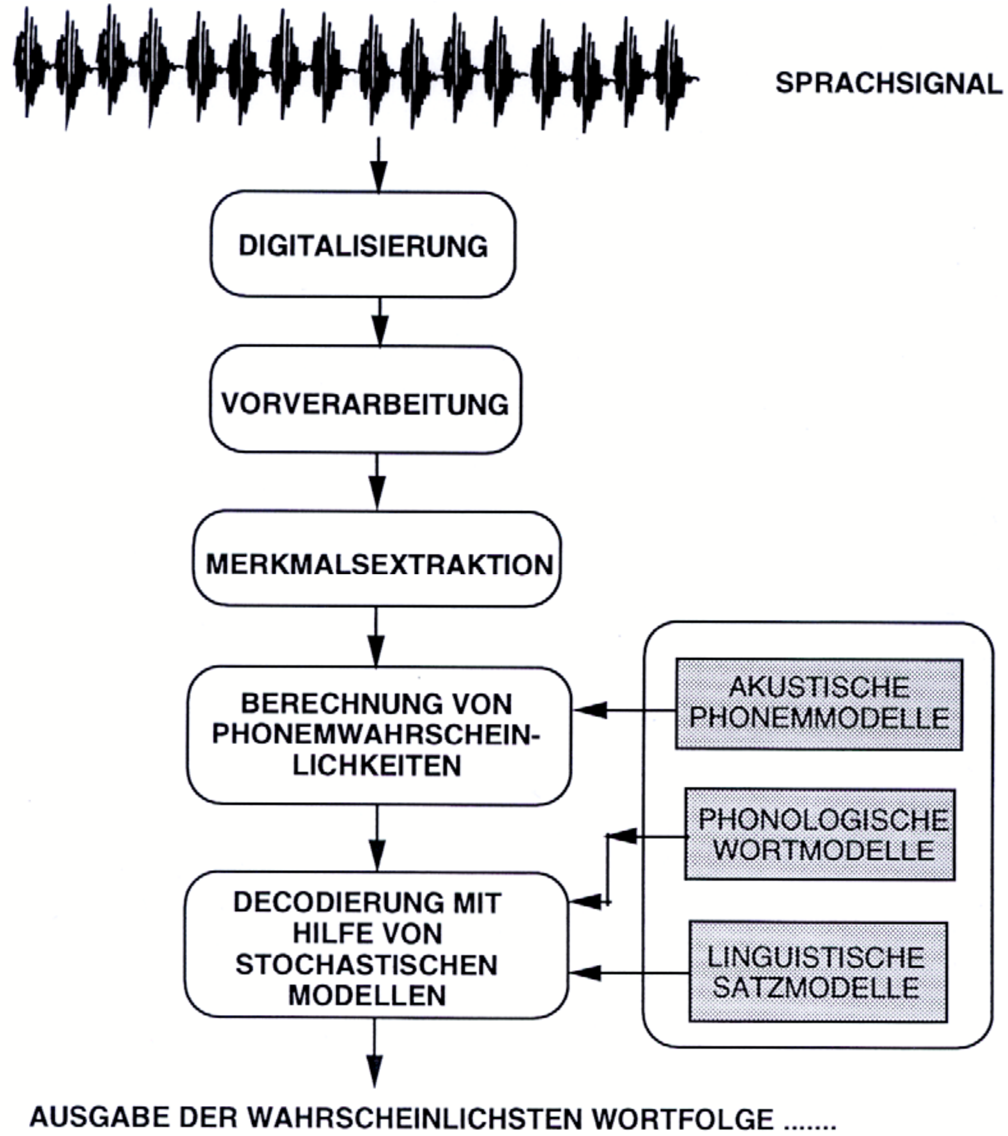


$$p(\underline{x}\,|\,K) = \frac{1}{\sqrt{(2\pi)^d \cdot |W_k|}} \cdot e^{-\frac{1}{2}(\underline{x}-\underline{m}_k)^T W_k^{-1} (\underline{x}-\underline{m}_k)}$$

Klassentrennungsfähigkeiten



Funktionsweise eines Spracherkennungssystems



Grundprinzip des Cepstrums

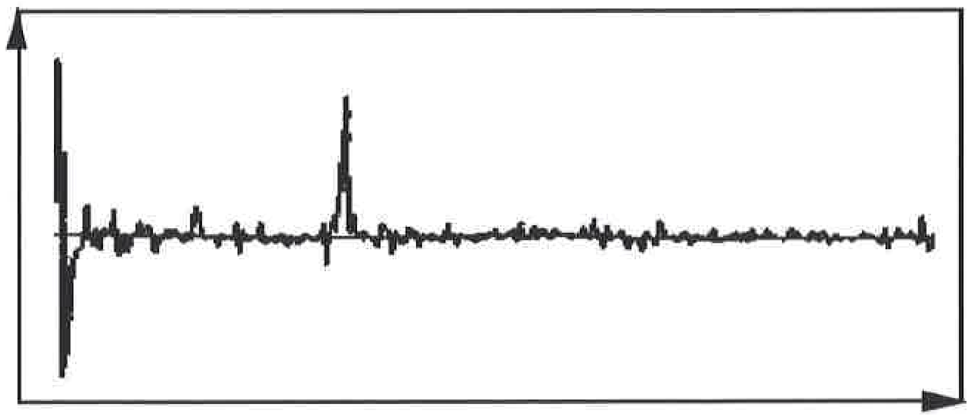
Sprachsignal — Spektrum — Cepstrum

$$\longrightarrow \boxed{\mathcal{F}} \longrightarrow \boxed{\log} \longrightarrow \boxed{\mathcal{F}^{-1}} \longrightarrow$$

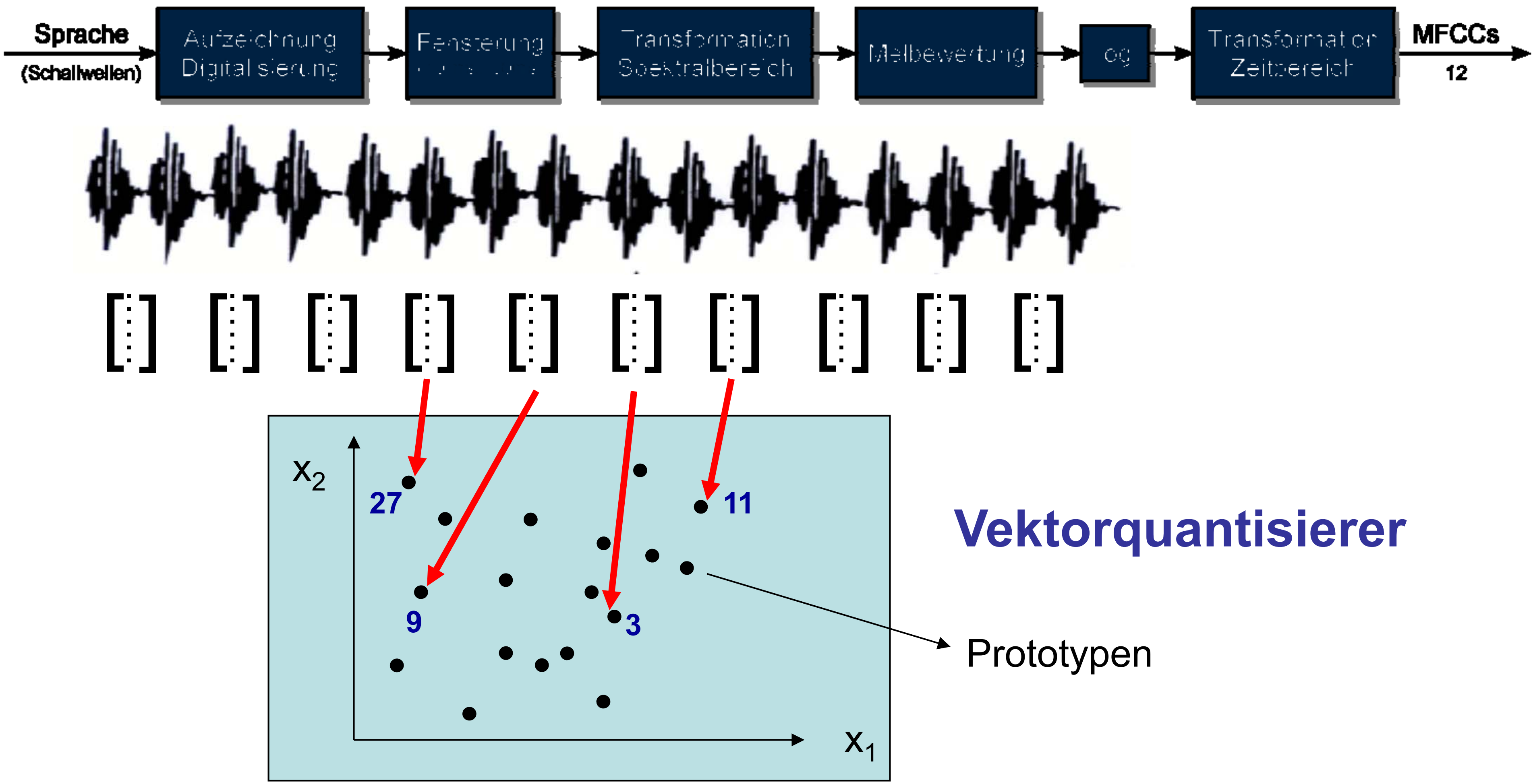
$$y(t) = h(t) * x(t) \qquad Y(\omega) = H(\omega) \cdot X(\omega) \qquad Y'(\omega) = \log H(\omega) + \log X(\omega) \qquad y'(t) = \mathcal{F}^{-1}\{\log H(\omega)\} + \mathcal{F}^{-1}\{\log X(\omega)\}$$

Praktische Berechnung:

- Selektion eines Zeitfensters für das betrachtete Sprachsignal
- Fourier-Transformation dieses Signals in den Frequenzbereich
- Bilden des Betrags des resultierenden (komplexen) Spektrums
- Logarithmierung des Amplitudenspektrums
- Rücktransformation mit inverser FT

