Modellbildung und Simulation

Sommersemester 2011

6. Vorlesung

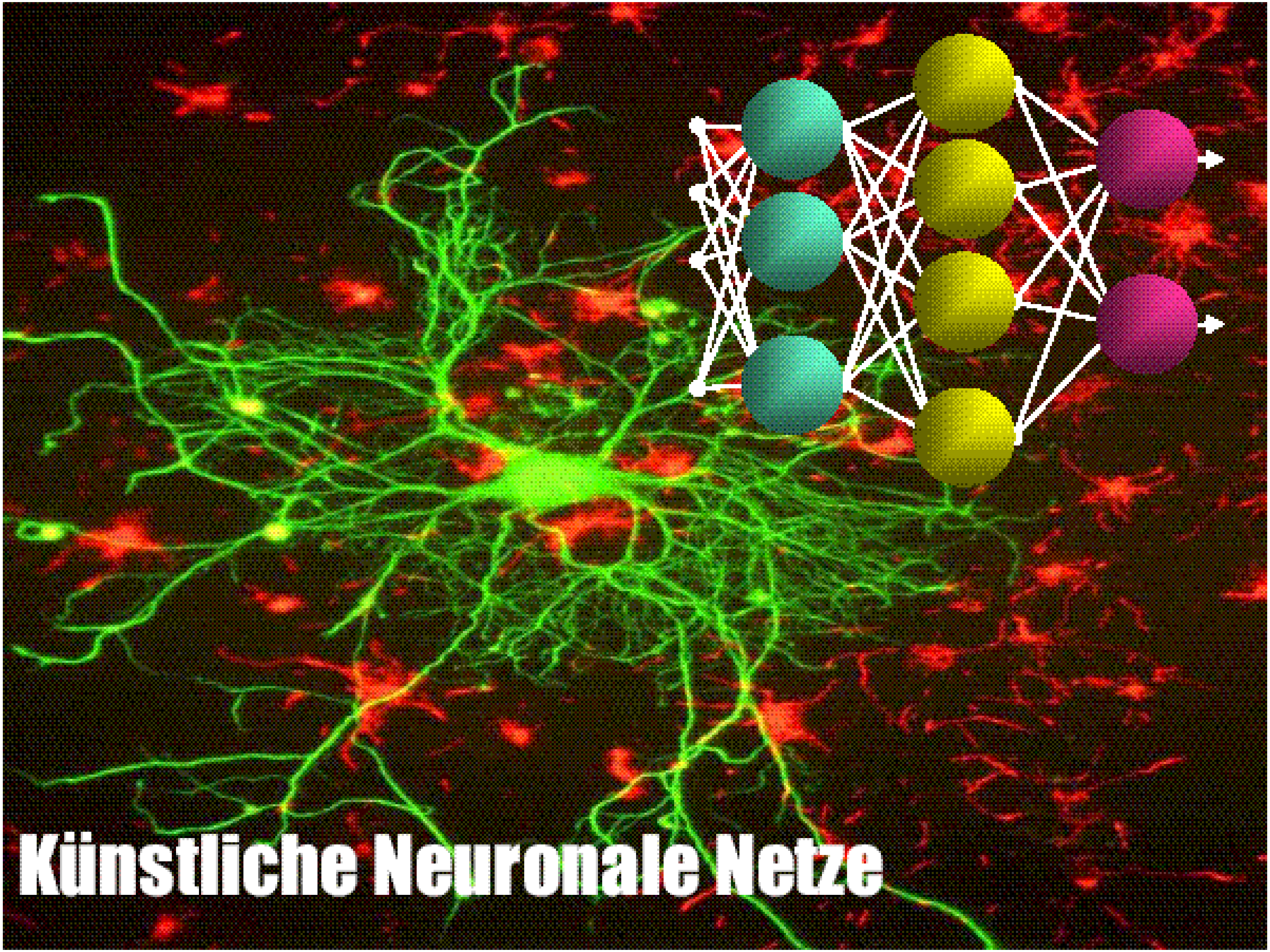
Klaus Kasper

Termine

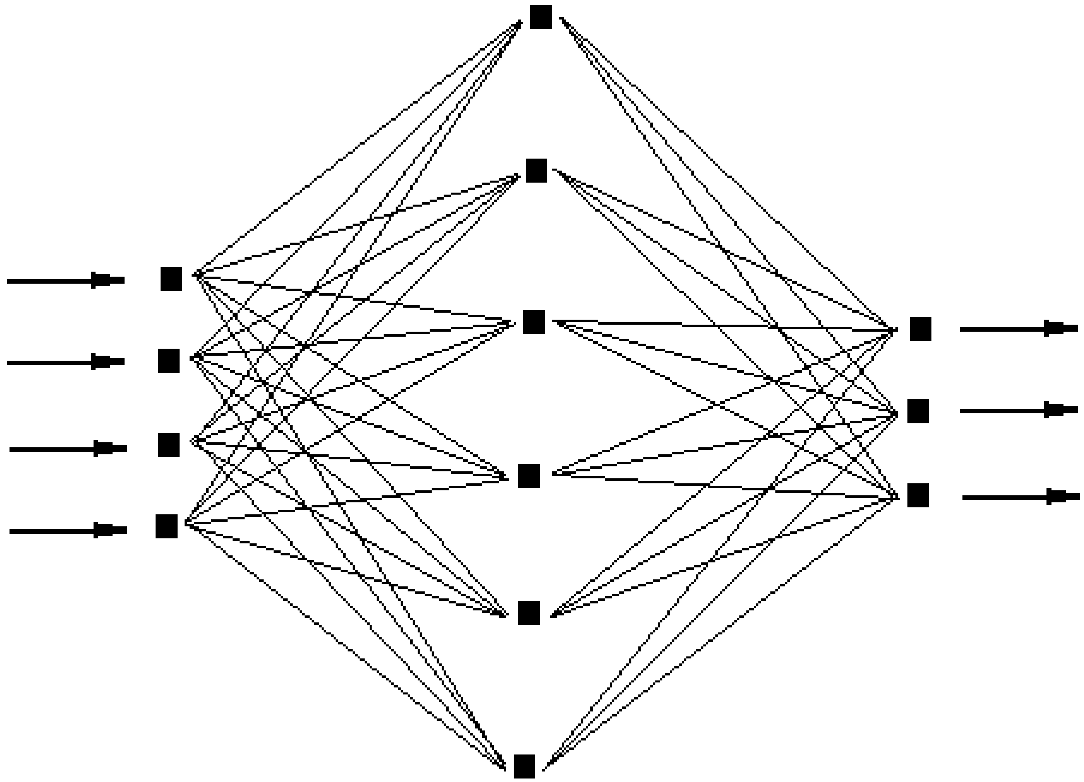
Veranstaltung	Vorlesung Mo 18-21 (X) D14/104	Praktikum Mo 18-21 (Y) D15/107	Praktikum Fr 12-15 (X) D15/107
1. Termin	28.03.11	04.04.11	08.04.11
2. Termin	11.04.11	18.04.11	29.04.11
3. Termin	02.05.11	09.05.11	13.05.11
4. Termin	16.05.11	23.05.11	27.05.11
5. Termin	30.05.11	06.06.11	17.06.11
6. Termin	20.06.11	27.06.11	01.07.11

Inhalt

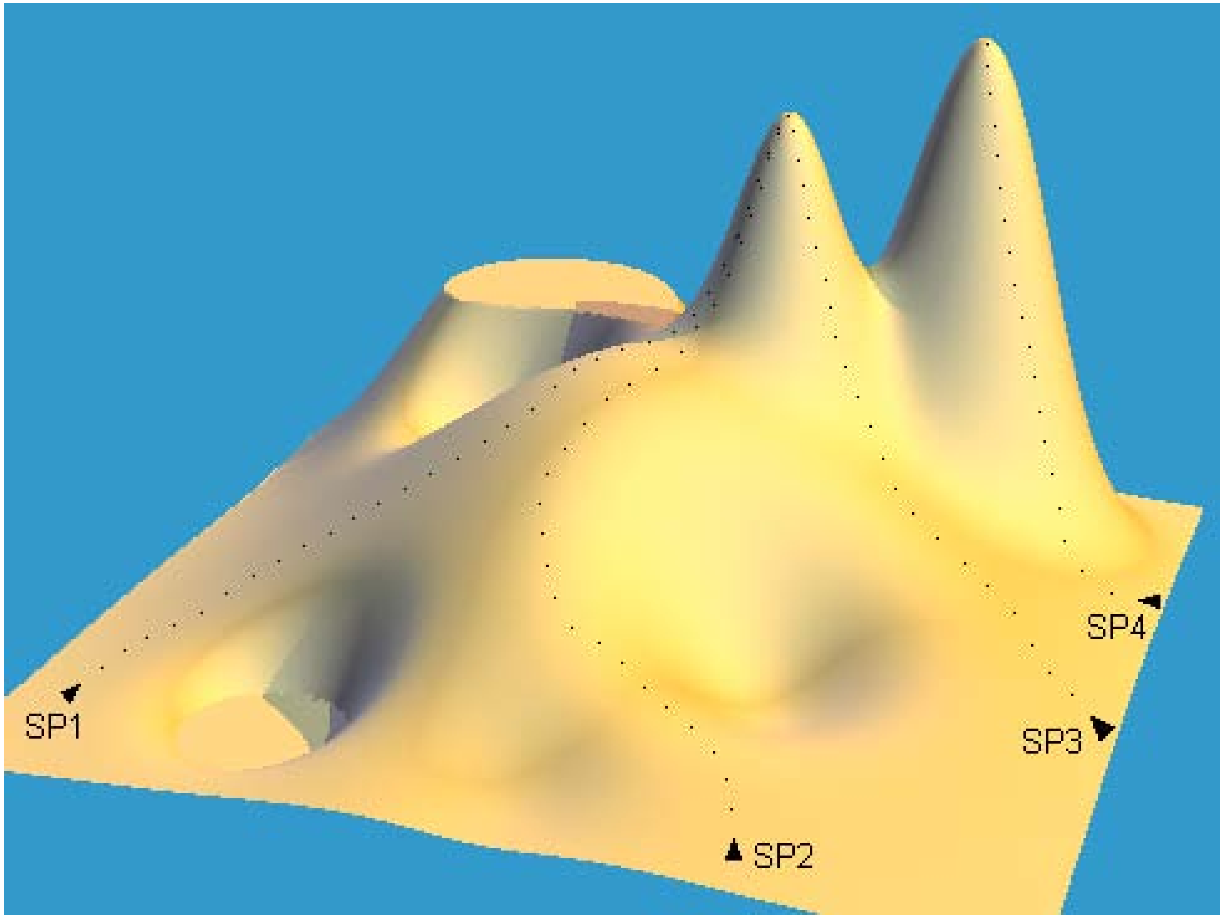
- Wiederholung
  - Mehrschichtiges Perzeptron (MLP)
  - Beschleunigung des Trainings
  - Generalisierung
  - Rekurrente Neuronale Netze
- Hidden Markov Modelle
  - Markov-Kette
  - Gaußdichten
  - Viterbi-Algorithmus


Künstliche Neuronale Netze

Mehrschichtiges Perzeptron (MLP)



Gradientenverfahren



Quelle: http://www.statistics4u.info/fundstat_germ/cc_optim_meth_gradient.html

Beschleunigung des Trainings

- Die Durchführung eines Trainingszyklus ist für große Netzwerke und große Datenmengen sehr aufwendig.
- Diverse Ansätze zur Beschleunigung des Trainings.
- Verwendung von Näherungen der zweiten Ableitung und Adaption der Lernrate.

Momentum Term I

Mit den Ziel die Konvergenz des Trainings zu steigern wird häufig ein sogenannter Momentum Term eingeführt.

$$\Delta w_{ji}^{n} = -\eta \cdot \frac{\partial E^{n}}{\partial w_{ji}} + \alpha \cdot \Delta w_{ji}^{n-1}$$

wobei $\alpha < 1$ sein muss und n den Update indiziert

Momentum Term II

konstanter Gradient für viele Updates:

$$\Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}} (1 + \alpha + \alpha^2 + \ldots)$$

$$= -\frac{\eta}{1-\alpha} \frac{\partial E}{\partial w_{ji}}$$

In monotonen Fehlerebenen kann das Training stark beschleunigt werden.

Durchführung Update
update_weight()

```
for(i=0;i < NoNeuron;i++) {
    for(j=0;j < NoNeuron;j++) {
        Update   = (-1.) * Eta * DeltaWeight[i][j]; // Schritt gemäß Gradient
        Update += Alpha * OldUpdate[i][j]; // Momentum Term
        Weight[i][j] += Update; // Gewicht wird geändert
        OldUpdate[i][j] = Update; // Speicherung des aktuellen Update
        DeltaWeight[i][j] = 0.; // Reset Gradient
    }
}
```

Durchführung des Trainings

- **Batch Update:** Update der Gewichte nach der Berechnung und Addition der Gewichtsänderungen für alle Trainingsmuster (hohe Stabilität)
- **Single Update:** Update der Gewichte nach jedem Muster (Beschleunigung)
- **Validierung:** Überprüfung des Lernerfolgs an einer Validierungsmenge (ungleich Testmenge)

Batch Update / Single Update

```
for(i=0;i < NbrCycle;i++) {
    for(j=0;j < NbrPattern;j++) {
        propagate(j);
        back_propagate(j);
        if(SingleUpdate) {
            update_weight();
            reset_delta();
        }
    }
    if(BatchUpdate) {
        update_weight();
        reset_delta();
    }
}
```

Berechnung Fehler

```
// Schleife über alle Pattern der Validierungsmenge
for (i=0, Error=0.0;i < NoPattern;i++) {
    propagate(Pattern[i]); // forward Berechnung
    for(j=StartOutput;j < NoNeuron;j++)
        Error += pow((Target[i][j]-Activity[j]),2); // !!j
}
```

