Methoden des Maschinellen Lernens

Philipp Viertel   Patrick Palsbröker

FH Bielefeld Campus Minden

VL 05: Markow Modelle und Conditional Random Fields

16. Mai 2018

Einführung

Einführung

Lernen mit Struktur mittels graphischer Modelle

Einführung

Lernen mit Struktur mittels graphischer Modelle

- **Hidden Markov Model (HMM)**
- **Maximum-entropy Markov Model (MEMM)**
- **Markov Random Field (MRF)**
- **Conditional Random Field (CRF)**
- Bayes Netzwerke
- Structural SVM

Einführung

Lernen mit Struktur mittels graphischer Modelle

- **Hidden Markov Model (HMM)**
- **Maximum-entropy Markov Model (MEMM)**
- **Markov Random Field (MRF)**
- **Conditional Random Field (CRF)**
- Bayes Netzwerke
- Structural SVM

Verarbeitung von räumlichen und zeitlichen Verhältnissen (Sprache, Handlungen, Bilder, Geodaten)

Markow Ketten

Markow Ketten

- Markow Ketten sind die einfachsten Markow Modelle

Markow Ketten

- Markow Ketten sind die einfachsten Markow Modelle
- Sequenz zufälliger Variablen $X = (X_1, X_2, ...)$ mit multivariater Verteilung, bedingt durch $P(X_i \mid X_{i-1}, X_{i-2}, ..., X_1)$

Markow Ketten

- Markow Ketten sind die einfachsten Markow Modelle
- Sequenz zufälliger Variablen $X = (X_1, X_2, ...)$ mit multivariater Verteilung, bedingt durch $P(X_i \mid X_{i-1}, X_{i-2}, ..., X_1)$
- Beliebtes Wetter-Beispiel: $X_i \in \mathcal{L} = \{sunny, rainy\}$

Markow Ketten

- Markow Ketten sind die einfachsten Markow Modelle
- Sequenz zufälliger Variablen $X = (X_1, X_2, ...)$ mit multivariater Verteilung, bedingt durch $P(X_i \mid X_{i-1}, X_{i-2}, ..., X_1)$
- Beliebtes Wetter-Beispiel: $X_i \in \mathcal{L} = \{sunny, rainy\}$
- Das Wetter an Tag i kann beeinflusst sein durch das Wetter an vorherigen Tagen

Markow Ketten

- Markow Ketten sind die einfachsten Markow Modelle
- Sequenz zufälliger Variablen $X = (X_1, X_2, ...)$ mit multivariater Verteilung, bedingt durch $P(X_i \mid X_{i-1}, X_{i-2}, ..., X_1)$
- Beliebtes Wetter-Beispiel: $X_i \in \mathcal{L} = \{sunny, rainy\}$
- Das Wetter an Tag i kann beeinflusst sein durch das Wetter an vorherigen Tagen
- Explizit ist aber nur das Wetter vom Vortag beeinflussend (implizit alle vorherigen (*knock-on effect*))

Markow Ketten

- Markow Ketten sind die einfachsten Markow Modelle
- Sequenz zufälliger Variablen $X = (X_1, X_2, ...)$ mit multivariater Verteilung, bedingt durch $P(X_i \mid X_{i-1}, X_{i-2}, ..., X_1)$
- Beliebtes Wetter-Beispiel: $X_i \in \mathcal{L} = \{sunny, rainy\}$
- Das Wetter an Tag i kann beeinflusst sein durch das Wetter an vorherigen Tagen
- Explizit ist aber nur das Wetter vom Vortag beeinflussend (implizit alle vorherigen (*knock-on effect*))
- *Erste Ordnung* Annahme: $P(X_i \mid X_{i-1}, X_{i-2}, ..., X_1) = P(X_i \mid X_{i-1})$

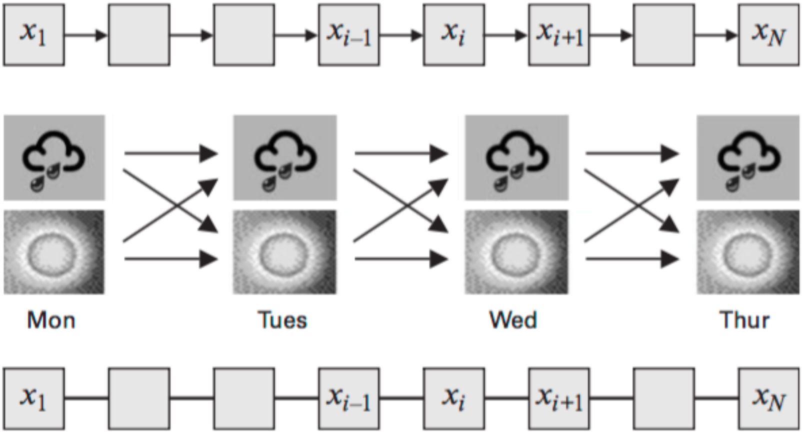


Abbildung: Quelle: [Blake et al.(2011)Blake, Kohli, and Rother]