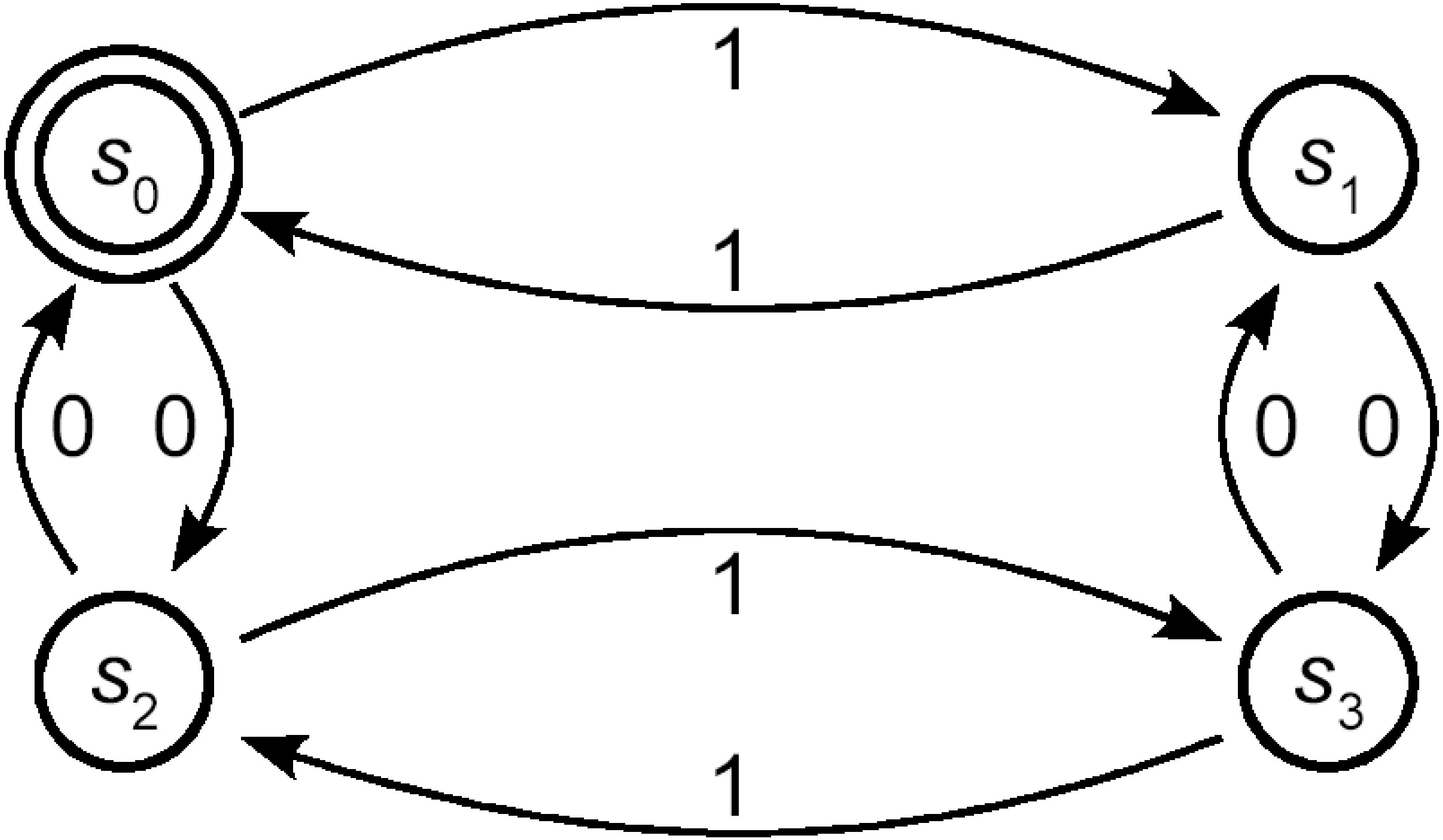


Merkmalsextraktion für die Spracherkennung



$\rightarrow$ **O = (. . . . . 27 9 3 11 . . . . .)**

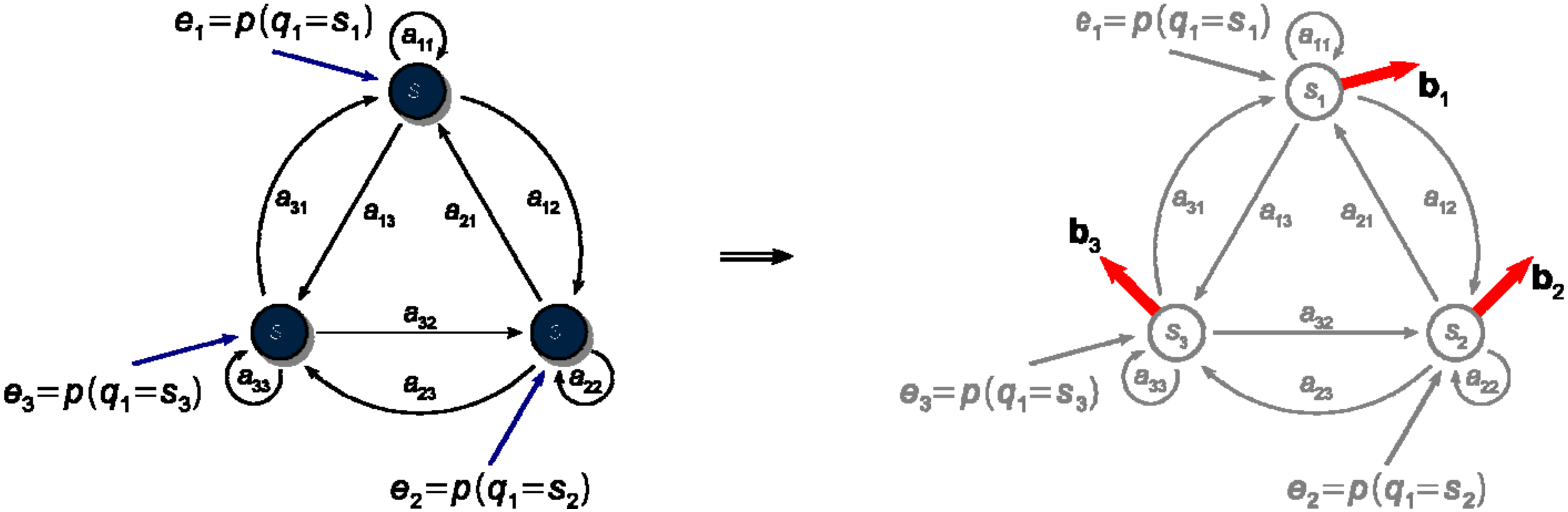
Zustandsautomat



Verarbeitung einer Symbolfolge

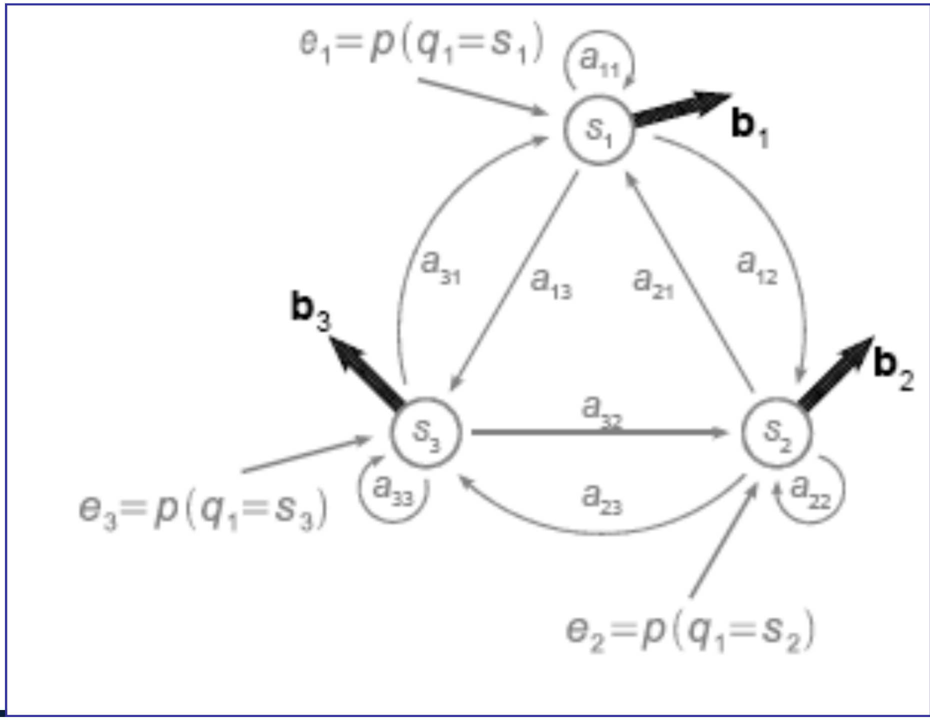
Symbolfolge		1		0		1		1		0	
Zustandsfolge	s_0		s_1		s_3		s_2		s_3		s_1

Markov-Modell und Hidden-Markov-Modell

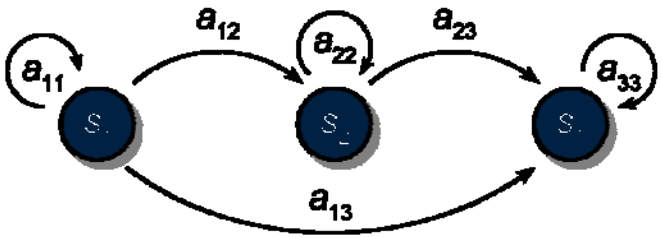


Ergodisches Hidden-Markov-Modell

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{pmatrix}, \text{ mit } (\forall i,j)\{a_{ij} > 0\}$$



Links-Rechts-Modell



$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix}$$

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ 0 & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{NN} \end{pmatrix}, \text{ also } a_{ij} = 0 \text{ für } j < i$$

Parameter von Hidden-Markov-Modellen

Matrix der Übergangs-
wahrscheinlichkeiten:

$$\mathbf{A} = p\{q_{t+1} = s_j | q_t = s_i\} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{pmatrix}$$

Vektor der Einsprungs-
wahrscheinlichkeiten:

$$\mathbf{e} = \vec{\pi} = \begin{pmatrix} p(q_1 = s_1) \\ p(q_1 = s_2) \\ \vdots \\ p(q_1 = s_N) \end{pmatrix}$$

beobachtete Symbolfolge:

$$\mathbf{o} = (o_1, o_2, \ldots, o_T)$$

Alphabet der möglichen
Ausgangssymbole:

$$\mathbf{v} = (v_1, v_2, \ldots, v_M)^{\mathsf{T}}$$

Beobachtungswahrscheinlichkeiten

Beobachtungswahrscheinlichkeiten: $b_{im} = p(v_m | s_i)$

Matrix der Beobachtungs-wahrscheinlichkeiten:

$$\mathbf{B} = \left(\begin{array}{c|c|c|c} b_{11} & b_{21} & \cdots & b_{N1} \\ b_{12} & b_{22} & \cdots & b_{N2} \\ \vdots & \vdots & \ddots & \vdots \\ \underbrace{b_{1M}}_{=\mathrm{b}_1} & \underbrace{b_{2M}}_{=\mathrm{b}_2} & \cdots & \underbrace{b_{NM}}_{=\mathrm{b}_N} \end{array} \right)$$

zusammengefasste Parameter des HMMs: $\lambda = (\mathbf{e}, \mathbf{A}, \mathbf{B})$

Beobachtungs- bzw. Produktionswahrscheinlichkeit: $p(\mathbf{o}|\lambda)$

Dabei durchlaufene (verborgene) Zustandsfolge: $\mathbf{q} = (q_1, q_2, \ldots, q_T)$

Grundlegende Regeln der Wahrscheinlichkeitstheorie

1) Bedingte Wahrscheinlichkeit:

2) Totale Wahrscheinlichkeit:

Verarbeitung einer Beobachtung

Grundlegende Formel für die Verarbeitung einer Beobachtung:

$$
\begin{aligned}
p_t = p(o_t, s_i \rightarrow s_j) &= p(o_t \mid s_i \rightarrow s_j) \cdot p(s_i \rightarrow s_j) \\
&= p(o_t \mid s(t) = s_j) \cdot p(s(t) = s_j \mid s(t-1) = s_i) \\
&= a_{ij} \cdot b_j(o_t)
\end{aligned}
$$

Beobachtungswahrscheinlichkeit

$$p(\mathbf{o}, \mathbf{q}|\lambda_k) = e_{q_1} b_{q_1}(o_1) \cdot \underbrace{a_{q_1 q_2} b_{q_2}(o_2) \cdots a_{q_{t-1} q_t} b_{q_t} \cdots a_{q_{T-1} q_T} b_{q_T}(o_T)}_{= \prod_{t=2}^{T} a_{q_{t-1} q_t} b_{q_t}(o_t)}$$

$$\mathbf{q} = (s_{q_1}, s_{q_2}, \ldots, s_{q_T})$$

$$p(\mathbf{o}|\lambda_k) = \sum_{\text{alle Pfade q}} p(\mathbf{o}, \mathbf{q}|\lambda_k)$$

$$p(\mathbf{o}|\lambda_k) = \sum_{q \in Q} e_{q_1} b_{q_1}(o_1) \prod_{t=2}^{T} a_{q_{t-1} q_t} b_{q_t}(o_t)$$

$$\mathrm{OP}_B \sim 2T \cdot N^T$$

(≈ 10^{72} für N = 5 und T = 100)

Vorwärtsalgorithmus

Deutlich effizientere Möglichkeit zur Berechnung der Produktionswahrscheinlichkeit

$$p(\mathbf{o}|\lambda_k)$$

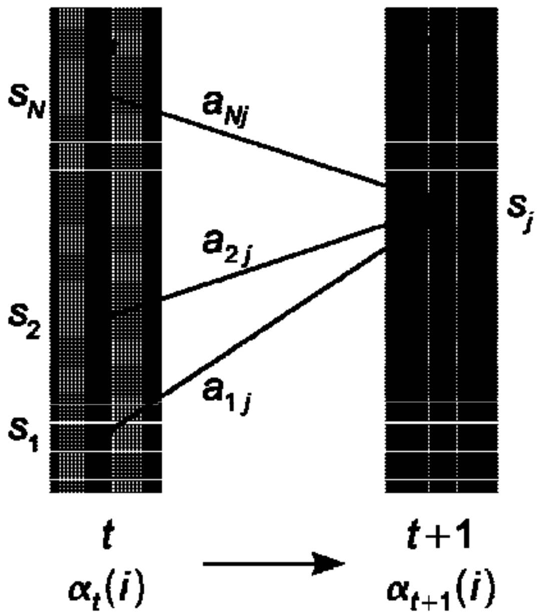
Basiert auf der iterativen Berechnung der folgenden Wahrscheinlichkeiten:

$$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, q_t = s_i|\lambda_k)$$

Zeitpunkt

Zustand

bis dahin gesehene Beobachtung



Iteration für Vorwärtsalgorithmus

$$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, q_t = s_i | \lambda_k)$$

$$\alpha_1(i) = e_i b_i(o_1), \qquad 1 \leq i \leq N$$

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(o_{t+1}), \qquad 1 \leq t \leq T-1; \quad 1 \leq j \leq N$$

$$P(\mathbf{o}|\lambda_k) = \sum_{i=1}^{N} \alpha_T(i)$$

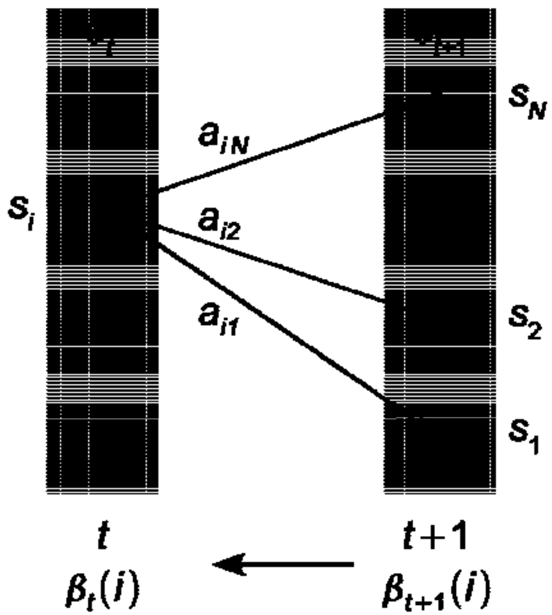
$$\mathrm{OP}_V \sim T \cdot N^2$$

(≈ 3000 für N = 5 und T = 100)

Rückwärts-Algorithmus

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T | q_t = s_i, \lambda_k)$$

Zeitpunkt   Zustand   verbleibende Beobachtung



Iteration für Rückwärts-Algorithmus

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T | q_t = s_i, \lambda_k)$$

$$\beta_T(i) = 1, \qquad 1 \leq i \leq N$$

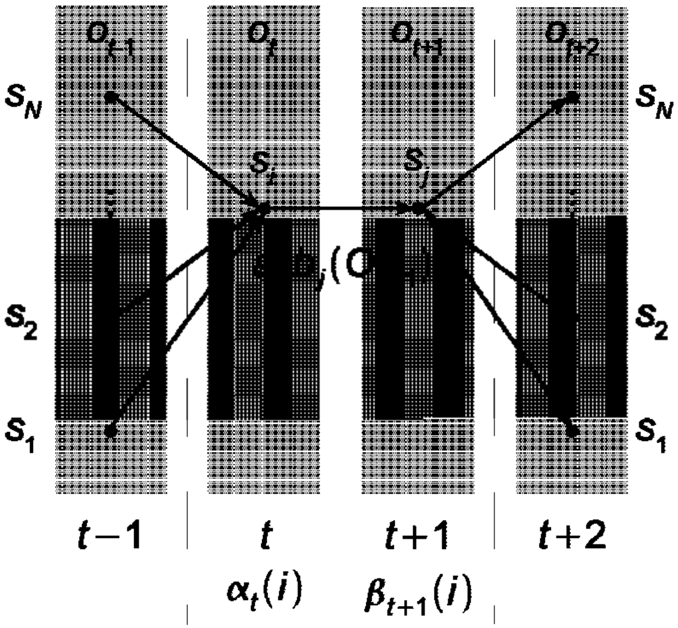
$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \qquad t = T-1, T-2, \ldots 1; \quad 1 \leq i \leq N$$