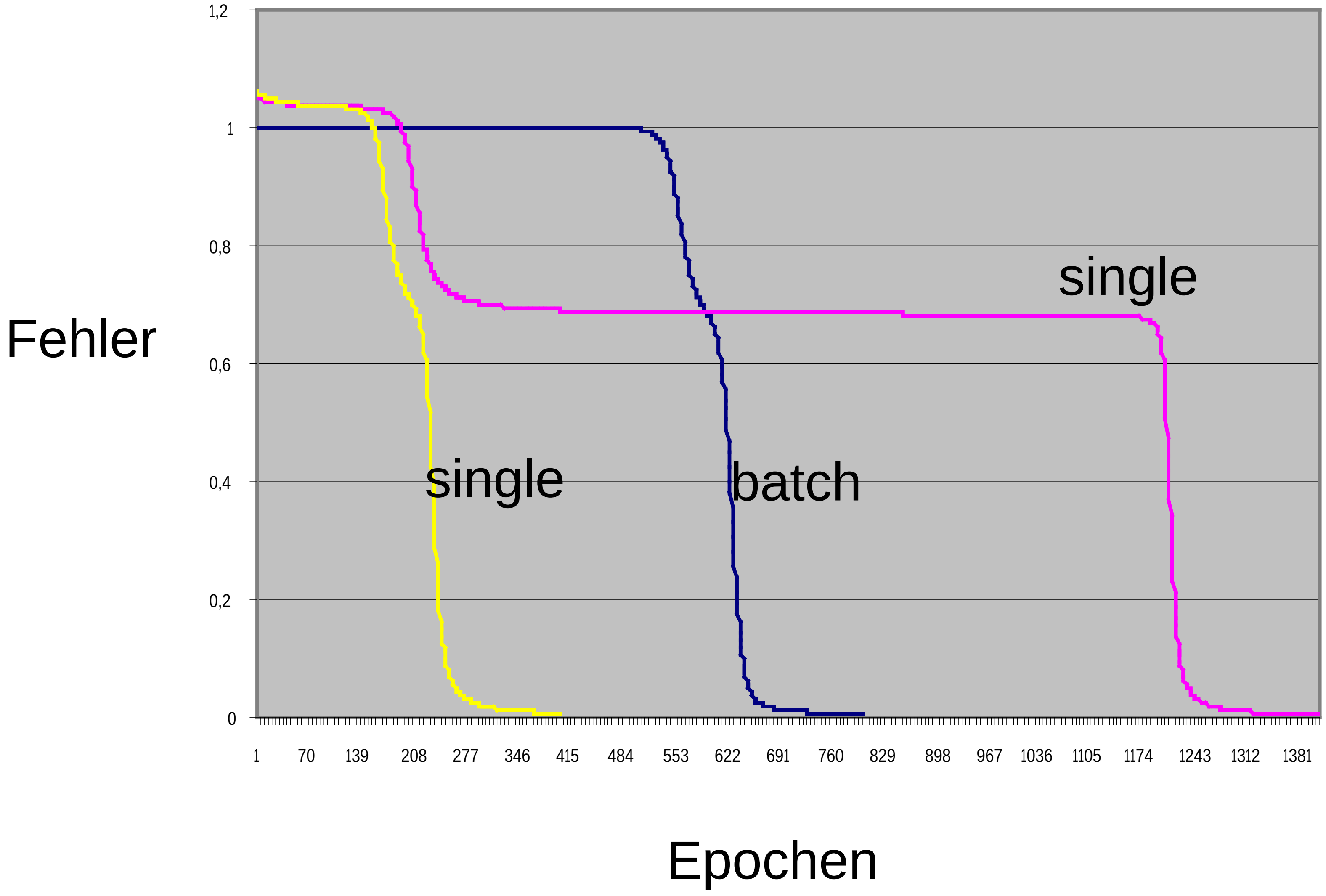
Lösung des XOR-Problems mit MLP

Hidden	2	2	2	2	4	4	2	2
Threshold	ja	ja	ja	nein	nein	nein	ja	ja
Eta	0.5	0.05	0.9	0.8	0.8	0.8	0.5	0.9
Alpha	0.9	0.9	0.0	0.9	0.9	0.9	0.9	0.0
Update	batch	batch	batch	batch	batch	single	single	single
Zyklen	500	4000	2000	1000 (10%)	500	300 (90%)	300 (95%)	1000 (95%)

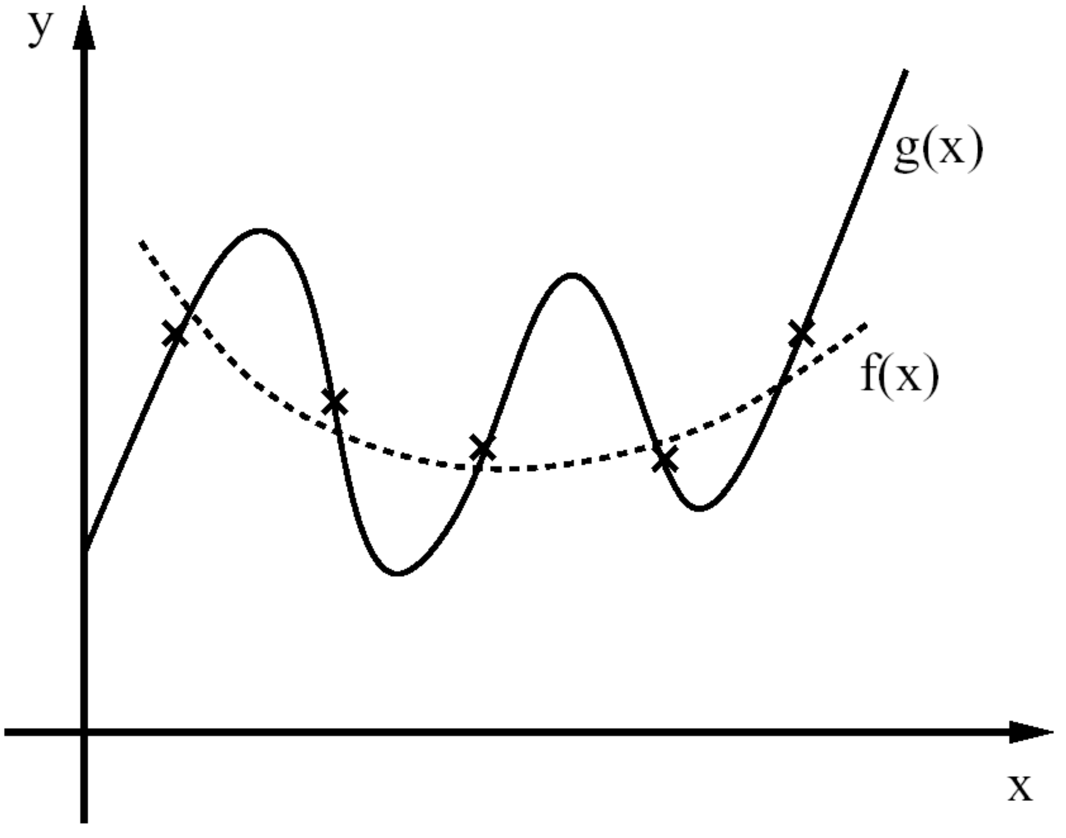
Abbruchbedingung: $E^2 < 0.1$ (Summe der Fehler für alle Muster)

Maximale Anzahl Epochen: 20.000

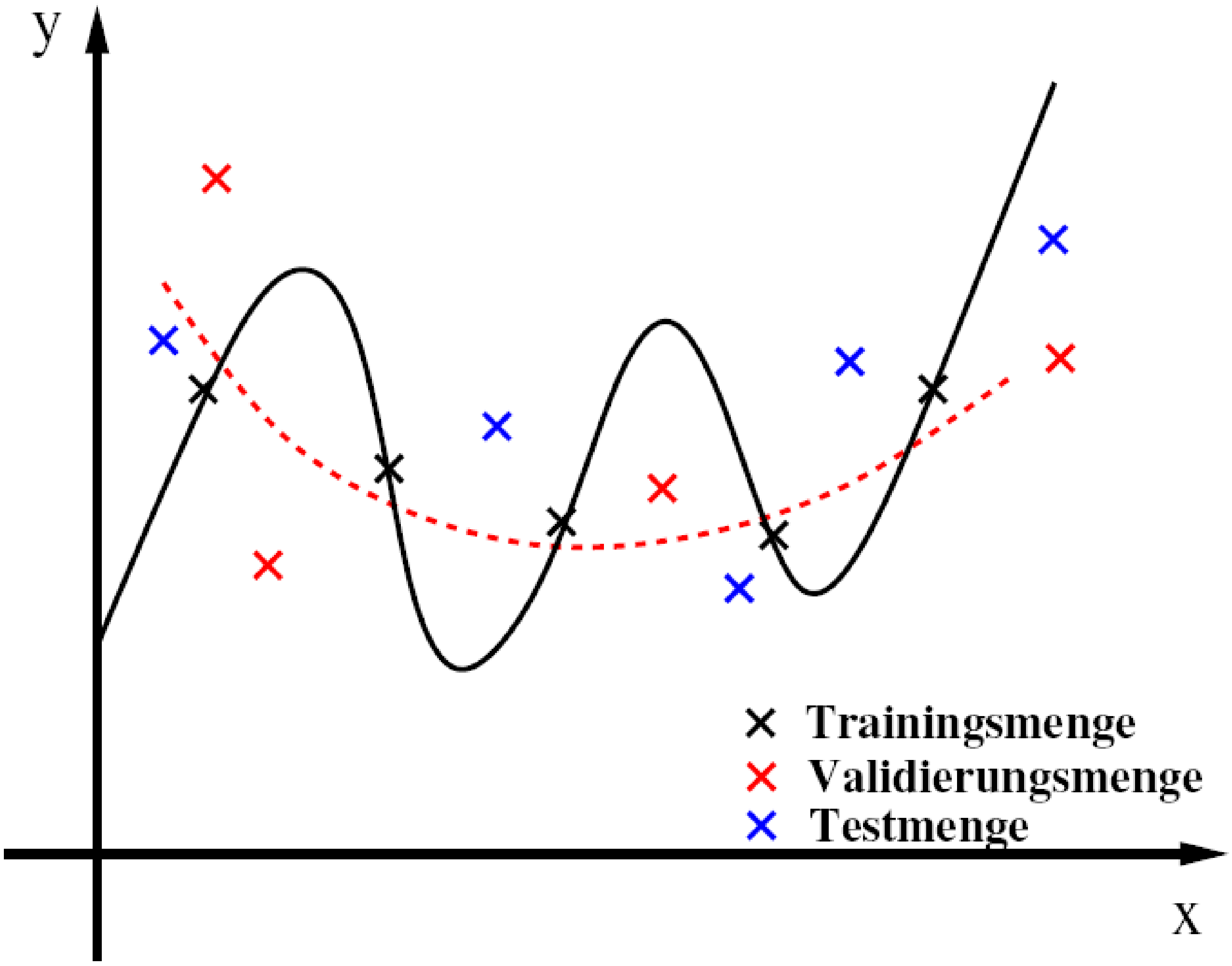
Training XOR



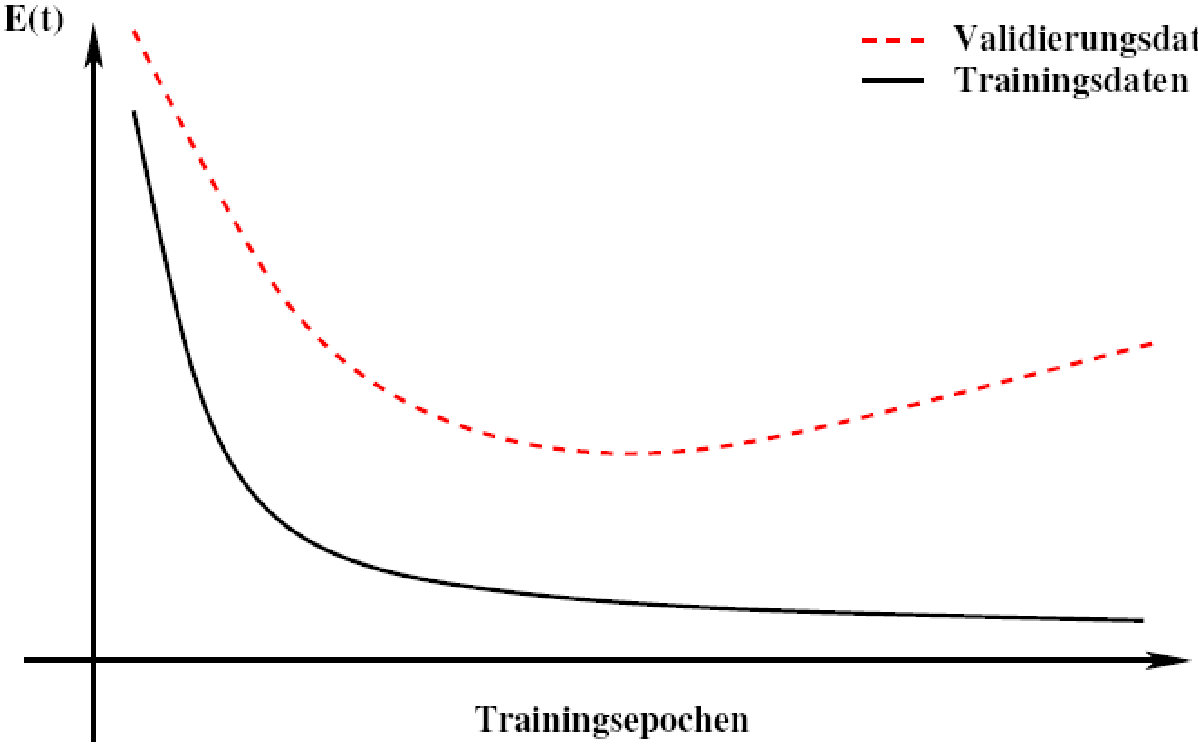
Overfitting

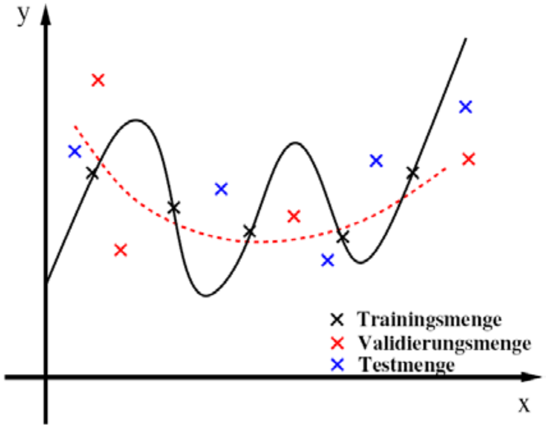


Validierung



Überwachung des Lernerfolgs





Drei Datensätze

- Trainingsdaten: Die Muster der Datenmenge werden für die Adaption der Gewichte verwendet.
- Validierungsdaten: Der berechnete Fehler für die Muster der Datenmenge wird für die Auswahl des besten Netzes verwendet.
- Testdaten: Die Muster der Datenmenge werden zur Qualitätskontrolle verwendet. Der Erfolg für die Testdaten ist ein Indikator für die Leistungsfähigkeit im realen Einsatz.

Potential MLP

- Datengetriebenes Verfahren, d.h. zur Lösung des Problems genügt die Sammlung und Parametrisierung hinreichend vieler repräsentativer Daten.
- Insbesondere ist keine Algorithmisierung der spezifischen Problemlösung erforderlich.
- Approximation beliebiger funktionaler Zusammenhänge.
- Realisierung eines optimalen Klassifikators.

Einsatz von MLP

- Musterklassifikation (z.B. XOR) (http://www.sund.de/netze/applets/BPN/bpn2/ochre.html)
  - Auf der Basis von Mustern wird die Klassenzugehörigkeit ermittelt (1 aus N-Codierung)
- Funktionsapproximation (http://neuron.eng.wayne.edu/bpFunctionApprox/bpFunctionApprox.html)
  - Funktionswerte werden auf der Basis von Funktionsargumenten ermittelt.
- Prädiktion (http://diwww.epfl.ch/mantra/tutorial/english/predic/html/index.html)
  - Zustände (z.B. Funktionswerte) werden auf der Basis vergangener Zustände prädiziert.

Einschätzung MLP

- Ist der Lernerfolg eines MLP (Bsp. XOR) von der Reihenfolge der Präsentation der Muster abhängig?
- Ist die Bewertung des *n-ten* Musters abhängig von der Bewertung des (*n-1*)-ten Musters?
- Kann ein MLP ein Gedächtnis ausbilden?
- Welche Struktur müsste ein Netz prinzipiell haben damit zeitliche Abhängigkeiten modelliert werden können?