Kombination von α- und β-Wahrscheinlichkeiten führen zu Gesamtwahr-scheinlichkeiten der ganzen Beobachtungssequenz o = (o_1, o_2, ……o_T)

weitere ableitbare Wahrscheinlichkeiten

$$\gamma_t(i) = p(q_t = s_i | \mathbf{o}, \lambda_k)$$

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(\mathbf{o}|\lambda_k)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$$



Baum-Welch-Algorithmus

$$\xi_t(i,j) = P(q_t = s_i, q_{t+1} = s_j | \mathbf{o}, \lambda_k)$$

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(\mathbf{o}|\lambda_k)} = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \alpha_t(i) \beta_t(i)}$$

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

Hilfsgrößen für Baum-Welch

$$\sum_{t=1}^{T-1} \gamma_t(i) \equiv \text{„alle Aufenthalte im Zustand } s_i\text{“}$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) \equiv \text{„alle Übergänge vom Zustand } s_i \text{ zum Zustand } s_j\text{“}$$

Parameterschätzung mit Baum-Welch

$$\hat{e}_i \equiv \text{„Wahrscheinlichkeit zum Zeitpunkt } t = 1 \text{ im Zustand } s_i \text{ zu sein“} = \gamma_1(i)$$

$$\hat{a}_{ij} \equiv \frac{\text{„alle Übergänge vom Zustand } s_i \text{ zum Zustand } s_j\text{“}}{\text{„alle Aufenthalte im Zustand } s_i\text{“}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

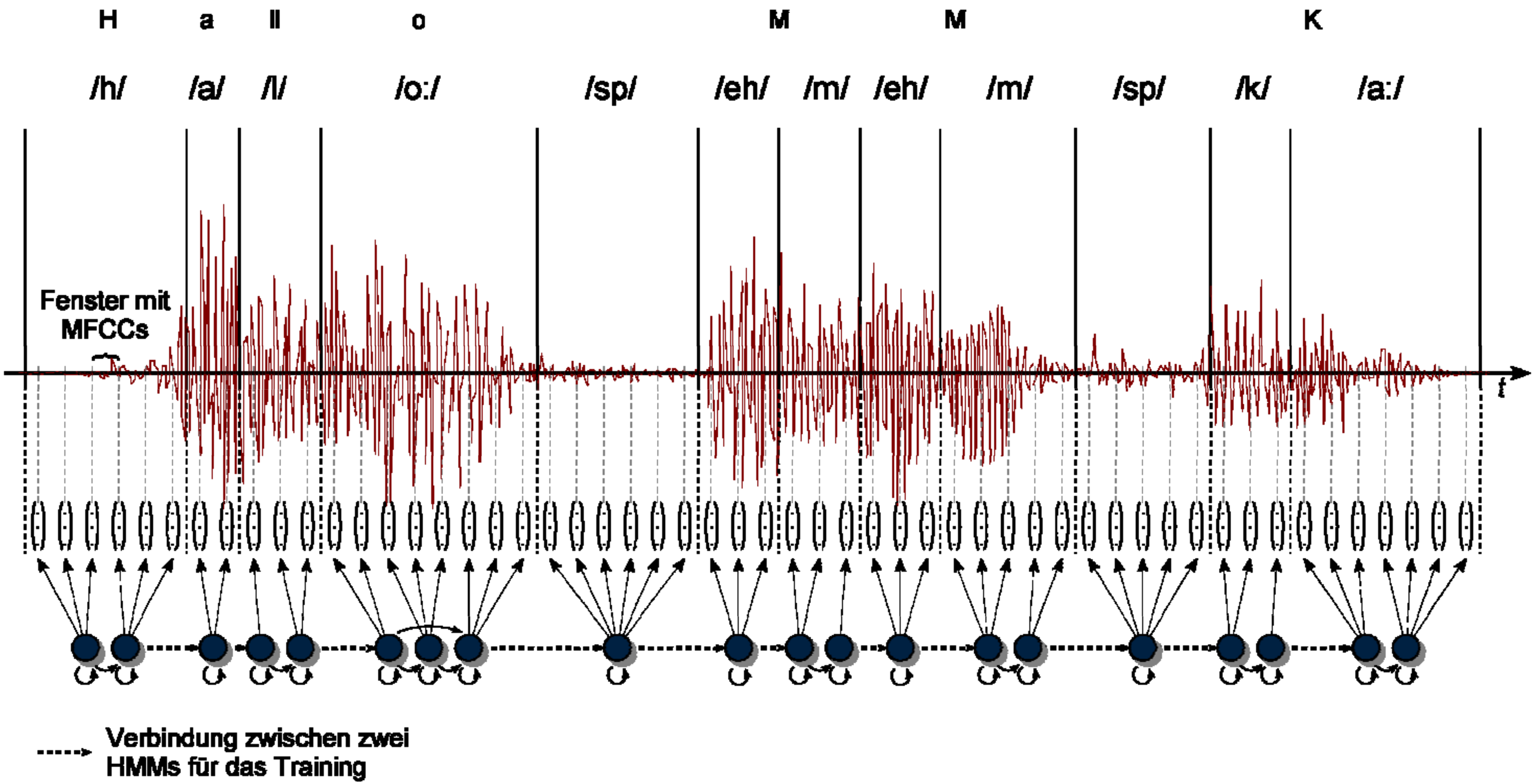
neue Ausgangswahrscheinlichkeiten mit Baum-Welch

$$\hat{b}_j(m) \equiv \frac{\text{„alle Aufenthalte im Zustand } s_j \text{ mit Beobachtung des Symbols } v_m\text{“}}{\text{„alle Aufenthalte im Zustand } s_j\text{“}} =$$

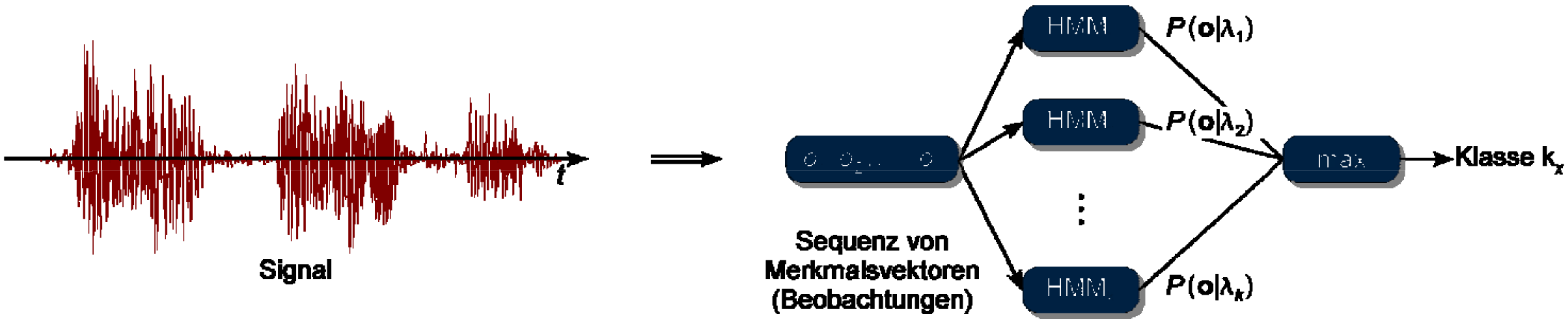
$$= \frac{\sum\limits_{t=1}^{T} \left(\gamma_t(j) \cdot \delta_{o_t v_m}\right)}{\sum\limits_{t=1}^{T} \gamma_t(j)}, \text{ mit Kronecker-Delta } \delta_{xy} = \begin{cases} 1, \text{ wenn } x = y \\ 0, \text{ wenn } x \neq y \end{cases}$$

$$\boxed{P(\mathbf{o}|\hat{\lambda}_k) \geq P(\mathbf{o}|\lambda_k)}$$

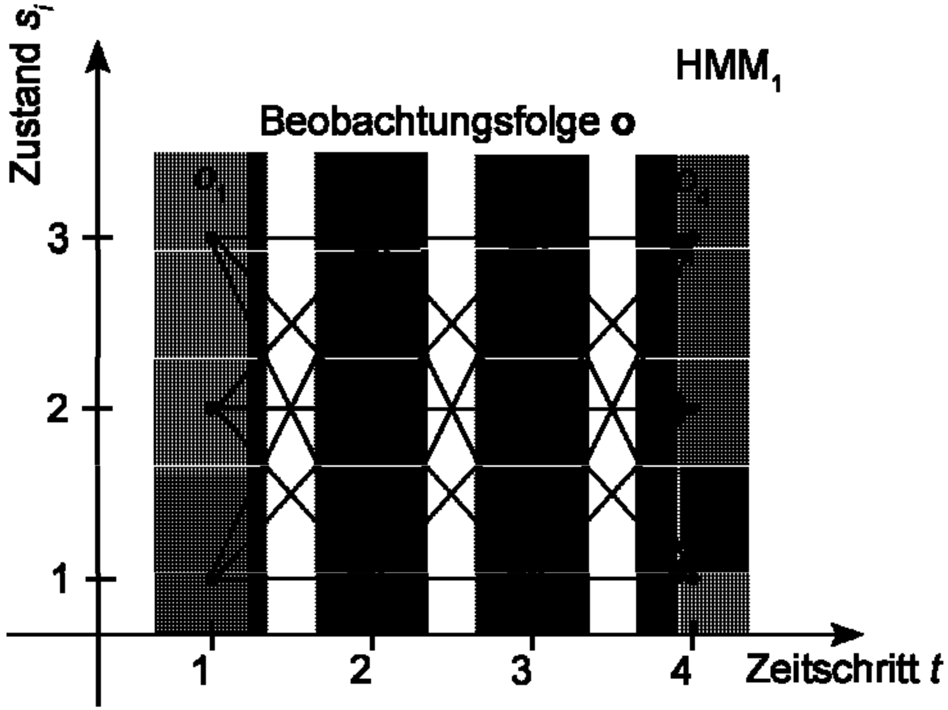
Training


H
a
ll
o
M
M
K
/h/
/a/
/l/
/o:/
/sp/
/eh/
/m/
/eh/
/m/
/sp/
/k/
/a:/
Fenster mit MFCCs
t
Verbindung zwischen zwei HMMs für das Training

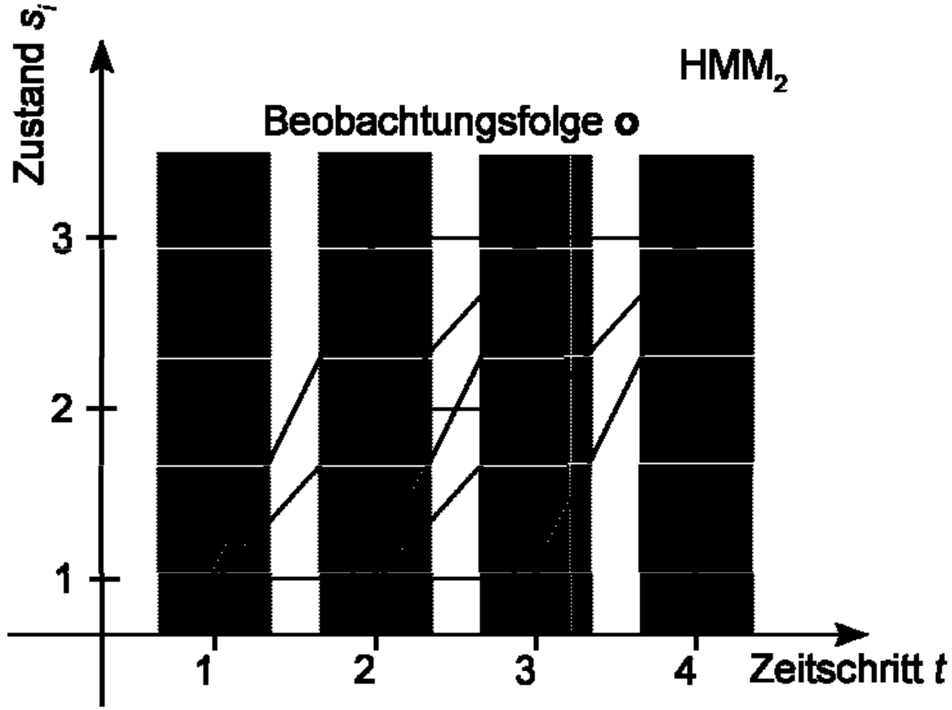
Einzelworterkennung



Trellis

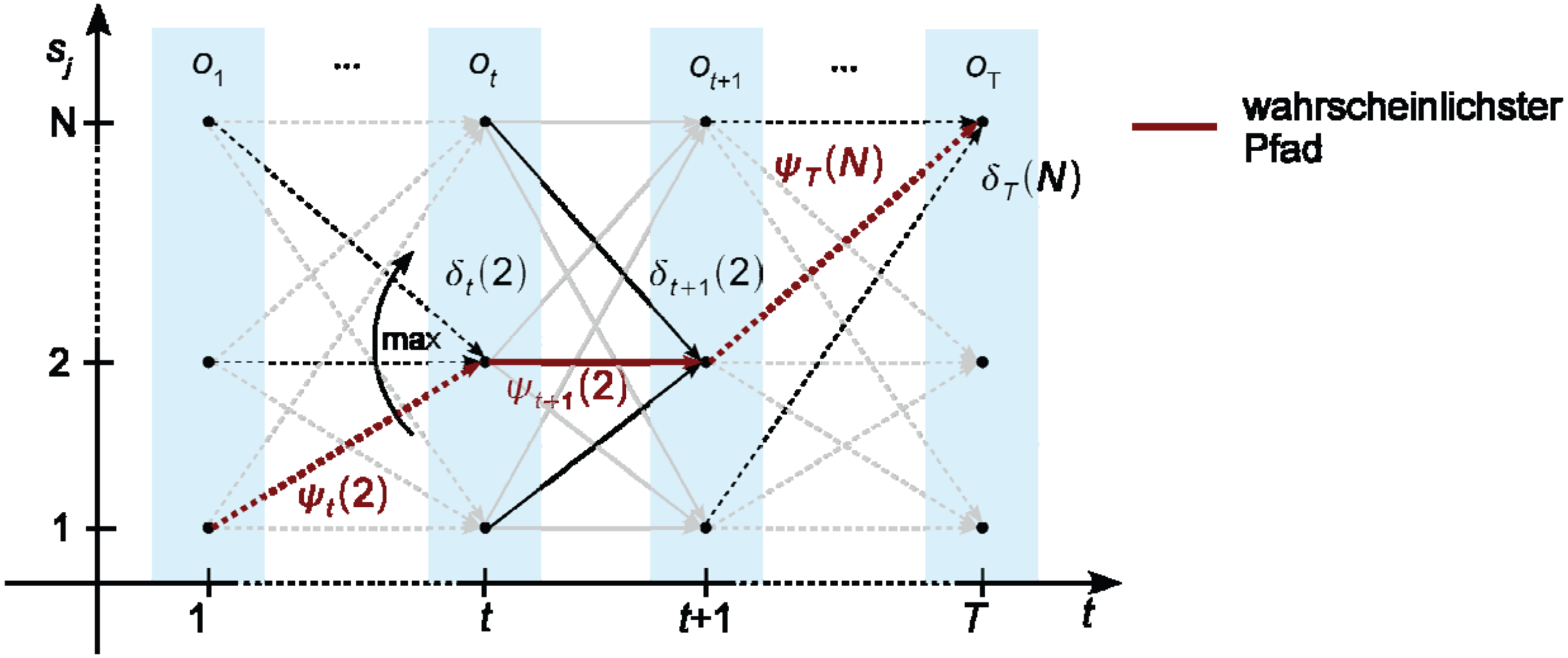


Zustand s_i
HMM$_1$
Beobachtungsfolge **o**
o_1
3
2
1
1
2
3
4
Zeitschritt t



Zustand s_i
HMM$_2$
Beobachtungsfolge **o**
3
2
1
1
2
3
4
Zeitschritt t

Viterbi-Algorithmus



Initialisierung:

$$\delta_1(i) = e_i b_i(o_1), \qquad 1 \leq i \leq N$$

$$\psi_1(i) = 0$$

Viterbi-Algorithmus

Rekursion:

$$\delta_t(j) = \max_{1 \leq i \leq N} \left[\delta_{t-1}(i) a_{ij}\right] b_j(o_t), \qquad 2 \leq t \leq T; \quad 1 \leq j \leq N$$

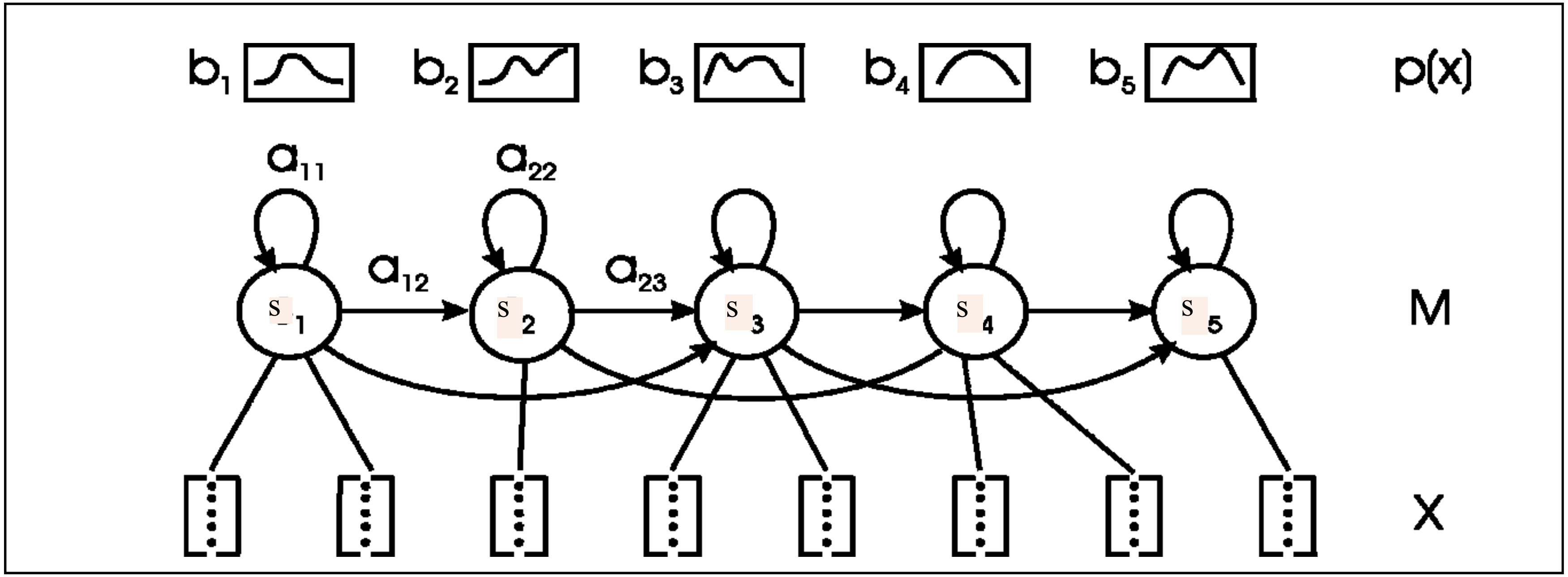
$$\psi_t(j) = \underset{1 \leq i \leq N}{\operatorname{argmax}} \left[\delta_{t-1}(i) a_{ij}\right], \qquad 2 \leq t \leq T; \quad 1 \leq j \leq N$$

Abschluss:

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)], \qquad q_T^* = \underset{1 \leq i \leq N}{\operatorname{argmax}} [\delta_T(i)]$$

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \ldots, 1$$

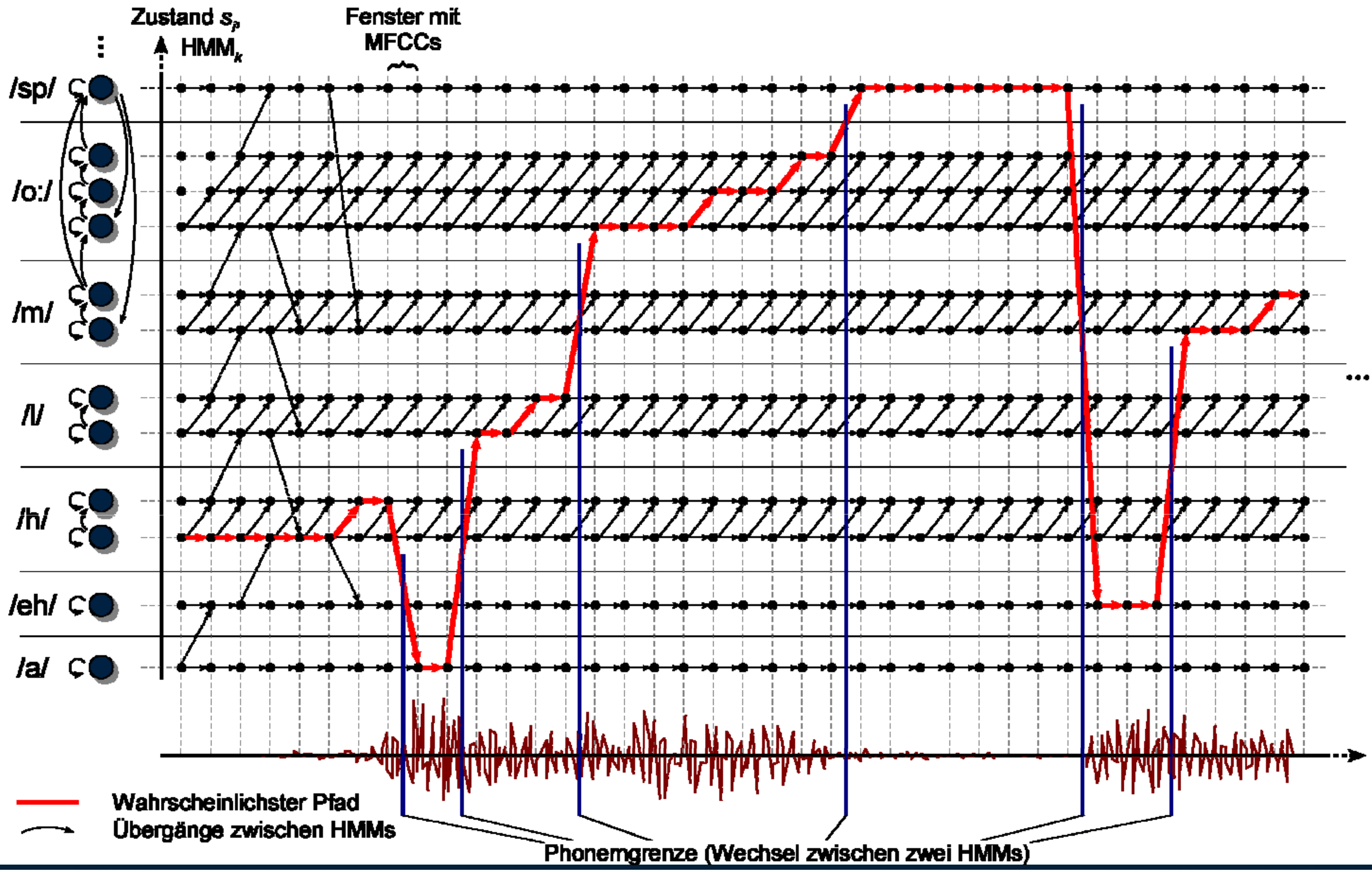
Ergebnis der Segmentierung



Phoneme

Phonem	Aussprache	Phonem	Aussprache	Phonem	Aussprache
/a/	Kampf	/a:/	Kahn	/ai/	weit
/au/	Haus	/ax/	mache	/b/	Ball
/d/	deutsch	/eh/	wenn	/eh:/	Affäre
/ey/	wen	/f/	fern	/g/	gern
/h/	Hand	/i/	Himmel	/i:/	Hier
/j/	Junge	/jh/	Joystick	/k/	Kind
/l/	links	/m/	matt	/n/	Nest
/ng/	lang	/o/	offen	/o:/	Ofen
/oe/	Hölle	/oe:/	Höhle	/oy/	freut
/p/	Paar	/r:/	rennen	/s/	fassen
/sh/	schön	/t/	Tafel	/u/	Mutter
/u:/	Mut	/v/	wer	/x/	lachen
/y/	Typ	/y:/	Kübel	/z/	singen
/zh/	Ingenieur	/sp/	„short pause“	/sil/	„silence“

Erkennung fließend gesprochener Sprache



Dekodierung

Generelle Formel hierfür:

$$M^* = \underset{M}{\operatorname{argmax}} \left\{ p(M \mid O) \right\}$$

mit

$$p(M \mid O) = \frac{p(O \mid M) \cdot P(M)}{p(O)}$$

p(O) beeinträchtigt nicht die korrekte Wahl der optimalen Modellfolge:

$$M^* = \underset{M}{\operatorname{argmax}} \; p(O \mid M) \cdot P(M)$$

aus Viterbi-Algorithmus

aus Sprachmodell

N-Gramme

- ✓ Sinnvoll für Erkennung von Modellsequenzen (z.B. Sprache, Handschrift, Zeichensprache ….)
- ✓ Wahrscheinlichkeit des Auftritts einer beliebigen Modellsequenz P(M) der Länge L kann ausgedrückt werden als:

$$P(M) = P(M_1) \cdot P(M_2 \mid M_1) \cdot P(M_3 \mid M_2 M_1) \cdot ..... \cdot P(M_L \mid M_1 M_2 \dots M_{L-1})$$

$$= \prod_{i=1}^{L} P(M_i \mid M_1 \dots M_{i-1})$$

- ✓ Vereinfachung: nur Betrachtung von bedingten Wahrscheinlichkeiten mit n-1 Vorgängern
- ✓ Bezeichnung hierfür: n-Grammme, z.B.

$$p(M_\ell \mid M_{\ell-1} M_{\ell-2} \dots M_{\ell-n+1})$$

Bigramme

- ✓ Notwendigkeit der Beschränkung auf kleine n (1 < n < 5):
  - ➢ Kontexteinfluss für große n wird immer geringer
  - ➢ Extreme Speicherplatz- und Zugriffsprobleme für große n
  - ➢ Beispiel: V=10.000 n=3 #n-grams=$10.000^3 = 10^{12}$
- ✓ Unigramme: Modellieren jede Einheit ohne Kontext, d.h.:

$$P(M) = P(M_1) \cdot P(M_2) \ldots \cdot P(M_L) = \prod_{i=1}^{L} P(M_i)$$

- ✓ Bigramme:

$$P(M) = P(M_1) \cdot P(M_2 \mid M_1) \cdot P(M_3 \mid M_2) \ldots \cdot P(M_L \mid M_{L-1}) = \prod_{i=1}^{L} P(M_i \mid M_{i-1})$$