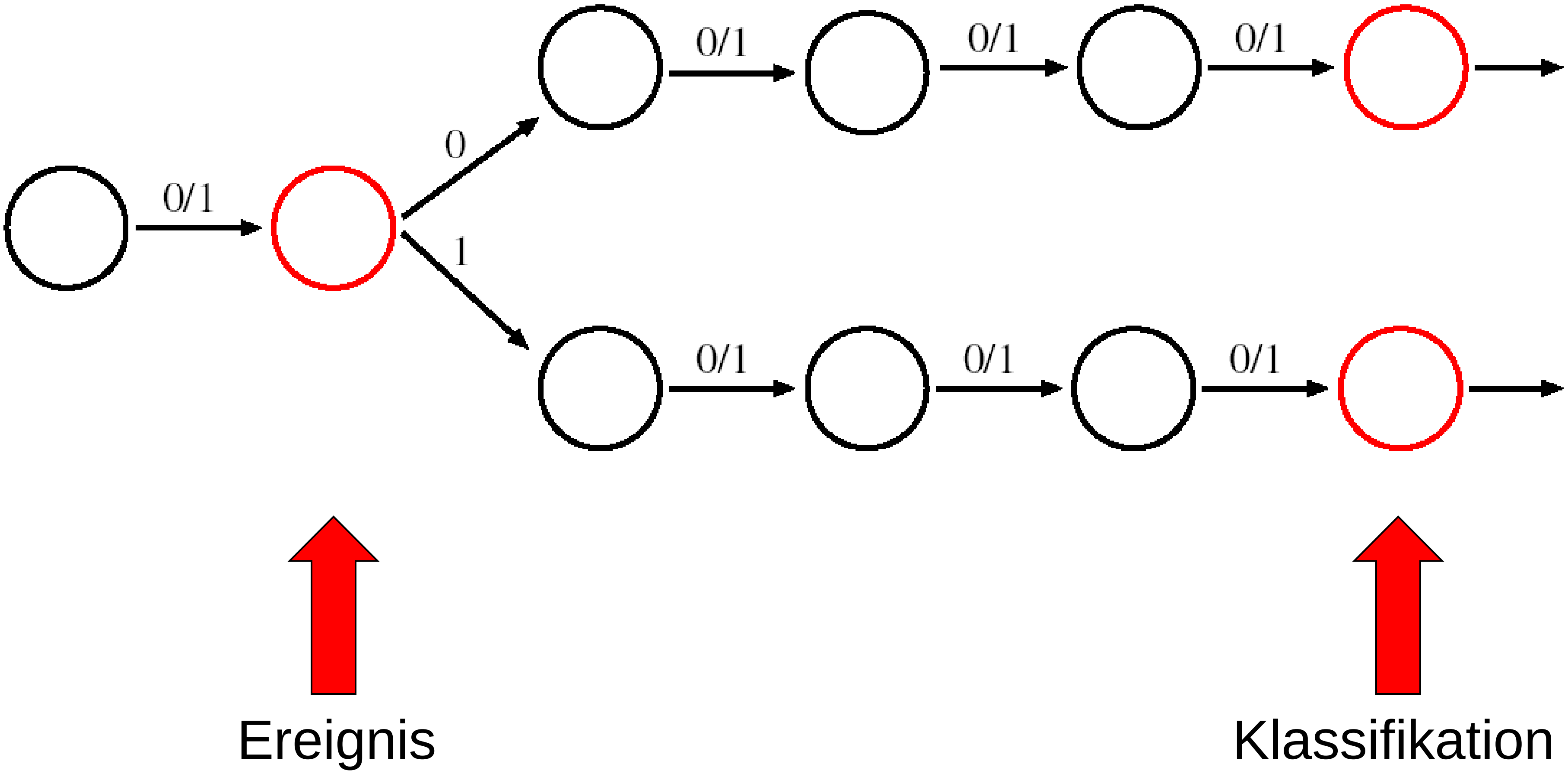
Gedächtnis

- Arbeitsgedächtnis (Kurzzeitgedächtnis)
  - begrenzte Kapazität (7 +/-2 Einheiten)
  - vergessen erfolgt durch Überschreiben
  - zur längerfristigen Speicherung Überführung ins Langzeitgedächtnis
- Langzeitgedächtnis
  - dauerhaftes Speichersystem
  - neues Einspeichern von Informationen wird als Lernen bezeichnet
  - Üben ist für Lernen grundlegend

Lernen

- Memorieren
  - auswendig lernen
  - Informationen abspeichern und bei Bedarf abrufen
  - kleine Abweichungen führen zu einer falschen Zuordnung
- Generalisieren
  - Regelhaftigkeit wird extrahiert
  - Muster werden auf Grund bestimmter Eigenschaften zugeordnet
  - gestörte Muster können mit Hilfe der gelernten strukturellen Zusammenhänge korrekt zugeordnet werden

Automat



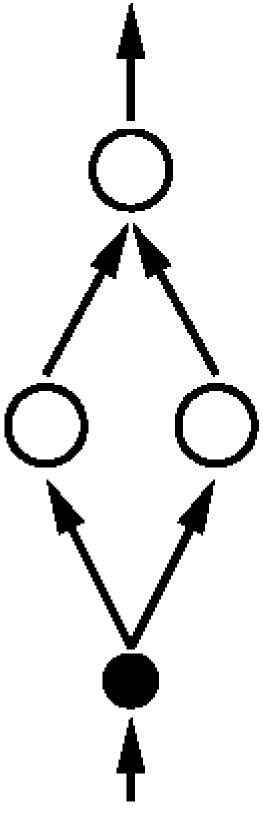
Limitationen MLP

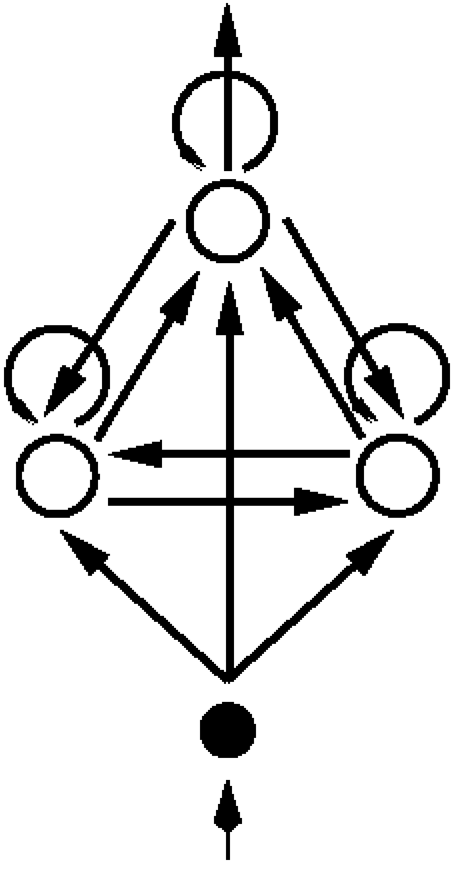
- Zeitliche Abhängigkeiten können nicht modelliert werden.
- Muster unterschiedlicher Länge können nicht verarbeitet werden.
- Die Länge des Arbeitsgedächtnisses muss explizit festgelegt werden.

Motivation für rekurrente KNN

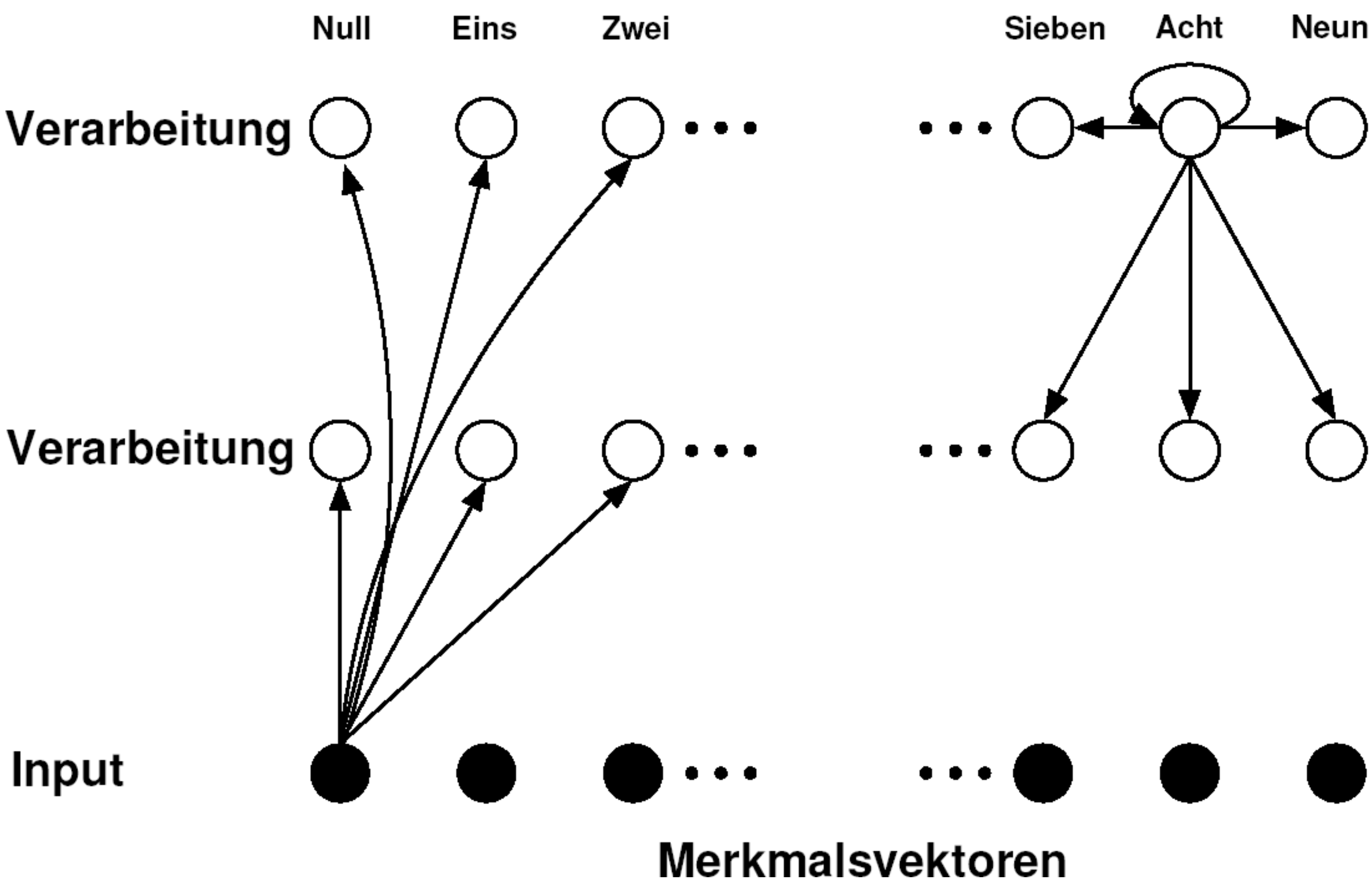
- Realisierung von Modellen, die zeitliche Abhängigkeiten berücksichtigen
- Insbesondere soll ein Arbeitsgedächtnis modelliert werden
- Muster unterschiedlicher Länge können für die Klassifikation oder Prognose verwendet werden.

KNN mit rekurrenten Verbindungen





RNN


Null
Eins
Zwei
Sieben
Acht
Neun
Verarbeitung
Verarbeitung
Input
Merkmalsvektoren

Formale Erweiterung

Aktivierung: $$a_i(t) = \sum_j w_{ji} x_j(t-1) - \theta$$

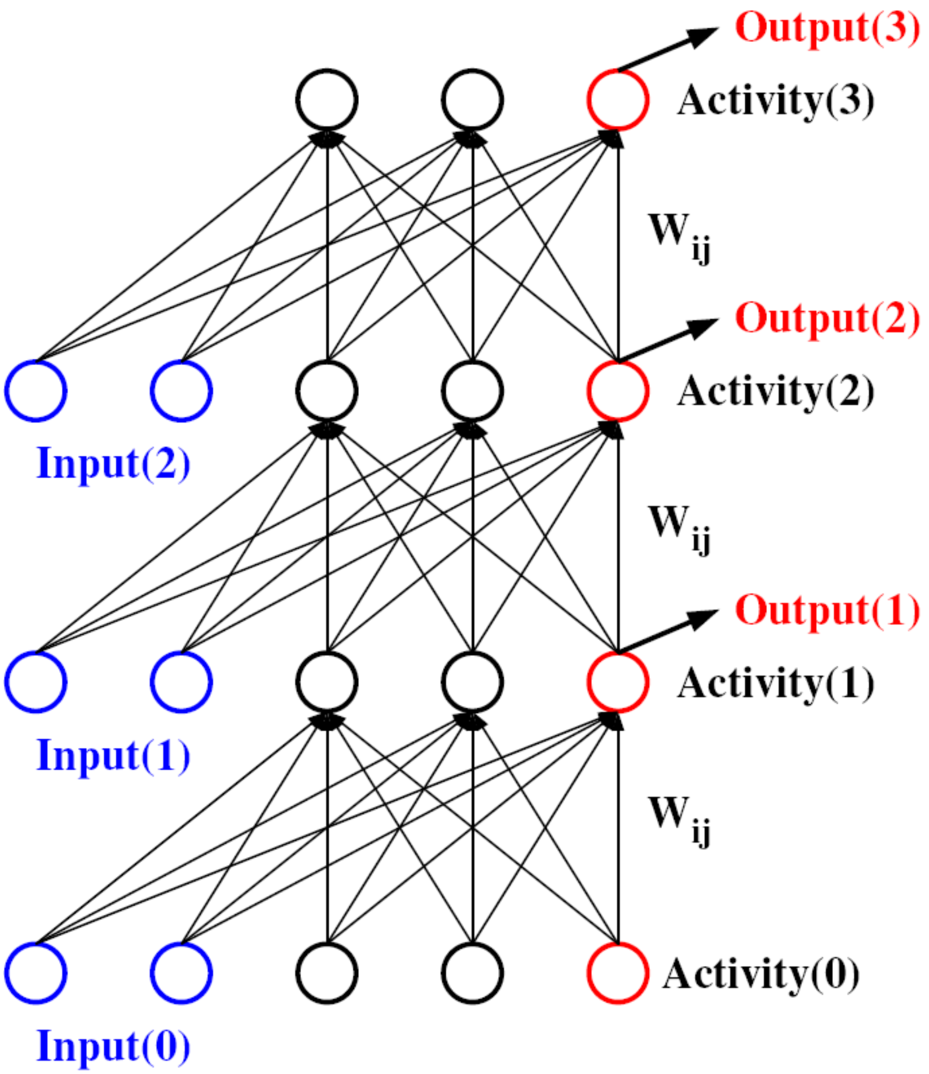
Aktivität: $$x_i(t) = f(a_i(t))$$

Transferfunktion: $$f(a_i(t))$$

Back Propagation Through Time (BPTT)