Trigramme

- ✓ Trigramme: $P(M) = P(M_1) \cdot P(M_2 \mid M_1) \cdot P(M_3 \mid M_2 M_1) \ldots \cdot P(M_L \mid M_{L-1} M_{L-2})$

$$= \prod_{i=1}^{L} P(M_i \mid M_{i-1} M_{i-2})$$

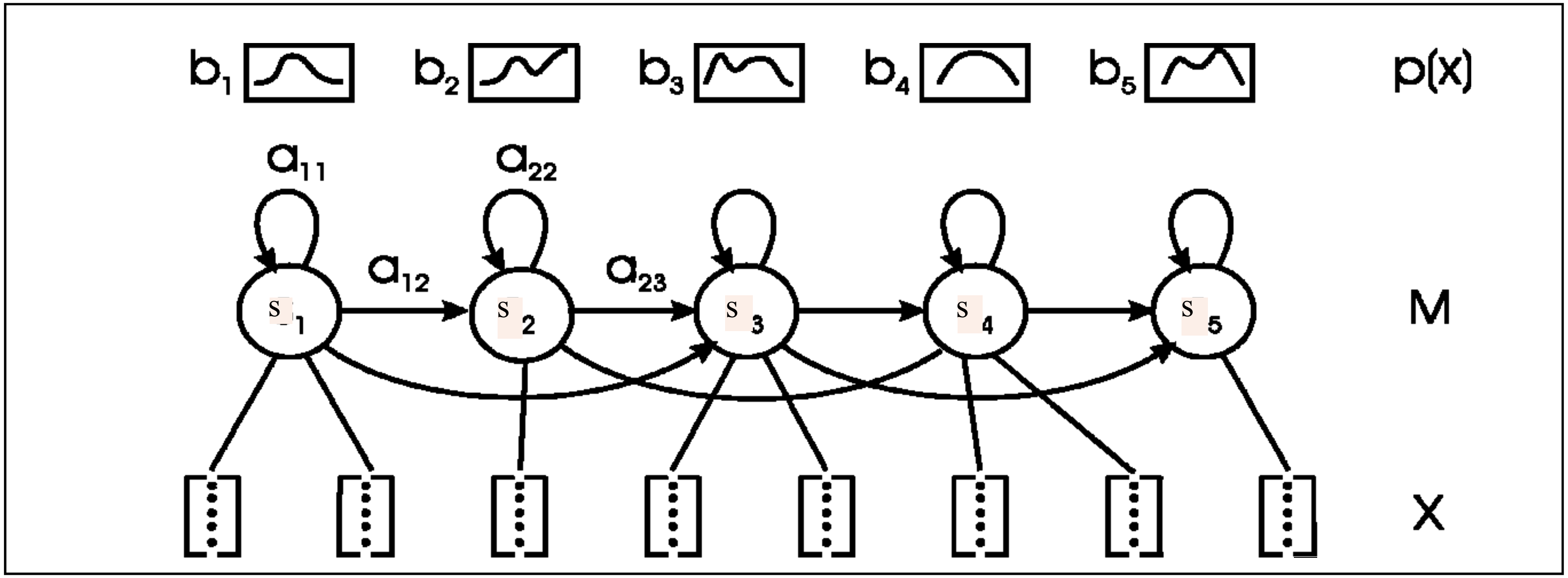
- ✓ Schätzung durch Wortzählung in großen Textdatenbasen und Anwendung der Bayes‘schen Regel:

$$P(M_\ell, M_{\ell\text{-}1} \ldots M_{\ell\text{-}n+1}) = P(M_\ell \mid M_{\ell\text{-}1} \ldots M_{\ell\text{-}n+1}) \cdot P(M_{\ell\text{-}1} \ldots M_{\ell\text{-}n+1})$$

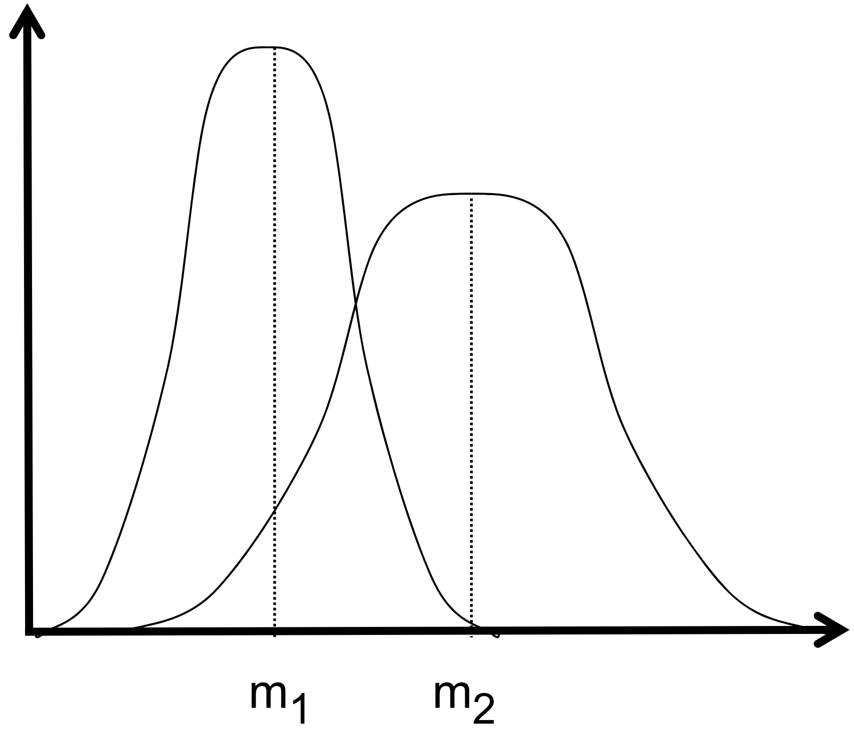
$$\Rightarrow P(M_\ell \mid M_{\ell-1} \ldots M_{\ell-n+1}) = \frac{P(M_\ell, M_{\ell\text{-}1} \ldots M_{\ell\text{-}n+1})}{P(M_{\ell\text{-}1} \ldots M_{\ell\text{-}n+1})} = \frac{\#(M_{\ell-n+1} \ldots M_{\ell-1} M_\ell)}{\#(M_{\ell-n+1} \ldots M_{\ell-1})}$$

- ✓ z.B. für Bigramme: $P(M_\ell \mid M_{\ell-1}) = \dfrac{\#(M_{\ell-1} M_\ell)}{\#(M_{\ell-1})}$

HMMs mit kontinuierlichen Ausgangsverteilungen

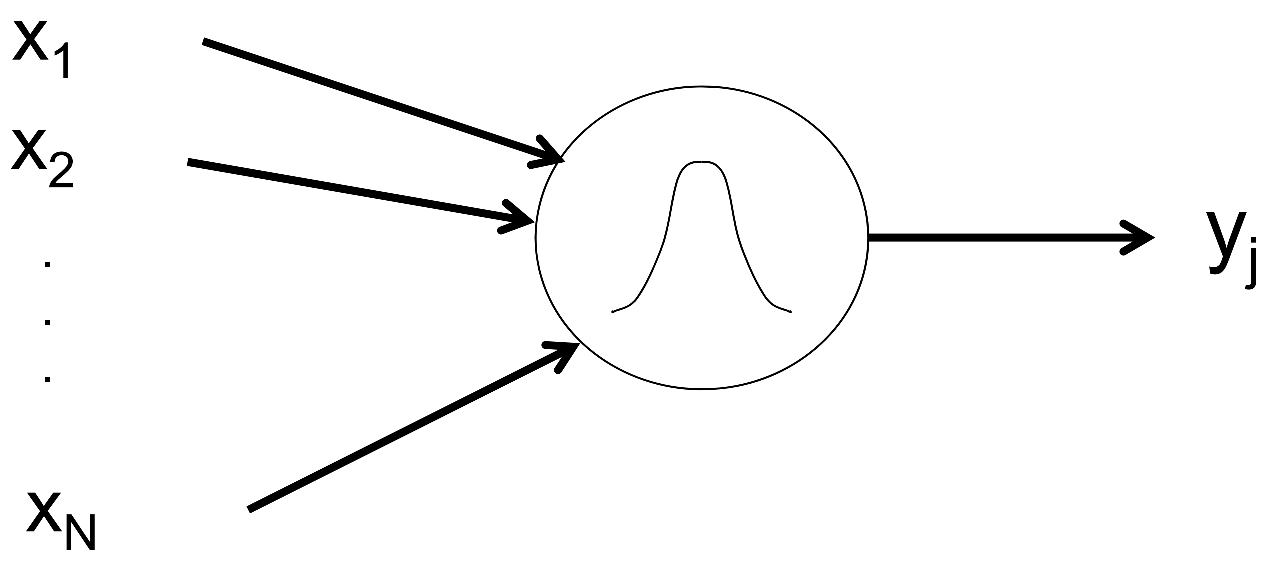


Gaußverteilung für Merkmalsvektoren



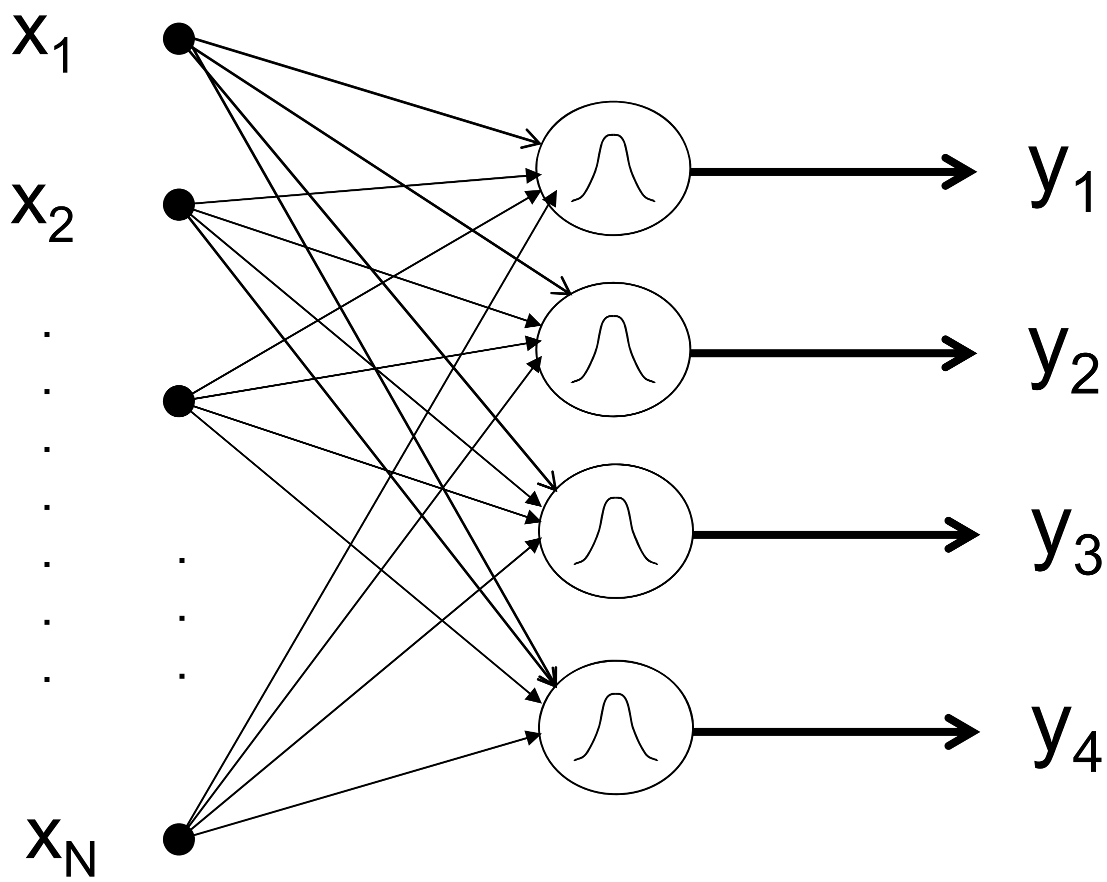
$$p(\underline{y} \mid K) = \frac{1}{\sqrt{(2\pi)^d \cdot \left| C_y \right|}} \cdot e^{-\frac{1}{2}(\underline{y}-\underline{m}_y)^T C_y^{-1} (\underline{y}-\underline{m}_y)}$$

Einzelnes Gauß-Element

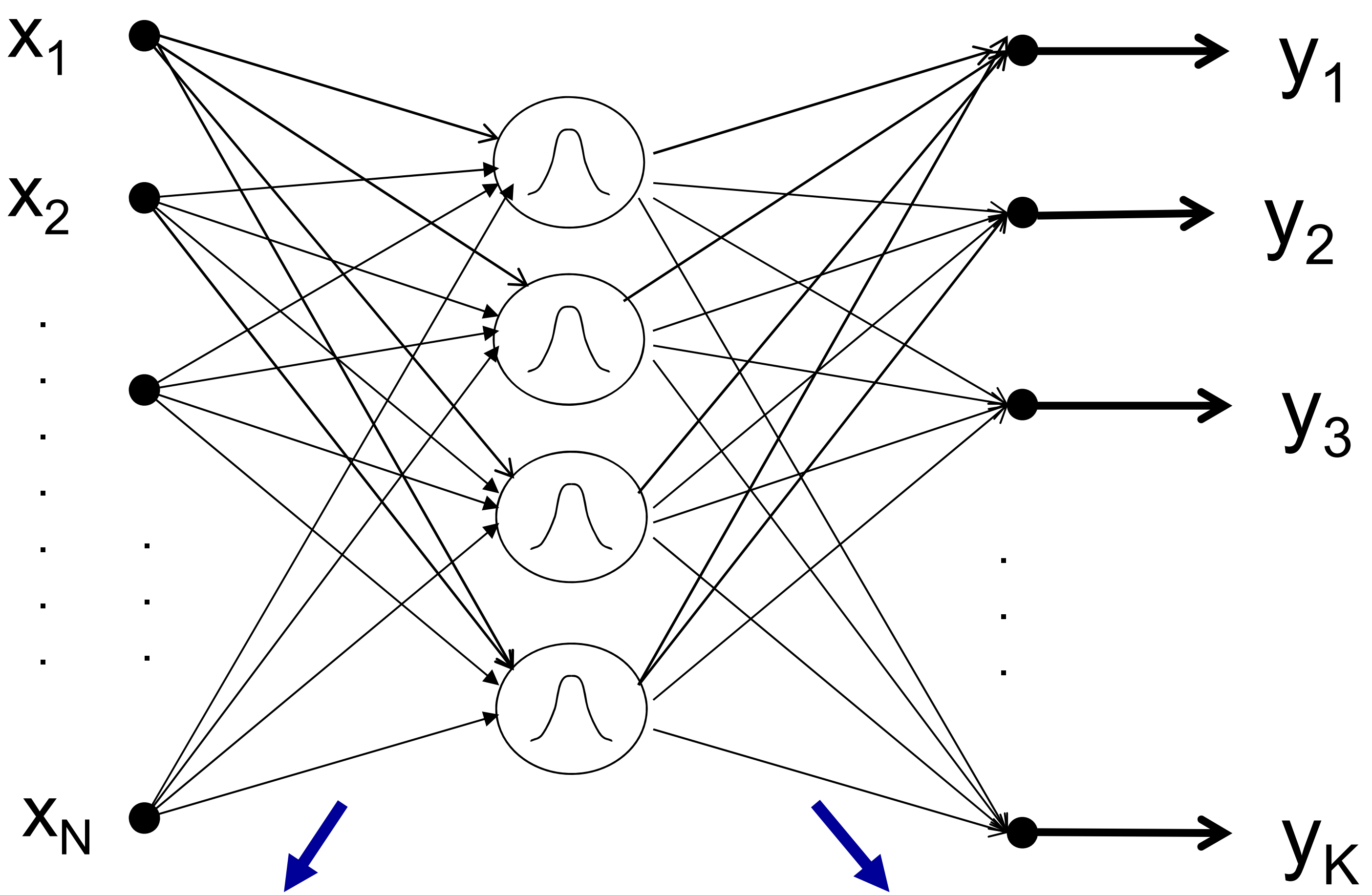


$$y_j = p(\underline{x} \mid j) = \frac{1}{\sqrt{(2\pi)^N \cdot \left|C_j\right|}} \cdot e^{-\frac{1}{2}(\underline{x}-m_j)^T C_j^{-1} (\underline{x}-m_j)}$$

Eine Gauß-Verteilung pro Zustand



Mehrere Gauß-Verteilungen pro Zustand



Netzparameter $\underline{m}_j$ und C_j

Netzparameter w_{jk}

Übergang zum RBF-Netzwerk

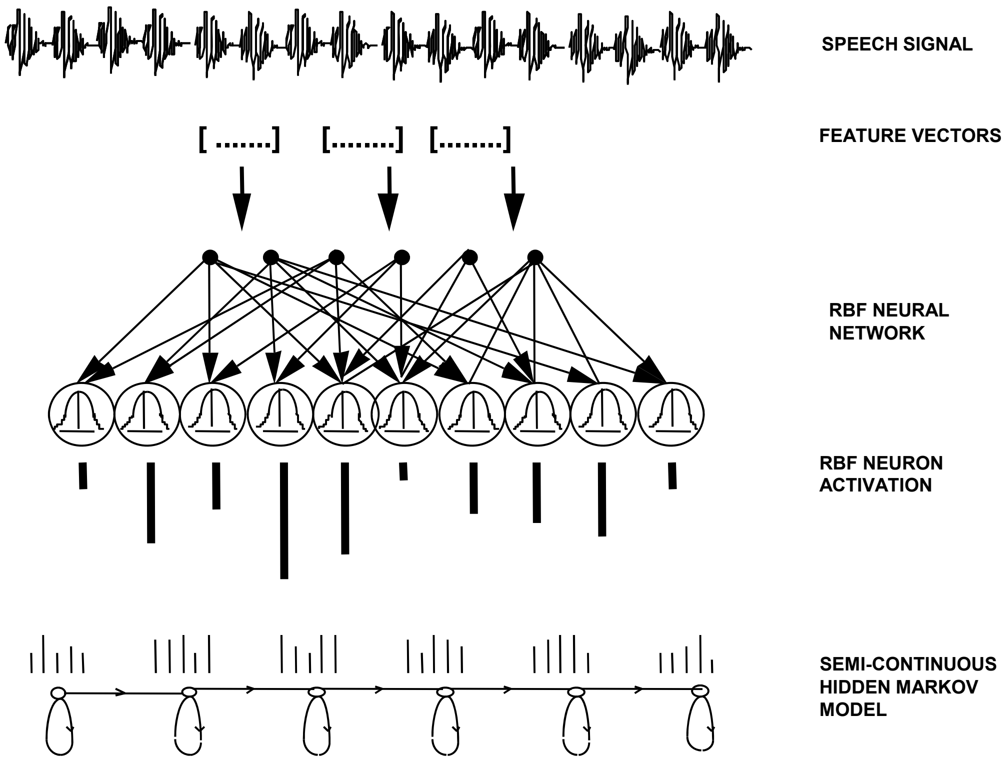
$$y_k = \sum_{j=1}^{J} w_{jk} \cdot \frac{1}{\sqrt{(2\pi)^N \cdot \left|C_j\right|}} \cdot e^{-\frac{1}{2}(\underline{x}-\underline{m}_j)^T C_j^{-1}(\underline{x}-\underline{m}_j)}$$

$$= \sum_{j=1}^{J} w_{jk} \cdot G_j(\underline{x}, \underline{m}_j, C_j)$$

j-te Gauß-Komponente

Dies entspricht der Formel einer **Gauß'schen Mischverteilung**

HMM mit Gauß'schen Mischverteilungen



$$p(x \mid s_i) = \sum_{j=1}^{J} w_{ji} \cdot G_j(\underline{x}, \underline{m}_j, C_j)$$

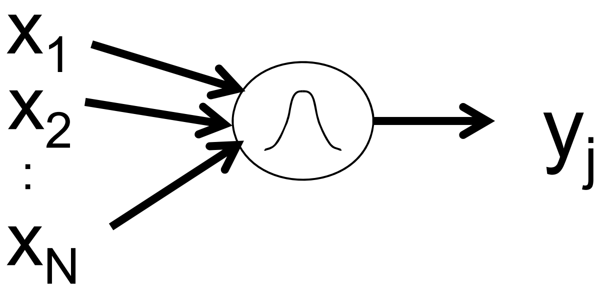
mit j-ter Gauß-Komponente:

$$G_j = \frac{1}{\sqrt{(2\pi)^d \cdot |W_j|}} \cdot e^{-\frac{1}{2}(\underline{x}-\underline{m}_j)^T W_j^{-1} (\underline{x}-\underline{m}_j)}$$

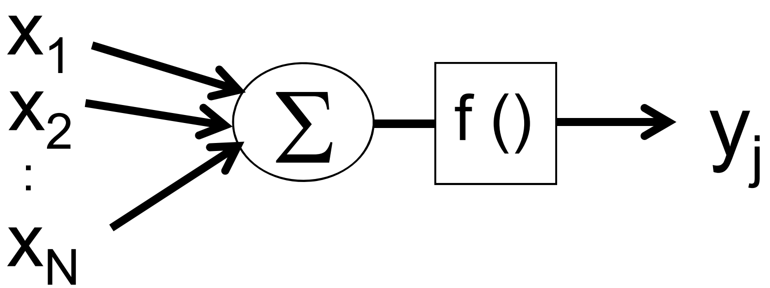
Übergang von RBF- zu MLP-Netz

MLP = Multilayer-Perceptron, entsteht

durch Ersetzen der RBF-Neuronen:



durch Perzeptron:



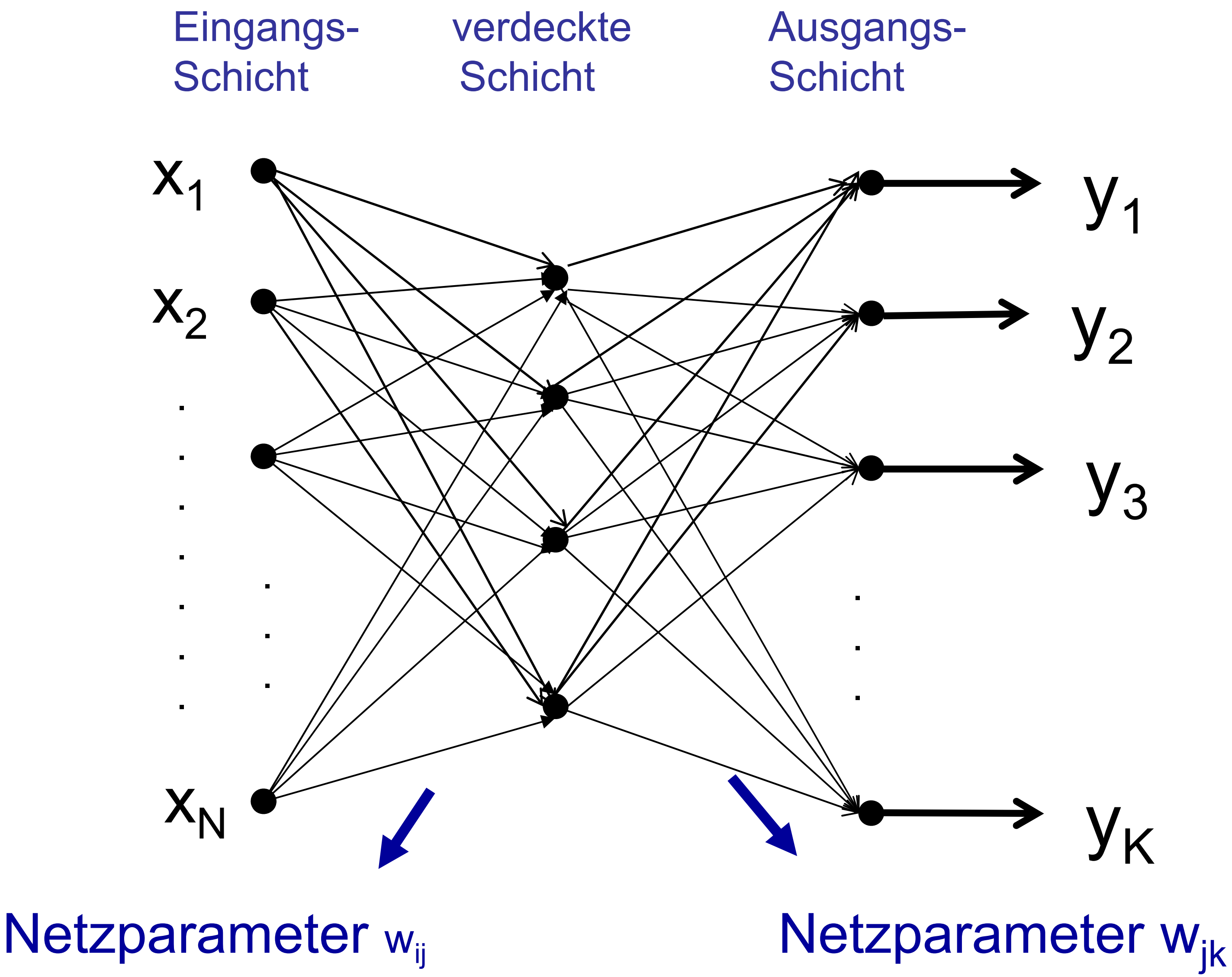
mit:

$$y_j = f\left(\sum_{i=1}^{N} w_i \cdot x_i\right)$$

und der nichtlinearen Funktion:

$$f(\Sigma) = \frac{1}{1 + e^{-\Sigma}}$$

MLP Neuronales Netzwerk



Training von Neuronalen Netzen

Trainingsdaten mit Klassenzugehörigkeit: $[\,\underline{x}(i), \Omega(i)\,]$, $i = 1, \ldots\ldots, I$

Klassenzugehörigkeit des i-ten Trainingsvektors ergibt den Zielwert des k-ten Ausgangsneurons zu:

$$\hat{y}_k(i) = \begin{cases} 1 & \text{für} \quad \Omega(i) = k \\ 0 & \text{sonst} \end{cases}$$

Gesamtfehler für alle Ausgangsneuronen und alle Trainingsbeispiele:

$$\varepsilon = \frac{1}{2} \sum_{i=1}^{I} \sum_{k=1}^{K} \left[\hat{y}_k(i) - y_k(i) \right]^2$$

Iterative Gewichtsverbesserung:

$$w_{nm}^{neu} = w_{nm}^{alt} - \alpha \cdot \frac{\partial \varepsilon}{\partial w_{nm}}$$

➔ Komplexere Berechnung der Ableitungen wg. Nichtlinearität f() und Gewichtungen w_{ij} der verdeckten Schicht

➔ **Error-Backpropagation**

Training auf Wahrscheinlichkeiten

Fehlerkriterium für das Training:

$$\varepsilon = E\{ \sum_{i=1}^{N} [\hat{y}_i(k) - y_i(k)]^2 \}$$

Minimum dieses Kriteriums:

$$\frac{\partial \varepsilon}{\partial y_i} = -2E\{\hat{y}_i - y_i\} = 0 \;\Rightarrow\; E\{y_i\} = E\{\hat{y}_i\} \quad \text{mit} \quad E\{\hat{y}_i\} = \frac{k_i}{K}$$