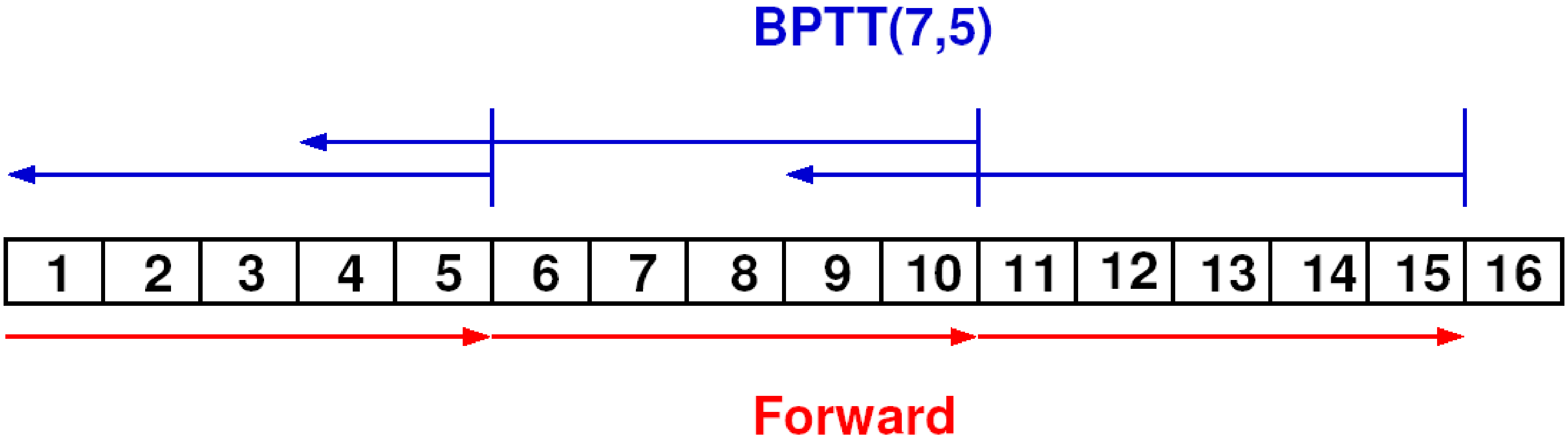
- Das Netzwerk wird in der Zeit entfaltet
- Die einzelnen Zeittakte können wie Schichten eines MLP behandelt werden
- Back Propagation Algorithmus kann angewendet werden

Entfaltung in der Zeit



BPTT(h,h´)

Nach h´ Zeittakten wird der Fehler h Zeittakte zurück propagiert (Williams & Zipser, 1990)



Formeldarstellung BPTT

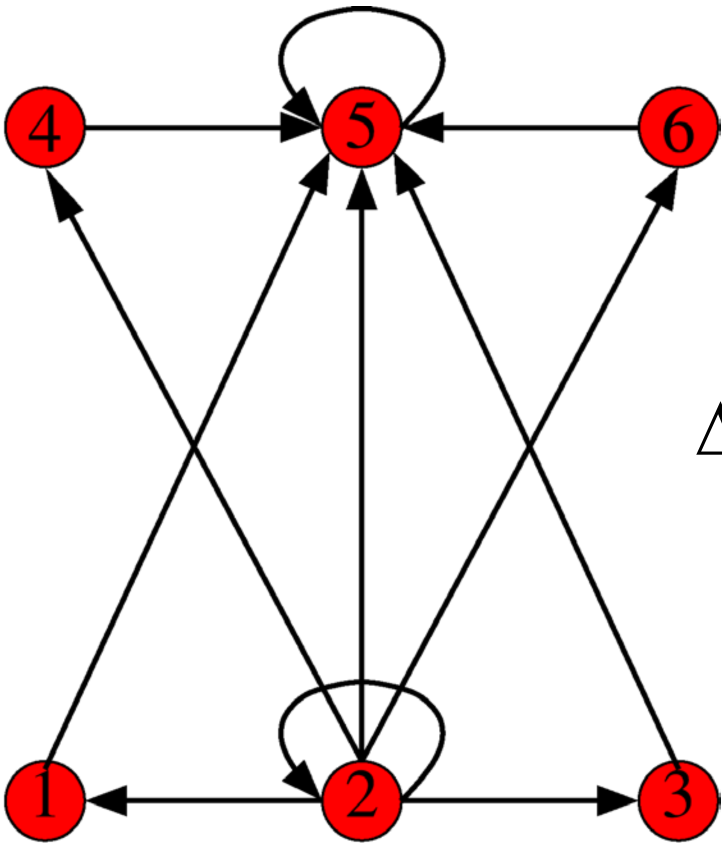
$$\delta_i^{inj}(t) = -f'(a_i(t)) \cdot (z_i(t) - x_i(t))$$

$$\delta_i^{imp}(t) = f'(a_i(t)) \cdot \sum_j w_{ij} \delta_j(t+1)$$

$$\Delta w_{ij} = -\eta \sum_t x_i(t-1) \cdot \delta_j(t)$$

Zusammenfassung BPTT

$$x_5(t) = f(w_{15}x_1(t-1) + w_{25}x_2(t-1) + w_{35}x_3(t-1)$$
$$+w_{45}x_4(t-1) + w_{55}x_5(t-1) + w_{65}x_6(t-1))$$



$$\Delta w_{25} = -\eta \sum_t x_2(t-1) \cdot \delta_5(t)$$

$$\delta_2(t) = f'(a_2(t)) \cdot (w_{21}\delta_1(t+1) + w_{22}\delta_2(t+1) + w_{23}\delta_3(t+1)$$
$$w_{24}\delta_4(t+1) + w_{25}\delta_5(t+1) + w_{26}\delta_6(t+1))$$

Gegenüberstellung

$$x_i = f\left(\sum_j w_{ji} x_j\right) \qquad x_i(t) = f\left(\sum_j w_{ji} x_j(t-1)\right)$$

$$\delta_i^{inj} = -f'(a_i)\cdot(z_i - x_i) \qquad \delta_i^{inj}(t) = -f'(a_i(t))\cdot(z_i(t) - x_i(t))$$

$$\delta_j^{imp} = f'(a_j)\sum_{i\in\mathbb{N}_j} w_{ji}\delta_i \qquad \delta_j^{imp}(t) = f'(a_j(t))\cdot\sum_i w_{ji}\delta_i(t+1)$$

$$\Delta w_{ji} = -\eta\cdot x_j\cdot\delta_i \qquad \Delta w_{ji} = -\eta\sum_t x_j(t)\cdot\delta_i(t+1)$$

Berechnung Forward Recurrent

```
for(i=StartHidden;i <= NoNeuron;i++) {
    Activation[i][t] = 0.0;
    for(j=0;j <= NoNeuron;j++) {
        Activation[i][t] += Weight[j][i] * Activity[j][t-1];
    }
    Activity[i][t] = sigmoid(Activation[i][t]);
    if(i ∈ Output) { // injizierte Fehler
        Delta[i][t] = Target[i][t] - Activity[i][t];
    }
}
```

Berechnung Backward Recurrent

```
for(t=T;t > 1;t--) {
  for(i=StartHidden;i < NoNeuron;i++) {
    Delta[i][t] *= Activity[i][t]*(1-Activity[i][t]);
    for(j=0;j < NoNeuron;j++) {
      DeltaWeight[j][i] += Activity[j][t-1] * Delta[i][t];
    }
    for(j=StartHidden;j < NoNeuron;j++) { // impliziter Fehler
      Delta[j][t-1] += Weight[j][i] * Delta[i][t];
    }
  }
}
```

Fazit Training

- Das Training von MLP und RNN mit dem Back Propagation Algorithmus kann auf die Hebbsche Regel zurück geführt werden.
- Der Back Propagation Algorithmus ist im Vergleich zu Verfahren der direkten Berechnung des Gradienten (Recurrent Learning) sehr effizient.

KNN als Modellierungstechnik

- Mit MLP stehen Werkzeuge zur effizienten Modellierung nichtlinearer Systeme zur Verfügung.
- Mit rekurrenten KNN können auch dynamische nichtlineare Systeme modelliert werden.
- Die gefundenen Lösungen sind einer Analyse nur schwer zugänglich.
- Die Integration von Vorwissen ist ein ungelöstes Problem.

Hidden Markov Modelle (HMM)

Beispiel

Rekonstruktion einer Kneipentour

Situation

- Es liegen n Rechnungen aus Gaststätten vor.
- Aus den Rechnungen gehen die Orte (Gaststätten) nicht hervor aber die Uhrzeit der Rechnungsstellung.
- Aus den Rechnungen können die Preise für Getränke und Speisen entnommen werden.
- Es liegen statistische Erhebungen zum Wechsel zwischen zentralen Orten im Rahmen von Kneipentouren vor.
- Es liegen statistische Erhebungen über die Gastronomiepreise in den Städten vor (Mittelwerte und Varianzen können berechnet werden).
- Wir nehmen an, dass die Preise normalverteilt sind.
- Welches ist auf dieser Basis die wahrscheinlichste Kneipentour?
- Problem: Kneipen können nicht aus den Rechnungen erschlossen werden (Hidden).

Modellbildung

1. Modell für die Abfolge der Kneipenbesuche (Prozess)

➡ Markov-Kette

2. Modell für die Bewertung der Speisen- und Getränkepreise (normalverteilt)

➡ Gaußdichten

Lösungsansatz

- Wir können die Rechnungen in eine zeitliche Abfolge bringen.
- Wir können für jede Rechnung an Hand der Gaußdichten berechnen mit welcher Wahrscheinlichkeit sie in der jeweiligen Stadt ausgestellt sein könnte.
- Wir kennen die Markov-Kette für die Beschreibung der Abfolge.
- Es fehlt noch die Verbindung zwischen der Abfolge und der Verteilung der Preise.

Modellbildung

1. Modell für die Abfolge der Kneipenbesuche (Prozess)

   Markov-Kette

2. Modell für die Bewertung der Speisen- und Getränkepreise (normalverteilt)

   Gaußdichten

Markov-Kette

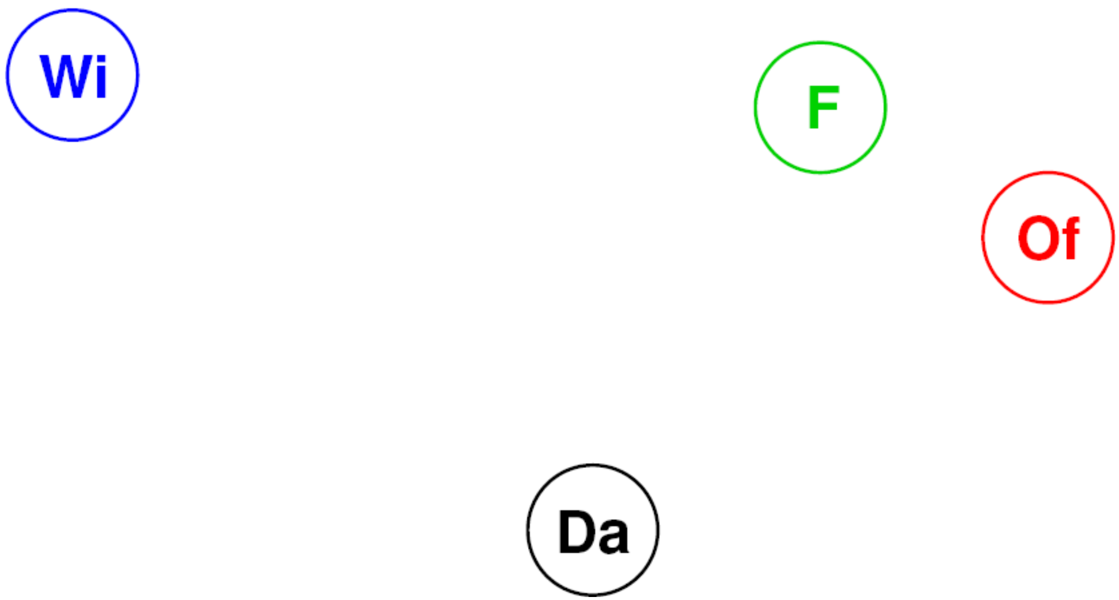
- Andrei Andreyevich Markov (1856-1922)

- Das stochastische Modell der zeit-diskreten Markov-Ketten mit endlich vielen Zuständen besteht aus den drei Komponenten: Zustandsraum, Anfangs-verteilung und Übergangsmatrix

Kneipentour Rhein-Main (1)

S_1: Wiesbaden (Wi)
S_2: Frankfurt (F)
S_3: Offenbach (Of)
S_4: Darmstadt (Da)

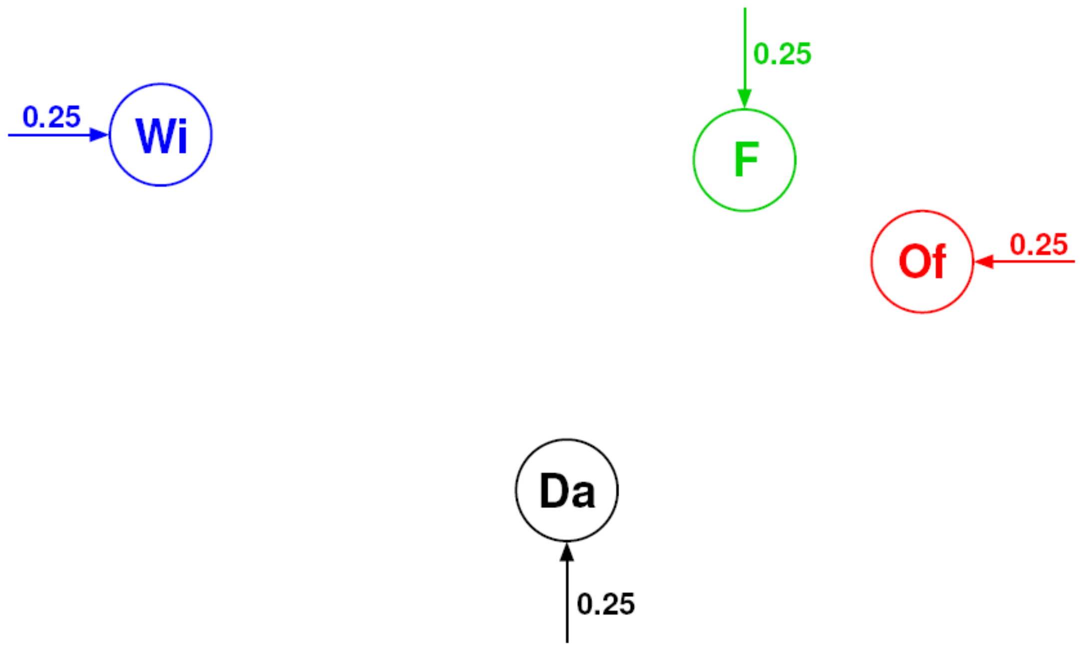
$$P_{init} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix} \qquad \underline{\underline{P}} = \begin{pmatrix} 0.7 & 0.2 & 0.05 & 0.05 \\ 0.0 & 0.85 & 0.15 & 0.0 \\ 0.1 & 0.1 & 0.6 & 0.2 \\ 0.1 & 0.3 & 0.1 & 0.5 \end{pmatrix}$$



Kneipentour Rhein-Main (2)

S_1: Wiesbaden (Wi)
S_2: Frankfurt (F)
S_3: Offenbach (Of)
S_4: Darmstadt (Da)

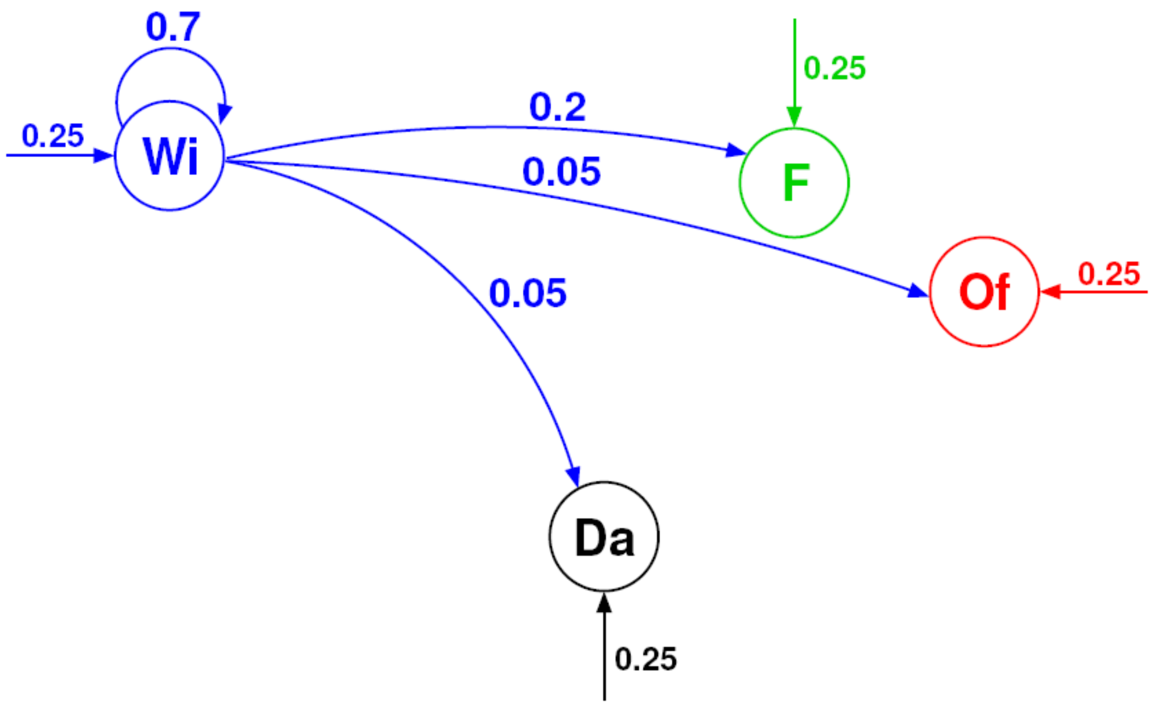
$$P_{init} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix} \qquad \underline{\underline{P}} = \begin{pmatrix} 0.7 & 0.2 & 0.05 & 0.05 \\ 0.0 & 0.85 & 0.15 & 0.0 \\ 0.1 & 0.1 & 0.6 & 0.2 \\ 0.1 & 0.3 & 0.1 & 0.5 \end{pmatrix}$$



Kneipentour Rhein-Main (3)

S_1: Wiesbaden (Wi)
S_2: Frankfurt (F)
S_3: Offenbach (Of)
S_4: Darmstadt (Da)

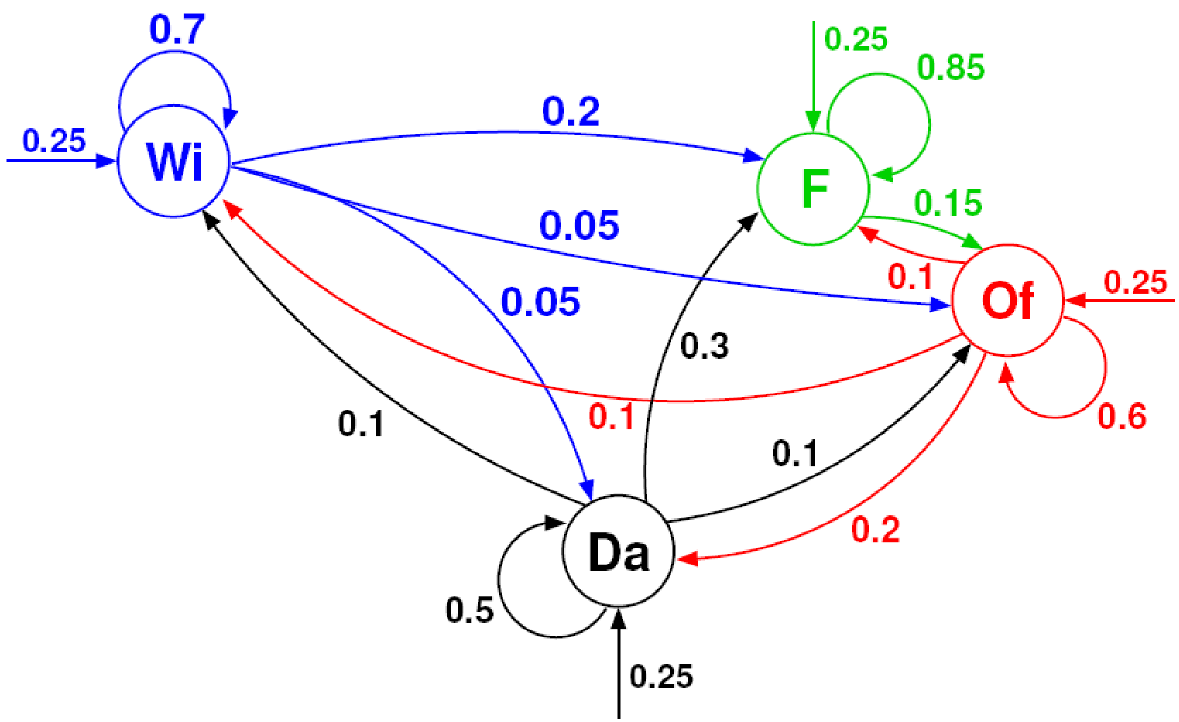
$$P_{init} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix} \qquad \underline{\underline{P}} = \begin{pmatrix} 0.7 & 0.2 & 0.05 & 0.05 \\ 0.0 & 0.85 & 0.15 & 0.0 \\ 0.1 & 0.1 & 0.6 & 0.2 \\ 0.1 & 0.3 & 0.1 & 0.5 \end{pmatrix}$$



Kneipentour Rhein-Main (4)

S_1: Wiesbaden (Wi)
S_2: Frankfurt (F)
S_3: Offenbach (Of)
S_4: Darmstadt (Da)

$$P_{init} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix} \qquad \underline{\underline{P}} = \begin{pmatrix} 0.7 & 0.2 & 0.05 & 0.05 \\ 0.0 & 0.85 & 0.15 & 0.0 \\ 0.1 & 0.1 & 0.6 & 0.2 \\ 0.1 & 0.3 & 0.1 & 0.5 \end{pmatrix}$$



Welche Fragen können beantwortet werden?

Wie wahrscheinlich ist die Kneipentour:
$S_4 – S_4 – S_3 – S_3 – S_1 – S_2$

$$P_{init} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix} \qquad \underline{\underline{P}} = \begin{pmatrix} 0.7 & 0.2 & 0.05 & 0.05 \\ 0.0 & 0.85 & 0.15 & 0.0 \\ 0.1 & 0.1 & 0.6 & 0.2 \\ 0.1 & 0.3 & 0.1 & 0.5 \end{pmatrix}$$

$$P(S_4, S_4, S_3, S_3, S_1, S_2) = 0{,}25 * 0{,}5 * 0{,}1 * 0{,}6 * 0{,}1 * 0{,}2 = 0{,}00015$$

Dieses Ereignis tritt einmal je 6.667 Kneipentoren auf.

Übung: Münzwurf (1)

Für ein Experiment stehen zwei unterschiedliche Münzen zur Verfügung:
1. Manipulierte Münze (MM)
2. Unveränderte Münze (UM)

Emissionswahrscheinlichkeiten

p (Kopf | UM):
p (Zahl | UM):
p (Kopf | MM): 0.8
p (Zahl | MM):

p_{init} (UM): 0.6
p_{init} (MM):

Übergangswahrscheinlichkeiten

	UM	MM
UM	0.95	
MM		0.04

Ergänzen Sie bitte die fehlenden Wahrscheinlichkeiten.

Bitte ermitteln Sie die wahrscheinlichste Sequenz für zwei Münzwürfe unter der Annahme, dass bei beiden Münzwürfen das Ergebnis „Kopf“ ist.

Übung: Münzwurf (2)

Für ein Experiment stehen zwei unterschiedliche Münzen zur Verfügung:
1. Manipulierte Münze (MM)
2. Unveränderte Münze (UM)

Emissionswahrscheinlichkeiten

p (Kopf | UM): 0.5
p (Zahl | UM): 0.5
p (Kopf | MM): 0.8
p (Zahl | MM): 0.2

p_{init} (UM): 0.6
p_{init} (MM): 0.4

Übergangswahrscheinlichkeiten

	UM	MM
UM	0.95	0.05
MM	0.96	0.04

Ergänzen Sie bitte die fehlenden Wahrscheinlichkeiten.

Bitte ermitteln Sie die wahrscheinlichste Sequenz für zwei Münzwürfe unter der Annahme, dass bei beiden Münzwürfen das Ergebnis „Kopf“ ist.

Übung: Münzwurf (3)

Bitte ermitteln Sie die wahrscheinlichste Sequenz für zwei Münzwürfe unter der Annahme, dass bei beiden Münzwürfen das Ergebnis „Kopf“ ist.

Emissionswahrscheinlichkeiten

p (Kopf | UM):	0.5
p (Zahl | UM):	0.5
p (Kopf | MM):	0.8
p (Zahl | MM):	0.2
p_{init} (UM):	0.6
p_{init} (MM):	0.4

Übergangswahrscheinlichkeiten

	UM	MM
UM	0.95	0.05
MM	0.96	0.04

Sequenz	p_{init}	p(K)	p(Ü)	p(K)	p (Sequenz)
UM, UM:					
UM, MM:					
MM, UM:					
MM, MM:					

Übung: Münzwurf (4)

Bitte ermitteln Sie die wahrscheinlichste Sequenz für zwei Münzwürfe unter der Annahme, dass bei beiden Münzwürfen das Ergebnis „Kopf“ ist.

Emissionswahrscheinlichkeiten

p (Kopf | UM):	0.5
p (Zahl | UM):	0.5
p (Kopf | MM):	0.8
p (Zahl | MM):	0.2
p_{init} (UM):	0.6
p_{init} (MM):	0.4

Übergangswahrscheinlichkeiten

	UM	MM
UM	0.95	0.05
MM	0.96	0.04

Sequenz	p_{init}	p(K)	p(Ü)	p(K)	p (Sequenz)
UM, UM:	0.6	0.5	0.95	0.5	14.25%
UM, MM:	0.6	0.5	0.05	0.8	1.20%
MM, UM:	0.4	0.8	0.96	0.5	15.36%
MM, MM:	0.4	0.8	0.04	0.8	1.02%

Übung: Münzwurf (5)

Bitte ermitteln Sie die wahrscheinlichste Sequenz für zwei Münzwürfe unter der Annahme, dass bei beiden Münzwürfen das Ergebnis „Kopf“ ist.

Sequenz	p_{init}	p(K)	p(Ü)	p(K)	p (Sequenz)
UM, UM:	0.6	0.5	0.95	0.5	14.25%
UM, MM:	0.6	0.5	0.05	0.8	1.20%
MM, UM:	0.4	0.8	0.96	0.5	15.36%
MM, MM:	0.4	0.8	0.04	0.8	1.02%

Warum addieren sich die Wahrscheinlichkeiten für die untersuchten Sequenzen nicht zu 1?

Modellbildung

1. Modell für die Abfolge der Kneipenbesuche (Prozess)

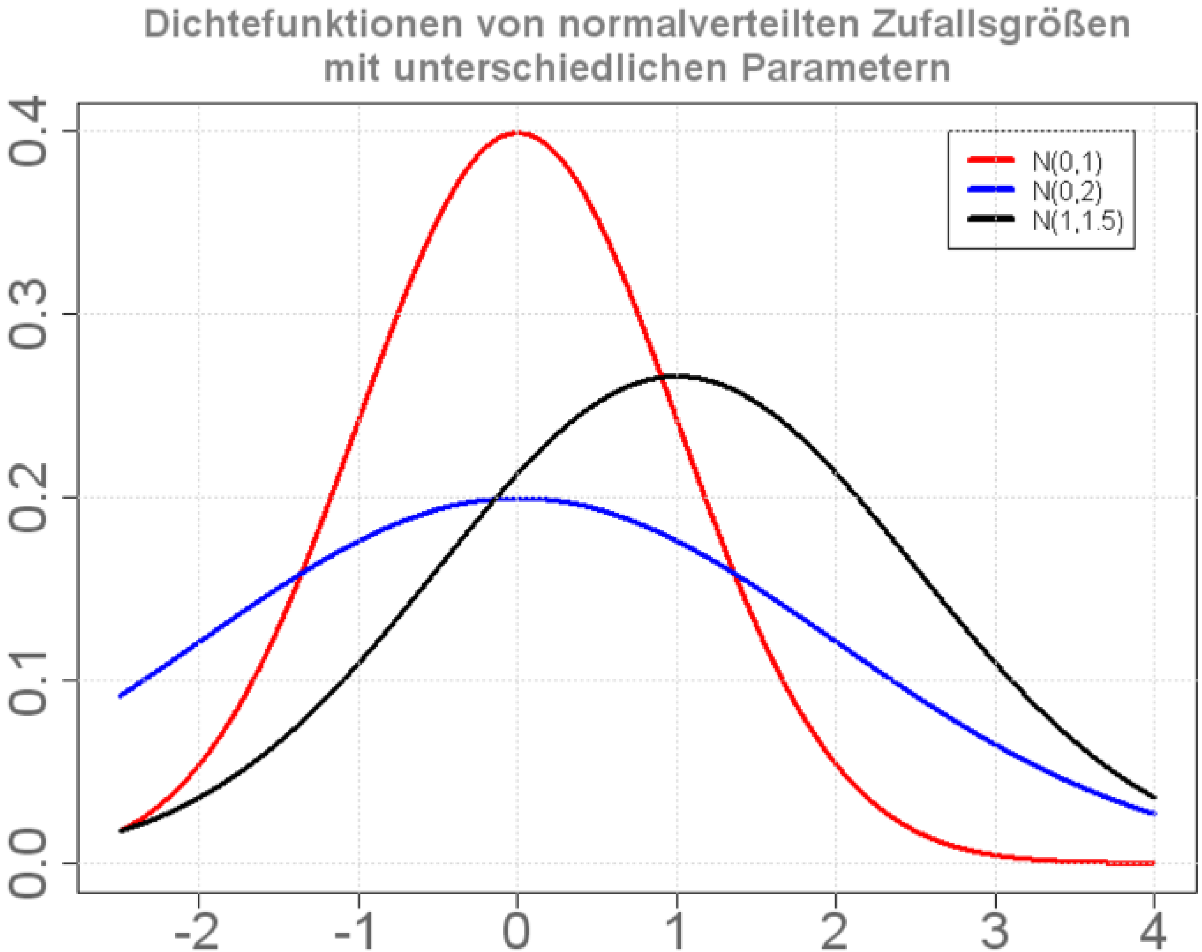
   Markov-Kette

2. Modell für die Bewertung der Speisen- und Getränkepreise (normalverteilt)

   Gaußdichten

Berechnung der Scores überlagerter Gaußdichten

Beispiele


Dichtefunktionen von normalverteilten Zufallsgrößen
mit unterschiedlichen Parametern
N(0,1)
N(0,2)
N(1,1.5)
0.0
0.1
0.2
0.3
0.4
-2
-1
0
1
2
3
4