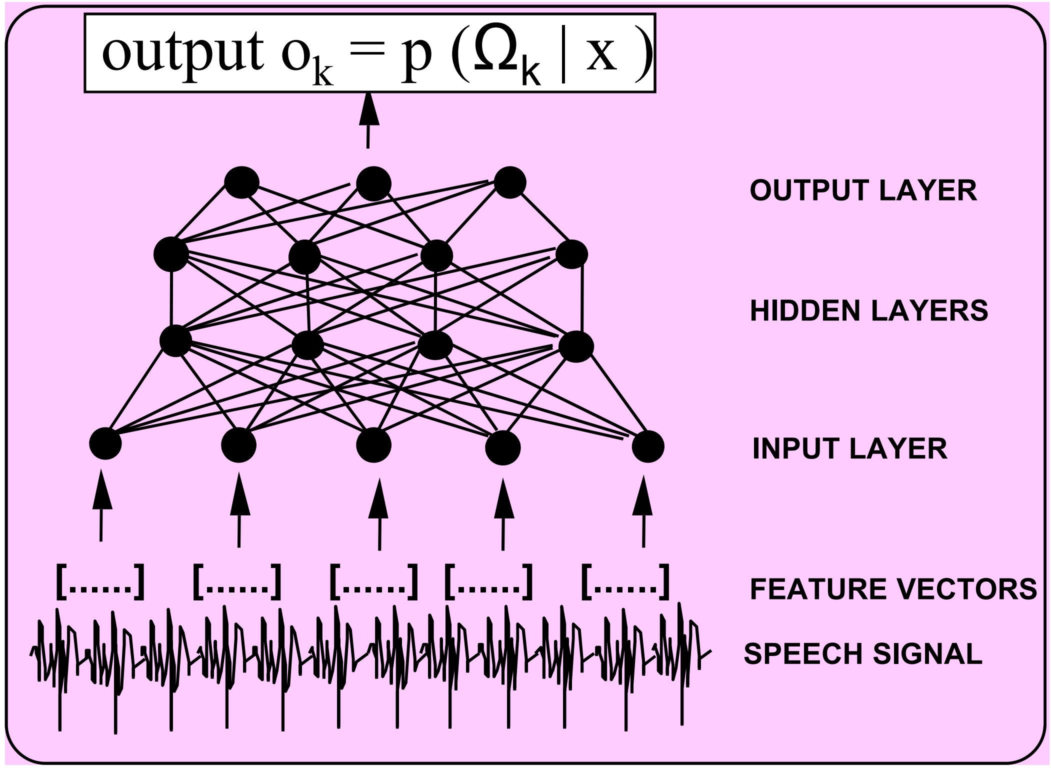
Resultierende Interpretation der Aktivierungen der Ausgangsschicht:

$$E\{y_i\} = \frac{k_i}{K} = p(\Omega_i | X) \qquad \Rightarrow\Rightarrow \qquad d_i = p(\Omega_i | \underline{x})$$

Voraussetzungen hierfür

- Präsentation der Sollwerte in binärer Form
- Erreichen eines möglichst globalen Minimums des Trainingsfehlers
- Ausreichend viele Trainingsdaten
  ➔ Big Data
- Genügend hohe Komplexität des Klassifikators um dieses Trainingsziel zu erreichen
  ➔ Deep Learning

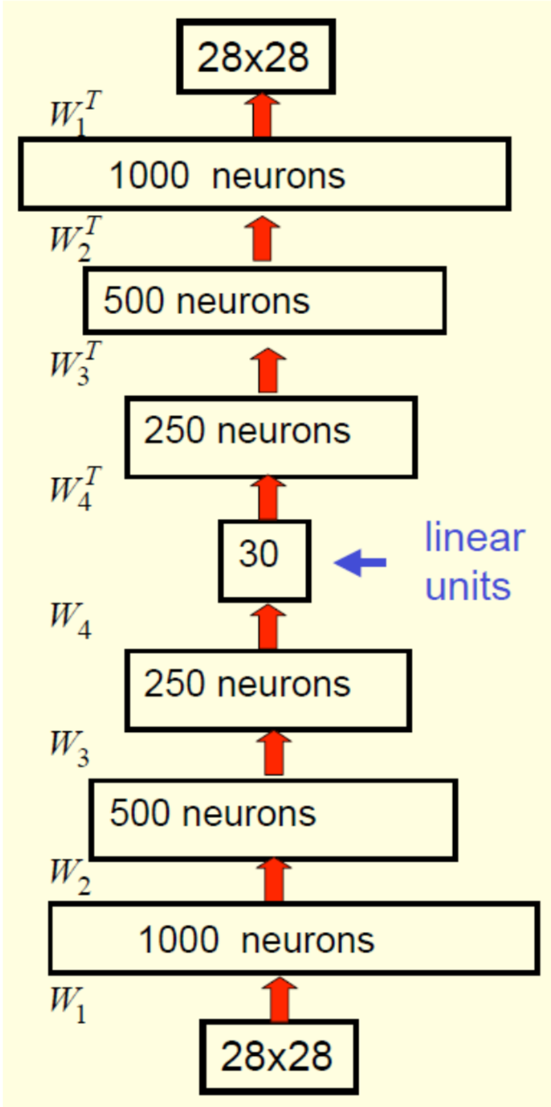
Generierung von Aposteriori-Wahrscheinlichkeiten



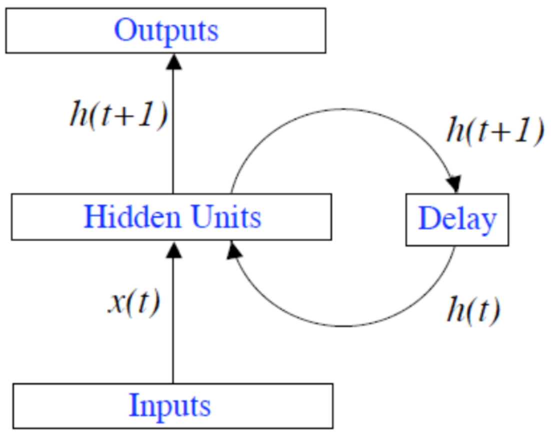
Softmax-Funktion in Ausgangs-schicht:

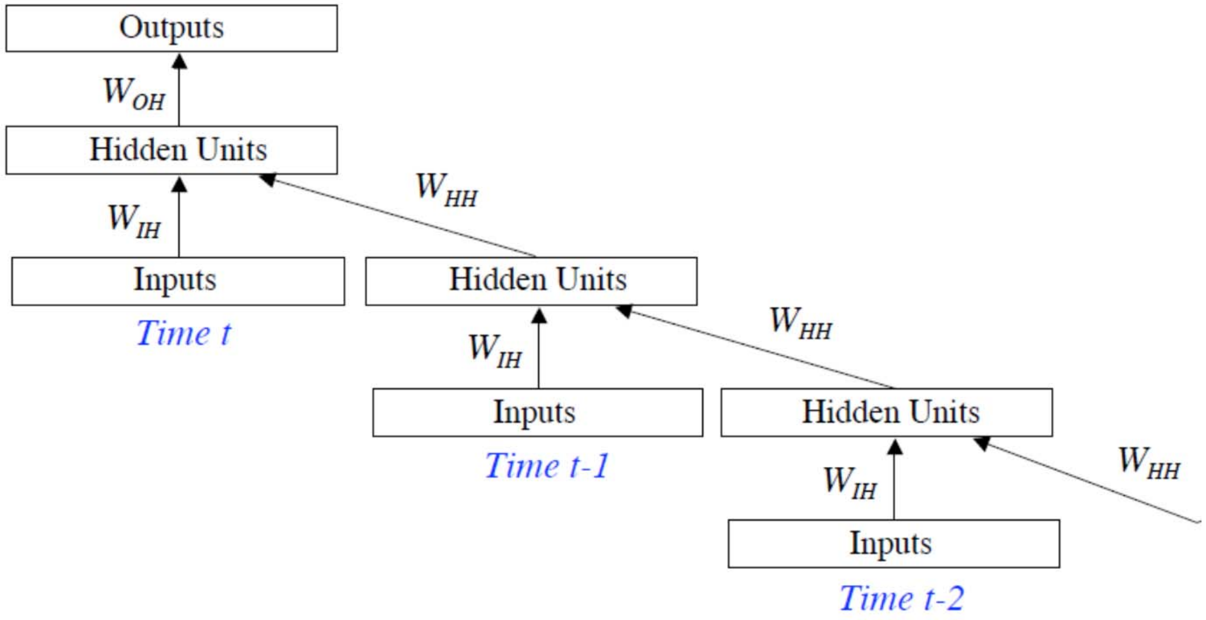
$$o_k = softmx\ (y_k) = \frac{e^{y_k}}{\sum_j e^{y_j}}$$

Deep Learning Ansatz

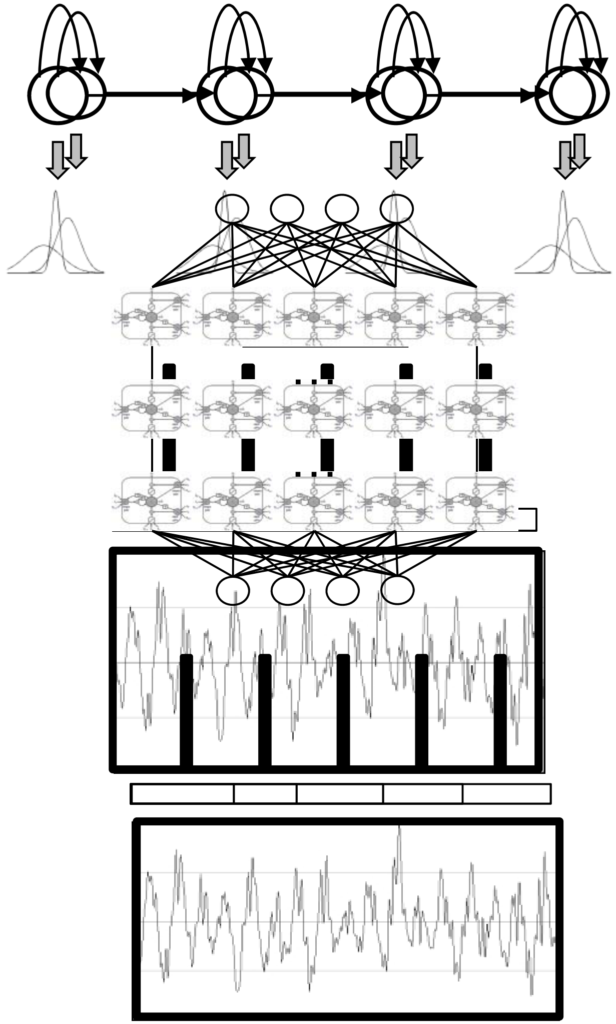


Rekurrente Neuronale Netze





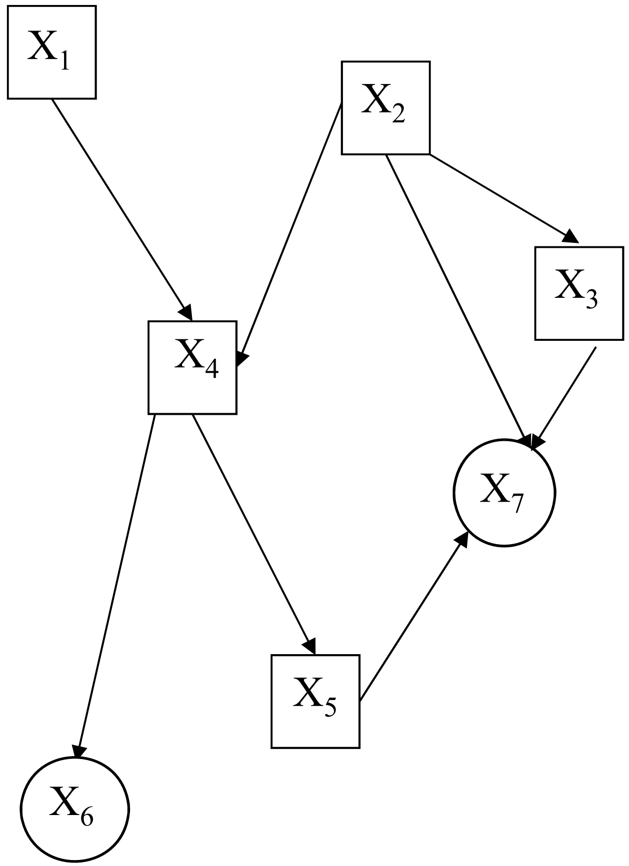
Hybride HMM-NN-Systeme



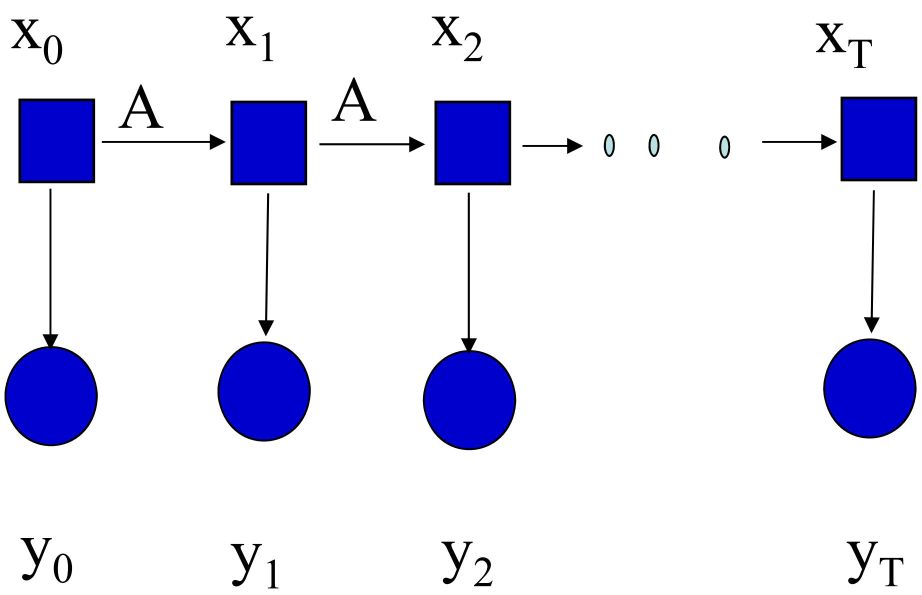
Graphische Modelle (GM)

Verbundwahrscheinlichkeit:

$$p(x_1,\ldots, x_6, x_7) = p(x_4|x_1,x_2) \cdot p(x_6|x_4) \cdot p(x_5|x_4) \cdot p(x_3|x_2) \cdot p(x_7|x_2,x_3,x_5) \cdot p(x_2) \cdot p(x_1)$$