Verbundwahrscheinlichkeit

falls a und b unabhängig sind:

$$p(a \mid b) = p(a) \cdot p(b)$$

$$\log(p(a \mid b)) = \log(p(a) \cdot p(b))$$

$$= \log(p(a)) + \log(p(b))$$

Überlagerung von Gaußdichten

$$v_i(\underline{m}(t)) = \frac{1}{\sqrt{(2\pi)^n \cdot \det(\underline{\underline{\Sigma}}_i)}} \cdot e^{-\frac{1}{2}(\underline{m}(t)-\underline{\mu}_i)^T (\underline{\underline{\Sigma}}_i)^{-1}(\underline{m}(t)-\underline{\mu}_i)}$$

$$b_i(\underline{m}(t)) = \sum_{j=1}^{m} g_i^j v_i^j(\underline{m}(t))$$

$$V_i^j(\underline{m}(t)) = g_i^j \cdot v_i^j(\underline{m}(t))$$

Glossar

$\underline{m}(t)$: Merkmalsvektor (bspw. Bier, Pizza, etc.)
$v_i(\underline{m}(t))$: Gaußdichte i (z.B. Frankfurt) für die Merkmalsvektoren
$b_i(\underline{m}(t))$: Bewertung i (z.B. Frankfurt) für die Merkmalsvektoren, die sich aus einer Überlagerung mehrerer Gaußdichten v (z.B. gehobene Gastronomie, Studentenkneipe) ergibt
$v_i^J(\underline{m}(t))$: Gaußdichte j (z.B. gehobene Gastronomie) für die Stadt i
g_i^J: Gewichtsfaktor der Gaußdichte j der Stadt i
$V_i^J(\underline{m}(t))$: Gaußdichte j (z.B. gehobene Gastronomie) für die Stadt i (incl. Gewichtsfaktor)
G_i^J: Konstante der Gaußdichte j für Stadt i, die neben dem Gewichtsfaktor auch die Normierungskonstante enthält
$\varphi_{l,i}^J$: Konstante der l-ten Komponente der Gaußdichte j für Stadt i

Auswertung Gaußdichten I

$$\ln(V_i^j(\underline{m}(t))) = \ln\left(\frac{g_i^j}{\sqrt{(2\pi)^n \cdot \det(\underline{\underline{\Sigma}}_i)}}\right) - \frac{1}{2}(\underline{m}(t) - \underline{\mu}_i^j)^T (\underline{\underline{\Sigma}}_i^j)^{-1}(\underline{m}(t) - \underline{\mu}_i^j)$$

$$G_i^j = \ln\left(\frac{g_i^j}{\sqrt{(2\pi)^n \cdot \det(\underline{\underline{\Sigma}}_i)}}\right)$$

unter der Bedingung, dass die Kovarianzmatrix eine Diagonalgestalt hat, gilt:

$$\ln(V_i^j(\underline{m}(t))) = G_i^j - \frac{1}{2}\sum_{l=1}^{n}\frac{(m_l(t) - \mu_{l,i}^j)^2}{\sigma_{l,i}^j}$$

Auswertung Gaußdichten II

$$\ln(V_i^j(\underline{m}(t))) = G_i^j - \frac{1}{2}\sum_{l=1}^{n}\frac{(m_l(t)-\mu_{l,i}^j)^2}{\sigma_{l,i}^j}$$

$$\varphi_{l,i}^j = \frac{1}{2\sigma_{l,i}^j}$$

$$\ln(V_i^j(\underline{m}(t))) = G_i^j - \sum_{l=1}^{n}\varphi_{l,i}^j(m_l(t)-\mu_{l,i}^j)^2$$

Implementierung Gaußdichte

$$\ln(V_i^j(\underline{m}(t))) = G_i^j - \sum_{l=1}^{n} \varphi_{l,i}^j (m_l(t) - \mu_{l,i}^j)^2$$

```
Gauss[i][j].Score = Gauss[i][j].Gfact;
for(k=0;k < n;k++) {
   Gauss[i][j].Score -= (Gauss[i][j].Pfact[k]
                          *(Feature[k]-Gauss[i][j].Mean[k])
                          *(Feature[k]-Gauss[i][j].Mean[k]);
}
```

Logarithmische Addition I

$$\ln(V_i^j(\underline{m}(t))) = G_i^j - \sum_{l=1}^{n} \varphi_{l,i}^j (m_l(t) - \mu_{l,i}^j)^2$$

$$\ln(b_i(\underline{m}(t))) = \ln(\sum_{j=1}^{m} V_i^j(\underline{m}(t)))$$

$$\ln(y) = \ln(\sum_{j=1}^{m} x_i)$$

Logarithmische Addition II

$$\ln(y) = \ln(\sum_{j=1}^{2} x_i) = \ln(x_1 + x_2) = \ln(x_1 \cdot (1 + \frac{x_2}{x_1})) = \ln x_1 + \ln(1 + \frac{x_2}{x_1})$$

$$\phi_1 = \ln(x_1) \qquad \phi_2 = \ln(x_2)$$

$$e^{\phi_1} = e^{\ln(x_1)} = x_1$$

$$\ln(y) = \phi_1 + \ln(1 + \frac{e^{\phi_2}}{e^{\phi_1}}) = \phi_1 + \ln(1 + e^{\phi_2 - \phi_1})$$

wobei: $\phi_1 \geq \phi_2$

Implementierung logarithmische Addition

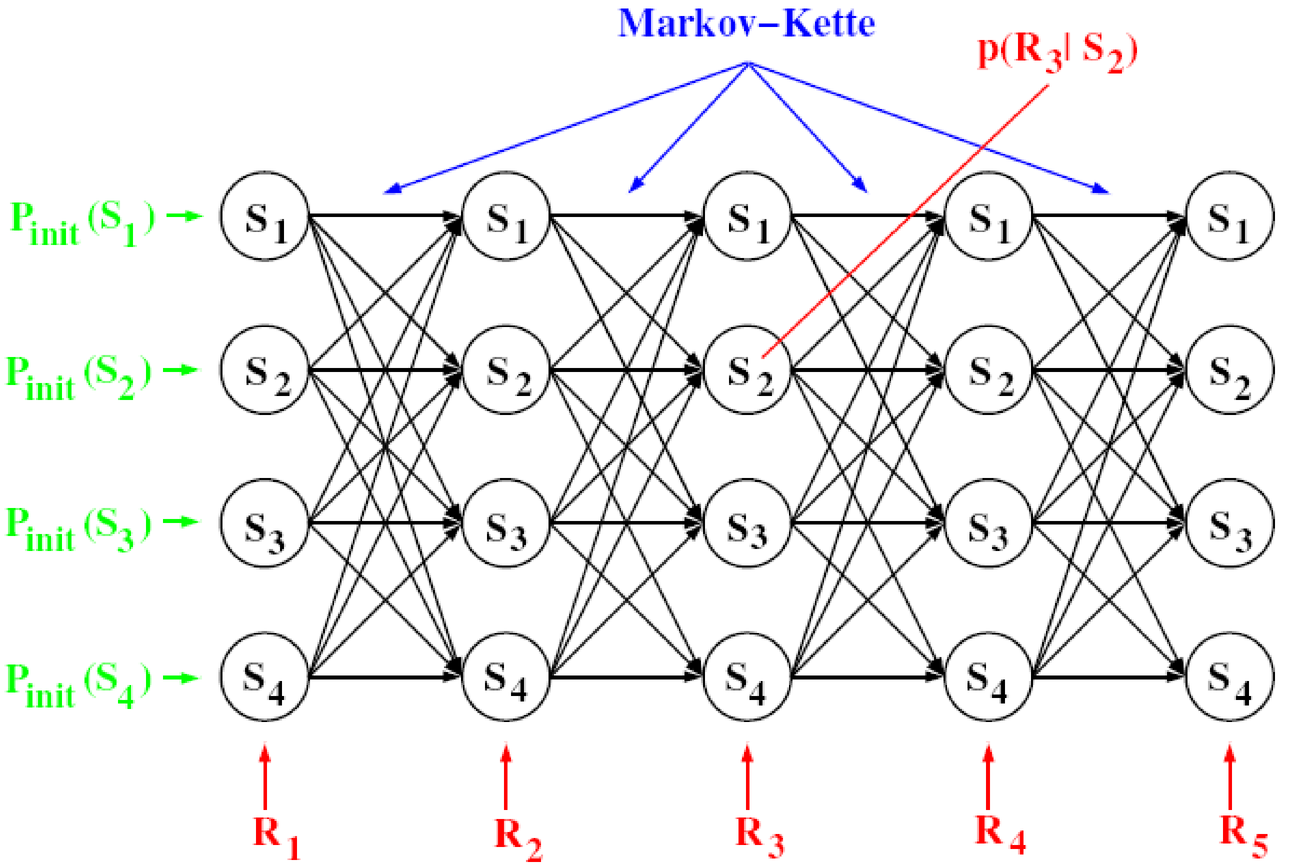
$$X_1 + \ln(1 + e^{X_2 - X_1}) \qquad \text{wobei: } X_1 \geq X_2$$

```
Score[i]=Gauss[i][0]
for(j=1;j < m;j++) {
  if(Gauss[i][j] > Score[i]) {
    Score[i] += Gauss[i][j] + ln(1+exp(Score[i]−Gauss[i][j])
  }
  else {
    Score[i] += Score[i] + ln(1+exp(Gauss[i][j]−Score[i])
  }
}
```

Zwischenstand

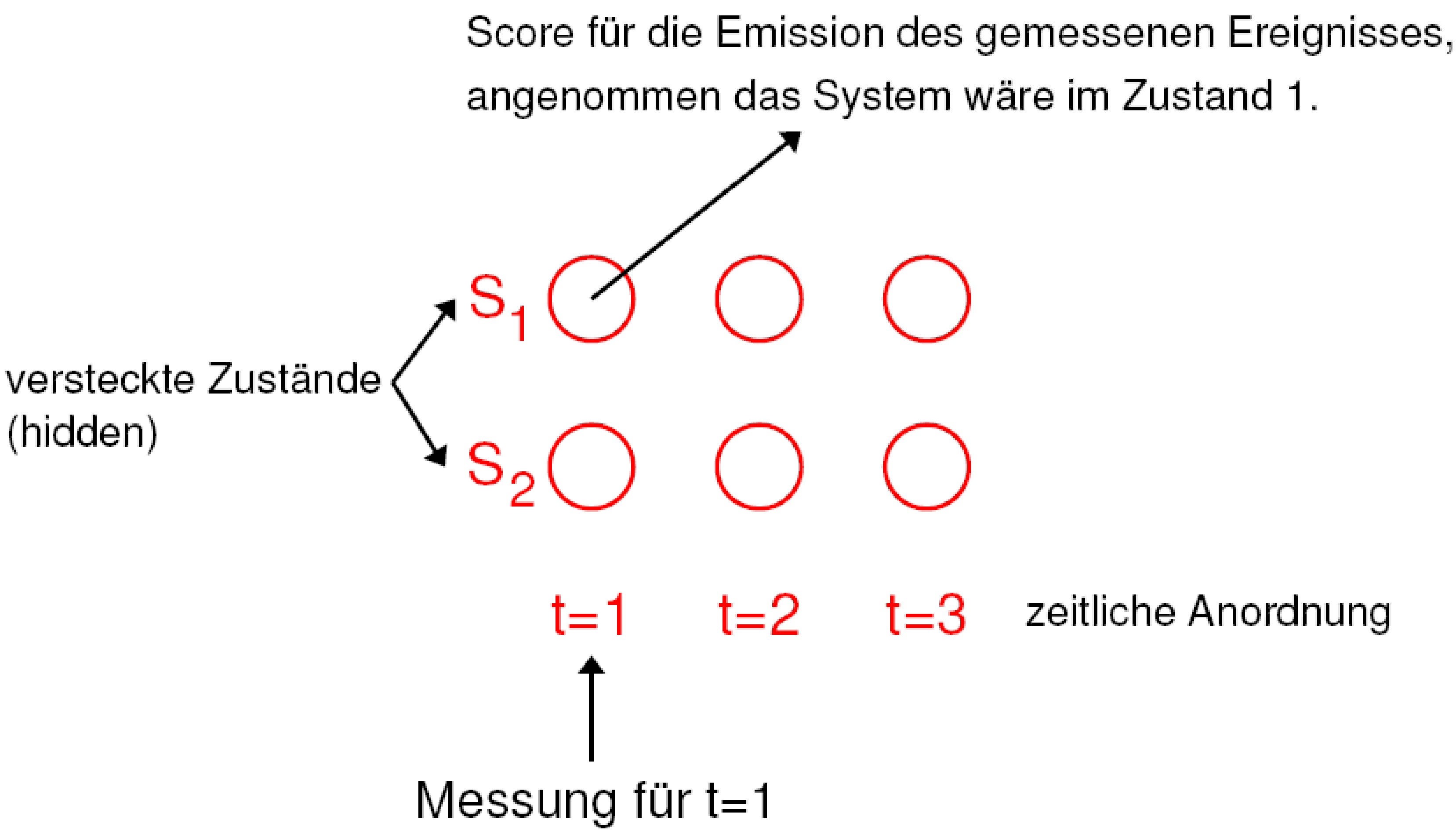
- Wir können die Rechnungen in eine zeitliche Abfolge bringen.
- Wir kennen die Markov-Kette für die Beschreibung der Abfolge.
- Wir können für jede Rechnung an Hand der Gaußdichten einen Score berechnen, dass sie in der jeweiligen Stadt ausgestellt sein könnte.
- Es fehlt noch die Verbindung zwischen der Abfolge und der Verteilung der Preise.

Lösungsansatz

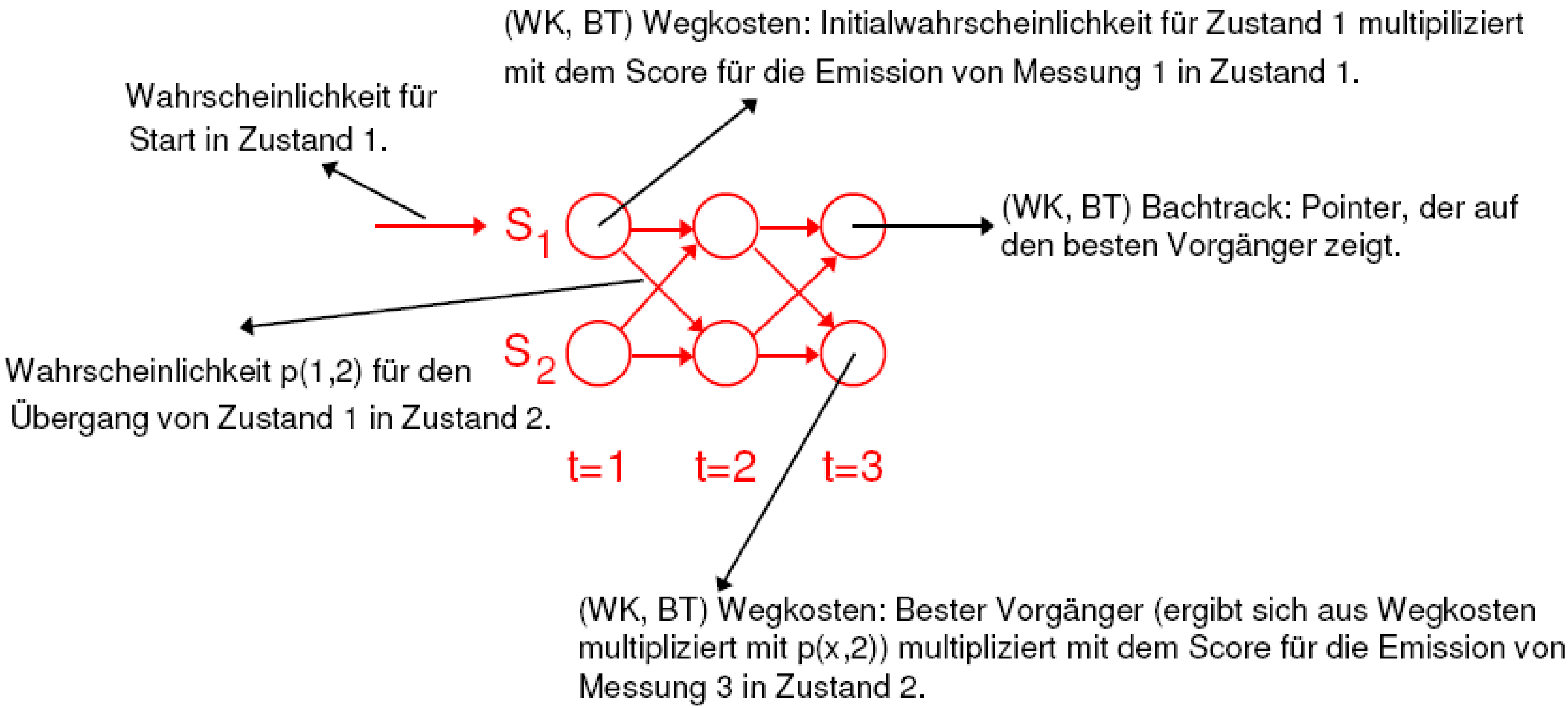


Wie kann der optimale Weg effizient berechnet werden?

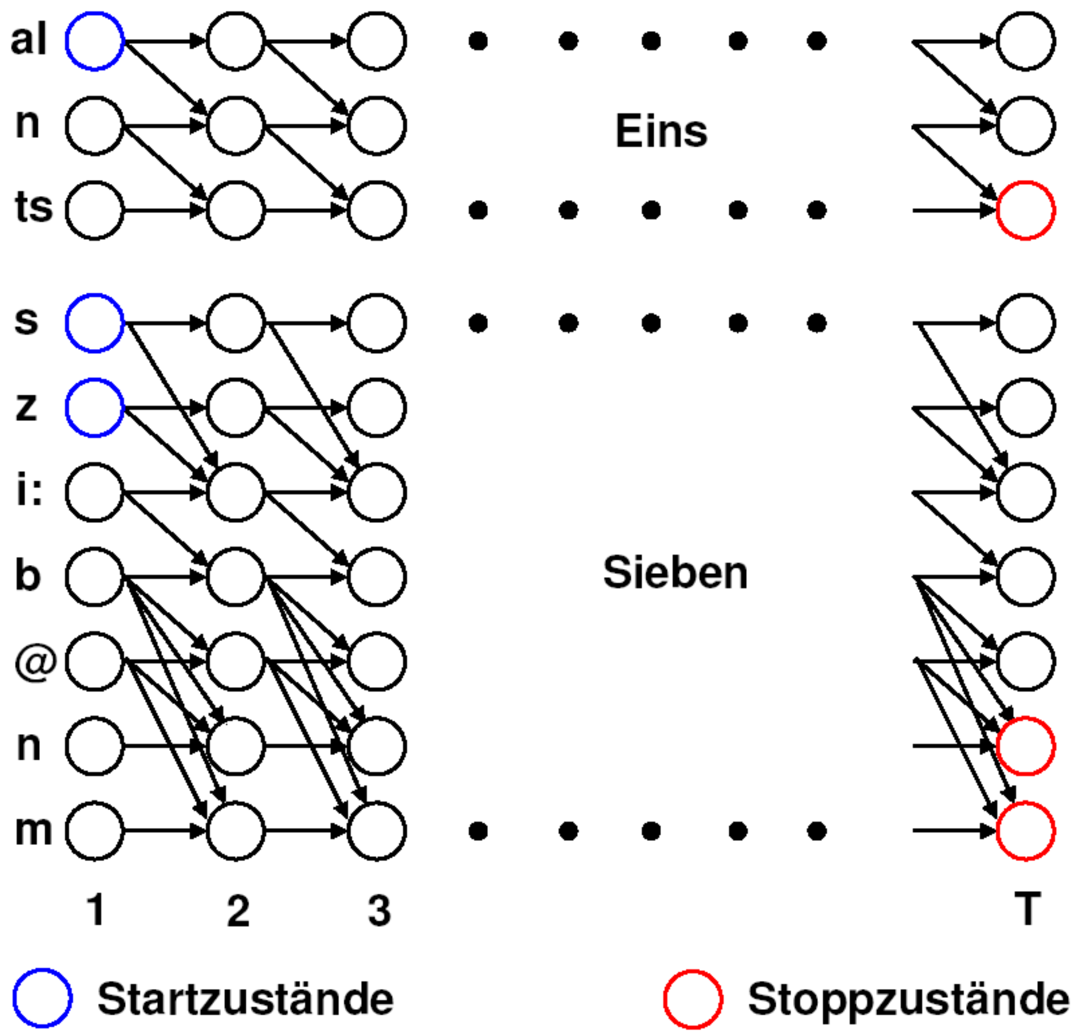
Score-Matrix



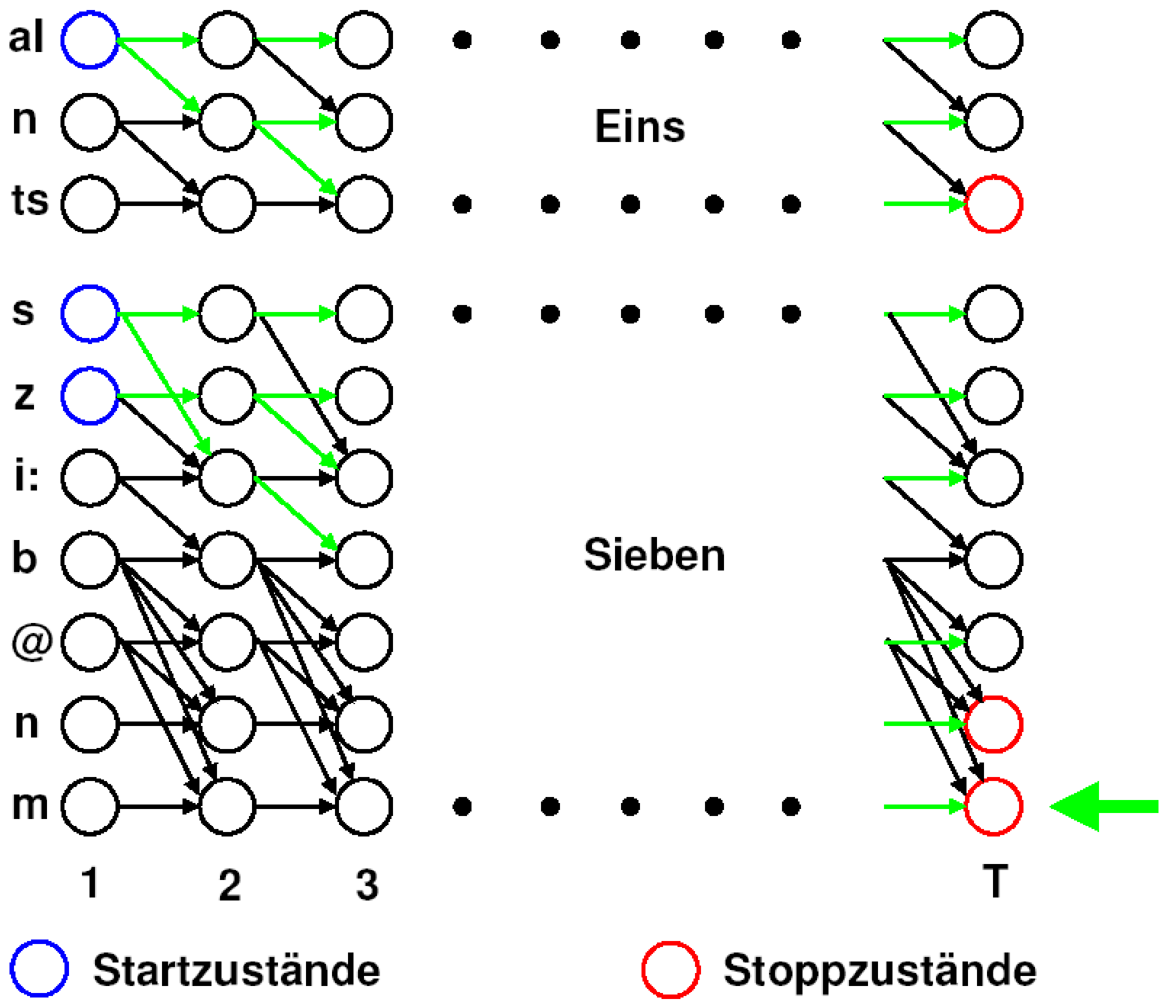
Wegkosten und Backtrack



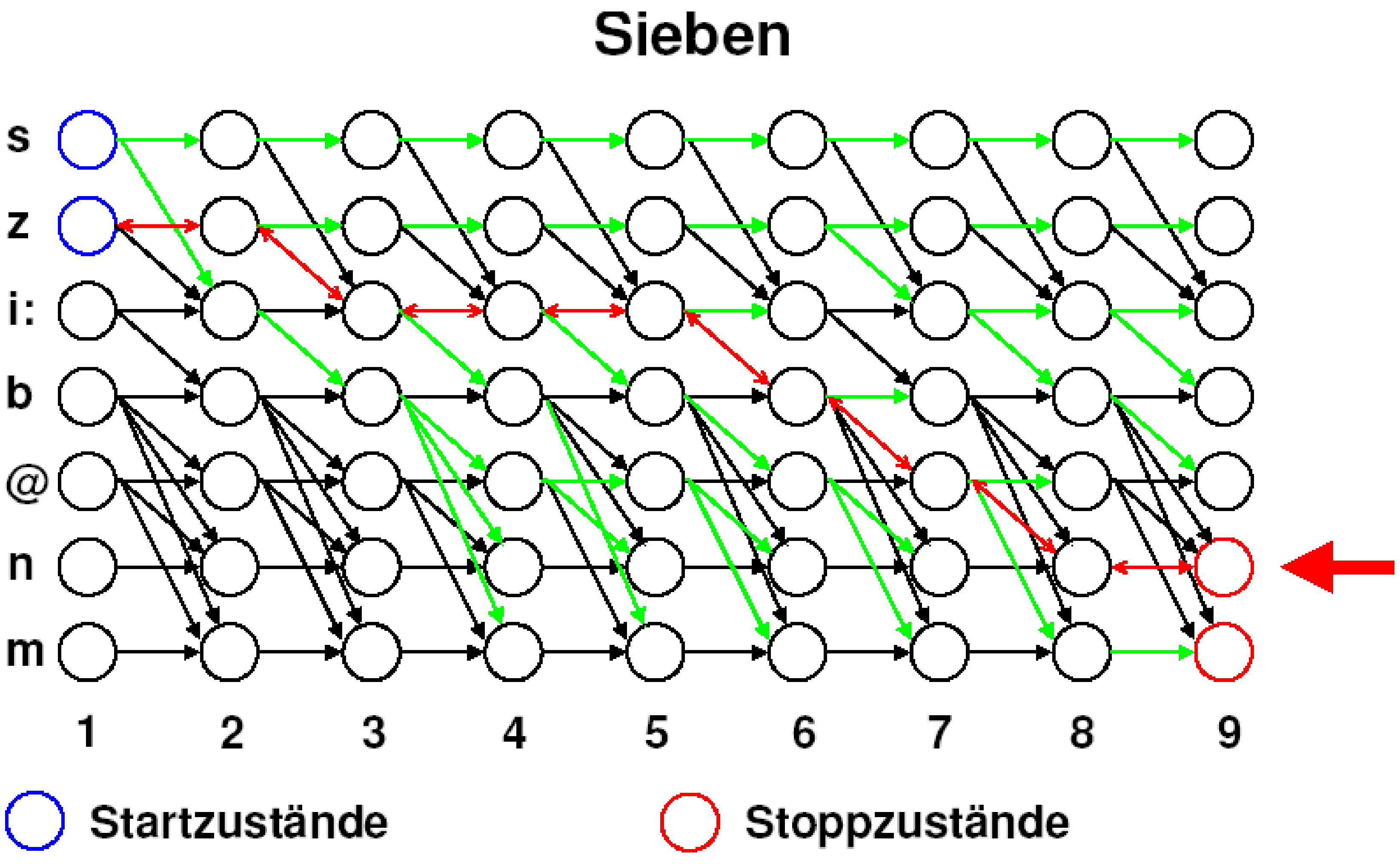
Viterbi (Schema)



Viterbi (Forward)



Viterbi (Backtrack)



Viterbi I

Ziel: $Z^* = \arg\max_{W_i} B(Z_i)$

wobei: Z^* Zustandsfolge mit maximaler Bewertung

$B(Z_i)$ Bewertung der Zustandsfolge i

Initialisierung: for($i=1; i \leq N_Z; i++$)

$$S_i(1) = \log(\pi_i) + b_i(\underline{m}(1));$$

wobei: N_Z alle Zustände der Markovkette

S_i akkumulierte Bewertung für Zustand i

π_i Initialisierungswahrscheinlichkeit für Zustand i

Viterbi II

Sukzessive Berechnung der akkumulierten Bewertung

for(t=2;t $\leq$ T;t++)

for(i=1;i $\leq$ N_Z;$i++$)

// Maximum über alle möglichen Vorgänger

$$S_i(t) = \max_{j \in V_i}[b_i(\underline{m}(t)) + \log(a_{ji}) + S_j(t-1)]$$

// Zeiger für spätere Decodierung

$$R_i(t) = \arg\max_{j \in V_i}[b_i(\underline{m}(t)) + \log(a_{ji}) + S_j(t-1)]$$

Viterbi III (Backtrack)