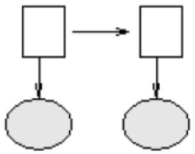


HMM als graphisches Modell

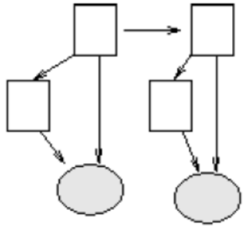


$$p(x,y) = p(x_0)\prod_{t=0}^{T} p(x_{t+1} \mid x_t)\prod_{t=0}^{T} p(y_t \mid x_t)$$

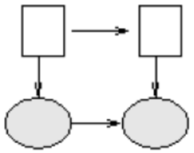
Populäre Graphische Modelle



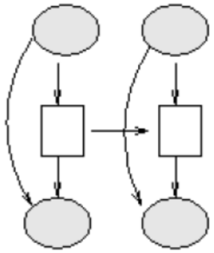
HMM with Gaussian output



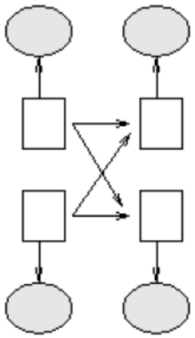
HMM with mixture of Gaussians output



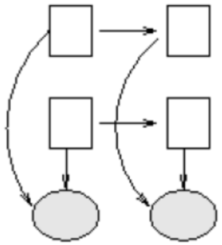
Auto Regressive HMM



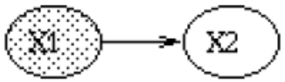
Input–output HMM



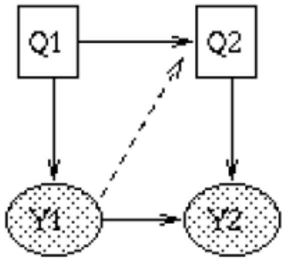
Coupled HMM



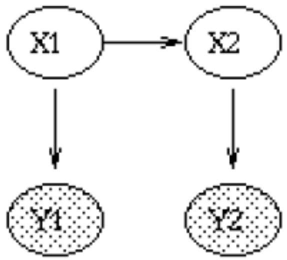
Factorial HMM



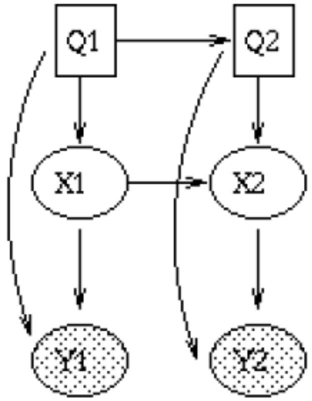
Auto Regressive model AR(1)



Switching AR model



Kalman filter model



Switching Kalman filter

Probabilistische Interferenz

Generell:

$$p(x_t \mid Y_1^t) = p(y_t \mid x_t) \cdot \int p(x_t \mid x_{t-1}) \cdot p(x_{t-1} \mid Y_1^{t-1}) \; dx_{t-1}$$

HMM-FB-Algorithmus:

$$\alpha_t(j) = p(o_t \mid x_t) \cdot \sum_i a_{ij} \cdot \alpha_{t-1}(i)$$

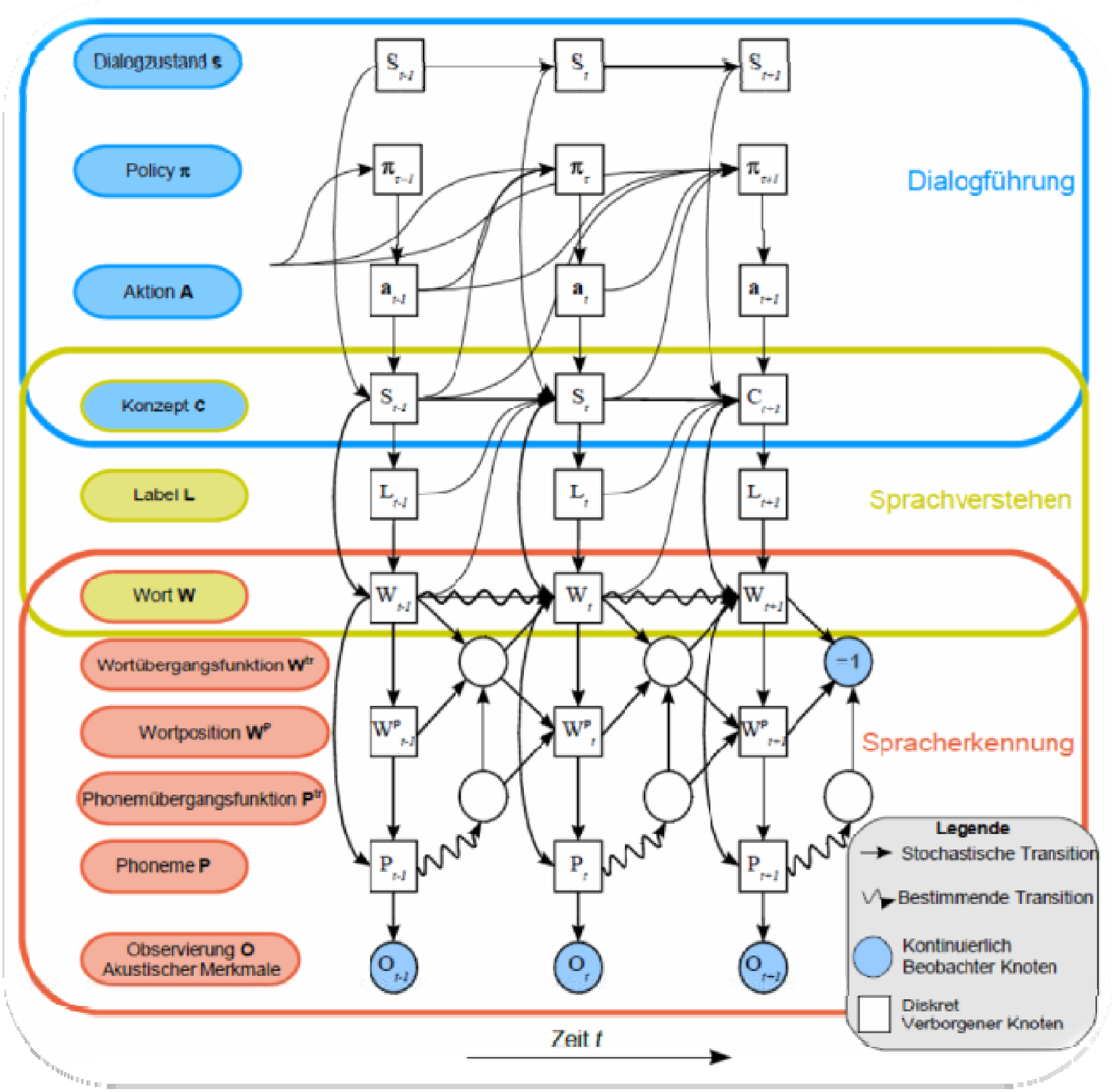
Kalman-Filter:

$$x_t = Ax_{t-1} + Bw_t \qquad y_t = Cx_t + Dv_t$$

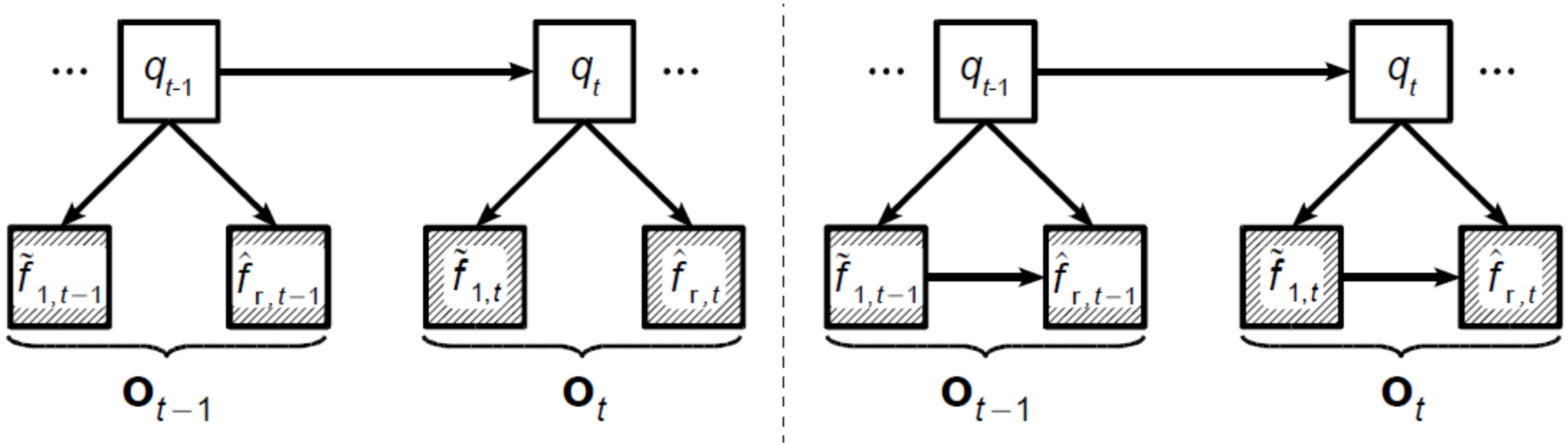
$$p(x_t \mid x_{t-1}) = \exp\left\{ -(x_t - Ax_{t-1})^T \Psi^{-1} (x_t - Ax_{t-1}) \right\}$$

$$p(y_t \mid x_t) = \exp\left\{ -(y_t - Cx_t)^T \Psi^{-1} (y_t - Cx_t) \right\}$$

Dialogsystem als Graphisches Modell



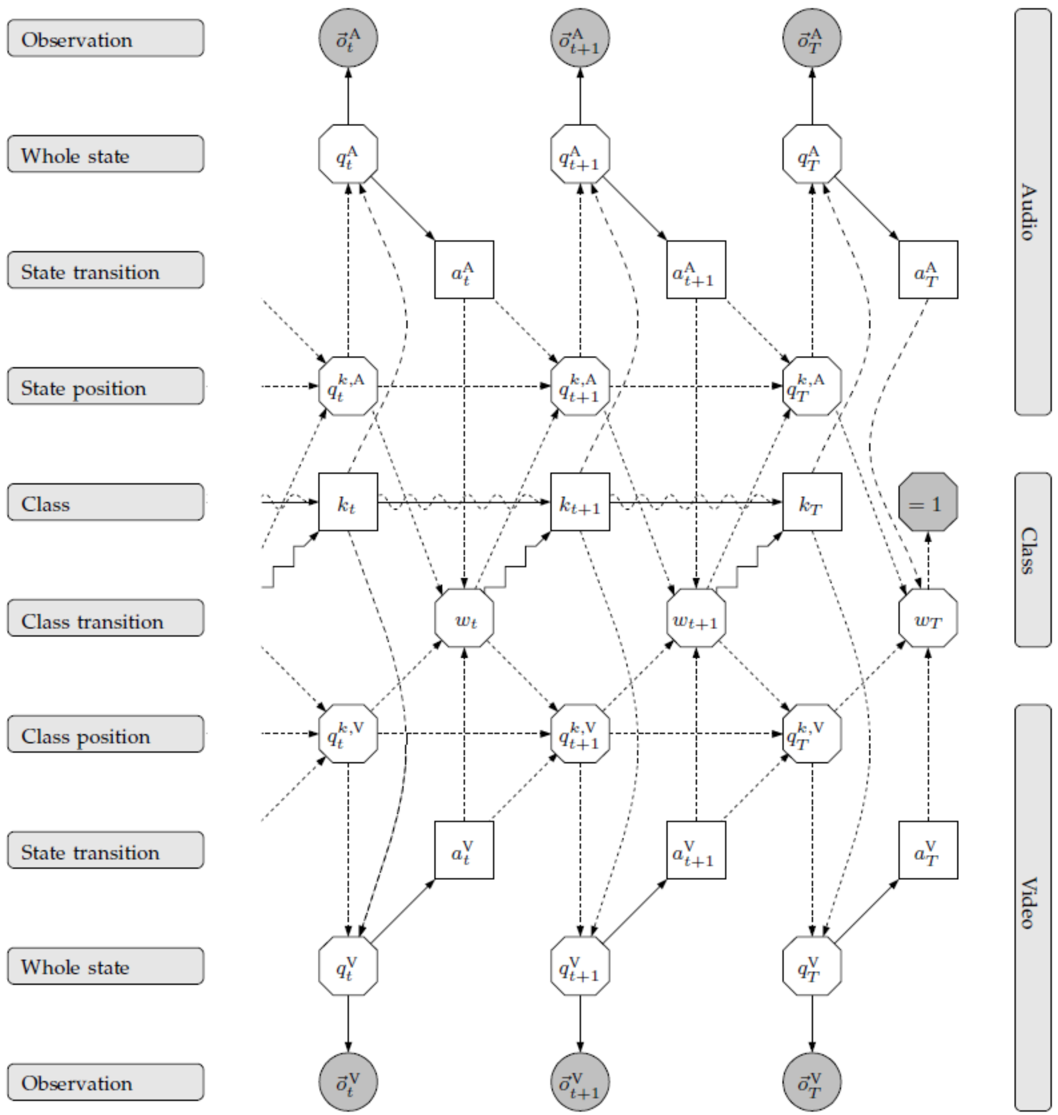
GM für die Modellierung der Abhängigkeit von Merkmalen



$$p(\mathbf{O}|\lambda) = \sum_{q \in \mathbf{Q}} \pi_{q_1} \underbrace{p(\tilde{f}_{1,1}|q_1)p(\hat{f}_{\mathrm{r},1}|q_1, \tilde{f}_{1,1}|q_1)}_{(**)} \cdot$$

$$\prod_{t=2}^{T} a_{q_{t-1}q_t} \underbrace{p(\tilde{f}_{1,t}|q_t)p(\hat{f}_{\mathrm{r},t}|\tilde{f}_{1,t}|q_t, q_t)}_{(**)}$$

GM für die audio-visuelle Erkennung



GM für synchrone Objektverfolgung und Gestikerkennnung