Initialisierung:

$$E(T) = \arg\max S_i(T)$$

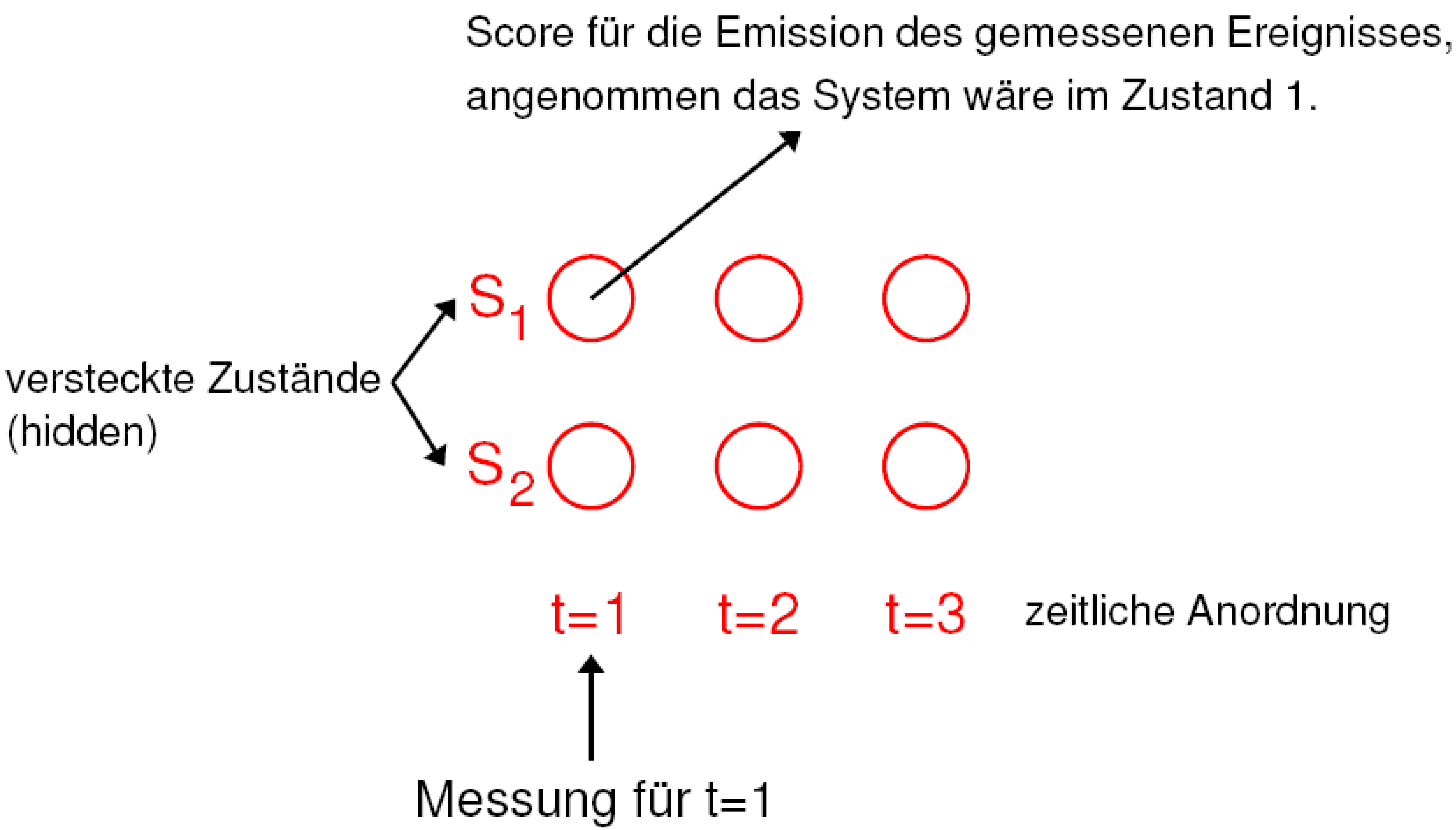
wobei: E(T) Ergebnisvektor

Sukzessive Decodierung:

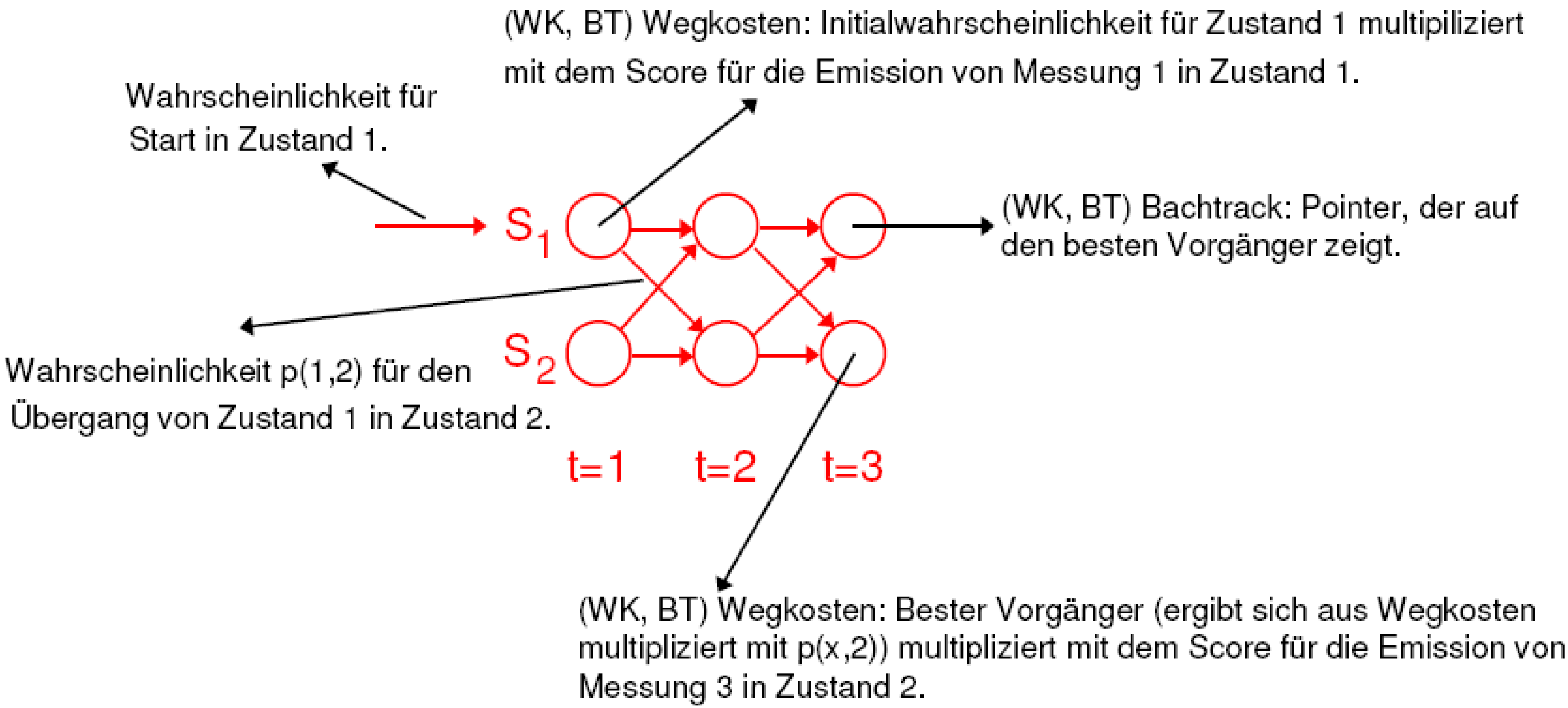
for(t=T; t ≥ 2;t--)

$$E(t-1) = R_{E(t)}(t)$$

Score-Matrix



Wegkosten und Backtrack



Übung: Viterbi (1)

Bitte ermitteln Sie die wahrscheinlichste Sequenz für zwei Münzwürfe unter der Annahme, dass bei beiden Münzwürfen das Ergebnis „Kopf“ ist.

Emissionswahrscheinlichkeiten

p (Kopf | UM):	0.5
p (Zahl | UM):	0.5
p (Kopf | MM):	0.8
p (Zahl | MM):	0.2
p_{init} (UM):	0.6
p_{init} (MM):	0.4

Score-Matrix

UM	0.5	0.5
MM	0.8	0.8
	1	2

Übergangswahrscheinlichkeiten

	UM	MM
UM	0.95	0.05
MM	0.96	0.04

Wegkosten, Backtrack

UM	0.3, UM	0.1536, MM
MM	0.32, MM	0.012, UM
	1	2

0.3 * 0.05 = 0.015; 0.32 * 0.04 = 0.0128

Übung: Viterbi (2)

Scores		
UM	0.5	0.5
MM	0.8	0.8
	1	2

Wegkosten		
UM	0.3, UM	0.1536, MM
MM	0.32, MM	0.012, UM
	1	2

Sequenz	p_{init}	p(K)	p(Ü)	p(K)	p (Sequenz)
UM, UM:	0.6	0.5	0.95	0.5	14.25%
UM, MM:	0.6	0.5	0.05	0.8	1.20%
MM, UM:	0.4	0.8	0.96	0.5	15.36%
MM, MM:	0.4	0.8	0.04	0.8	1.02%

Beispiel 1

	Wiesbaden		Frankfurt		Offenbach		Darmstadt	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
Bier	3.5	0.9	5.0	0.7	2.2	0.9	1.8	1.4
Wein	4.3	1.2	6.3	1.0	3.0	1.2	4.1	1.9
Pizza	7.5	1.8	9.5	1.6	5.5	1.8	4.3	2.6
Pasta	9.3	1.5	10.3	1.3	4.5	1.5	9.1	3.1

	Bier	Wein	Pizza	Pasta
1	3.5	4.3	7.5	9.3
2	5.0	6.3	9.5	10.3
3	2.2	3.0	5.5	4.5
4	1.8	4.1	4.3	9,1

Erwartung?

Wi-F-Of-Da

Analyse 1

Scores:

-4.210	-8.571	-14.644	-8.690
-9.104	-3.863	-32.846	-22.601
-14.644	-28.761	-4.210	-12.256
-8.226	-15.570	-9.273	-5.207

Wegkosten, Backpointer:

-5.60,1	-14.52,1	-29.52,1	-28.17,3
-10.49,2	-11.07,1	-44.08,2	-42.08,3
-16.03,3	-37.35,1	-17.18,2	-29.94,3
-9.61,4	-24.16,1	-26.79,1	-23.99,3

Hypothese: Wi-F-Of-Da

P_{init}: 0.25 (ln(0.25) = -1.39)

Übergang Of-Da: 0.2 (ln(0.2) = -1.61)

Beispiel 2

	Wiesbaden		Frankfurt		Offenbach		Darmstadt	
	MW	V	MW	V	MW	V	MW	V
Bier	3.5	0.9	5.0	0.7	2.2	0.9	1.8	1.4
Wein	4.3	1.2	6.3	1.0	3.0	1.2	4.1	1.9
Pizza	7.5	1.8	9.5	1.6	5.5	1.8	4.3	2.6
Pasta	9.3	1.5	10.3	1.3	4.5	1.5	9.1	3.1

	Bier	Wein	Pizza	Pasta
1	5.0	6.3	9.5	10.3
2	3.5	4.3	7.5	9.3
3	2.2	3.0	5.5	4.5
4	1.8	4.1	4.3	9,1

Erwartung?

F-Wi-Of-Da

Analyse 2

Scores:

-8.571	-4.210	-14.644	-8.690
-3.863	-9.104	-32.846	-22.601
-28.761	-14.644	-4.210	-12.256
-15.570	-8.226	-9.273	-5.207

Wegkosten, Backpointer:

-9.96,1	-14.52,1	-29.52,1	-31.62,3
-5.25,2	-14.52,2	-47.52,2	-45.53,3
-30.15,3	-21.79,2	-20.62,2	-33.39,3
-16.96,4	-21.18,1	-26.79,1	-27.44,3

Hypothese: F-F-Of-Da

P_{init}: 0.25 (ln(0.25) = -1.39)

Übergang F-Wi: 0.0 (ln(0.0) = -inf)

Beispiel 3

	Wiesbaden		Frankfurt		Offenbach		Darmstadt	
	MW	V	MW	V	MW	V	MW	V
Bier	3.5	0.9	5.0	0.7	2.2	0.9	1.8	1.4
Wein	4.3	1.2	6.3	1.0	3.0	1.2	4.1	1.9
Pizza	7.5	1.8	9.5	1.6	5.5	1.8	4.3	2.6
Pasta	9.3	1.5	10.3	1.3	4.5	1.5	9.1	3.1

	Bier	Wein	Pizza	Pasta
1	2.0	3.3	6.5	6.3
2	5.5	4.3	7.5	10.3
3	2.2	3.0	5.5	4.5
4	1.8	4.1	4.3	6.1

Erwartung?

Of-F-Of-Da

Analyse 3

Scores:

-9.154	-6.765	-14.644	-12.090
-23.758	-7.291	-32.846	-28.831
-5.627	-23.288	-4.210	-6.056
-7.585	-12.309	-9.273	-6.659

Wegkosten, Backpointer:

-10.54,1	-16.08,3	-31.08,1	-37.11,3
-25.14,2	-16.61,3	-49.62,2	-53.85,3
-7.01,3	-30.81,3	-22.71,2	-29.28,3
-8.97,4	-20.93,3	-28.35,1	-30.98,3

Hypothese: Of-F-Of-Of

P_{init}: 0.25 (ln(0.25) = -1.39)

Übergang Wi-Of: 0.05 (ln(0.05) = -3.0)

Übergang F-Of: 0.15 (ln(0.15) = -1.9)

Hidden Markov Modell (HMM)

- Es existiert eine Modellierung für die Abfolge der Zustände (Markov-Kette).
- Es existiert eine zustandsabhängige Modellierung von Merkmalen (Bsp. Preise in der Gastronomie).
- Die Zustände können nicht beobachtet werden.
- Merkmale können hingegen beobachtet werden.
- Aus den beobachteten Merkmalen soll auf die Abfolge der Zustände geschlossen werden.

Anwendungen HMM

- Mustererkennung
- Bioinformatik
- Computerlinguistik
- Zeitreihenanalyse

Klausur

Vorbemerkung: Für die Aufgaben 1 – 12 können Sie jeweils 10 Punkte erreichen. Wählen Sie bitte10 Aufgaben zur Bearbeitung aus. Bitte markieren Sie die von Ihnen bearbeiteten Aufgaben eindeutig.

Für Musik CDs werden Audiosignale mit 44,1 kHz abgetastet. In der digitalen Telefonie (ISDN) werden Sprachsignale mit 8 kHz abgetastet.

a) Bitte erläutern Sie die Hintergründe der unterschiedlichen Abtastraten.

b) Überschlagen Sie bitte welche Datenmenge (gemessen in kbit) für eine Sekunde Audio in beiden Fällen benötigt wird. Falls Sie für Ihre Schätzung Annahmen treffen, erklären Sie diese bitte.

Klausur

In welcher Situation wird der SNR zweier Signale gleich 0; in welcher Situation wird der SNR unendlich?

Welche Elemente der Kovarianzmatrix können negativ werden? Bitte erläutern Sie Ihre Antwort.

Warum wird die Sigmoidfunktion häufig als Transferfunktion bei mehrschichtigen Perzeptrons (MLPs) eingesetzt?

Warum werden die Gewichte für das Training Künstlicher Neuronaler Netze mit dem Back-Propagation Algorithmus zufällig initialisiert?

Bitte geben Sie zwei Beispiele für natürliche Zufallsprozesse.

Was ist ein Pseudo-Zufallsprozess?

Klausur

Montag, 04.07.2011, 18:00 – 19:30

Raum D14/104

Viel Erfolg!