		Gestern(X_{i-1})	
		Regen	Sonne
Heute(X_i)	Regen	0.4	0.8
	Sonne	0.6	0.2

- Die Kette von Beobachtungen einer Markow Kette erster Ordnung, besitzt folgende Wahrscheinlichkeitsverteilung bei N Beobachtungen:

- Die Kette von Beobachtungen einer Markow Kette erster Ordnung, besitzt folgende Wahrscheinlichkeitsverteilung bei N Beobachtungen:
- $p(x_1, ..., x_n) = \prod_{n=1}^{N} p(x_n \mid x_1, ..., x_{n-1}) = p(x_1) \prod_{n=2}^{N} p(x_n \mid x_{n-1})$

- Die Kette von Beobachtungen einer Markow Kette erster Ordnung, besitzt folgende Wahrscheinlichkeitsverteilung bei N Beobachtungen:
- $p(x_1, ..., x_n) = \prod_{n=1}^{N} p(x_n \mid x_1, ..., x_{n-1}) = p(x_1) \prod_{n=2}^{N} p(x_n \mid x_{n-1})$
- Markow Ketten werden uns wieder beim Thema Reinforcement Learning begegnen

- Die Kette von Beobachtungen einer Markow Kette erster Ordnung, besitzt folgende Wahrscheinlichkeitsverteilung bei N Beobachtungen:
- $p(x_1, ..., x_n) = \prod_{n=1}^{N} p(x_n \mid x_1, ..., x_{n-1}) = p(x_1) \prod_{n=2}^{N} p(x_n \mid x_{n-1})$
- Markow Ketten werden uns wieder beim Thema Reinforcement Learning begegnen
- Wichtiger in dieser VL und aufbauend auf Markow Ketten: **Hidden Markov Models**

Hidden Markov Models

Hidden Markov Models

- Modelle die Systeme abbilden auf Basis von Markow-Ketten und unbeobachteten (hidden) Zuständen

Hidden Markov Models

- Modelle die Systeme abbilden auf Basis von Markow-Ketten und unbeobachteten (hidden) Zuständen
- Markow-Kette: Das System geht auf zufällige Weise von einem Zustand in einen anderen über, Übergangswahrscheinlichkeiten hängen nur vom jeweils aktuellen Zustand ab und sind über die Zeit konstant

Hidden Markov Models

- Modelle die Systeme abbilden auf Basis von Markow-Ketten und unbeobachteten (hidden) Zuständen
- Markow-Kette: Das System geht auf zufällige Weise von einem Zustand in einen anderen über, Übergangswahrscheinlichkeiten hängen nur vom jeweils aktuellen Zustand ab und sind über die Zeit konstant
- **Ziel:** Aussagen zu treffen über die verborgenen Zustände auf Grundlage der beobachteten Sequenz

Hidden Markov Models

- Modelle die Systeme abbilden auf Basis von Markow-Ketten und unbeobachteten (hidden) Zuständen
- Markow-Kette: Das System geht auf zufällige Weise von einem Zustand in einen anderen über, Übergangswahrscheinlichkeiten hängen nur vom jeweils aktuellen Zustand ab und sind über die Zeit konstant
- **Ziel:** Aussagen zu treffen über die verborgenen Zustände auf Grundlage der beobachteten Sequenz
- Anwendung: u.a. Spracherkennung, Bioinformatik, Psychologie, Robotik

Kernidee:

Kernidee:

- Gegeben sind zwei diskrete Zufallsprozesse oder -zustände x_t, y_t

Kernidee:

- Gegeben sind zwei diskrete Zufallsprozesse oder -zustände x_t, y_t
- Nur y_t ist beobachtbar! $\rightarrow$ ein Modell soll die Wirkung bzw. den Verlauf von x_t erklären

Kernidee:

- Gegeben sind zwei diskrete Zufallsprozesse oder -zustände x_t, y_t
- Nur y_t ist beobachtbar! $\rightarrow$ ein Modell soll die Wirkung bzw. den Verlauf von x_t erklären
- HMMs erfüllen die zwei Markow Eigenschaften:
  - 1. Markow-Eigenschaft: Der aktuelle Wert des ersten Zustands hängt ausschließlich von seinem letzten Wert ab

Kernidee:

- Gegeben sind zwei diskrete Zufallsprozesse oder -zustände x_t, y_t
- Nur y_t ist beobachtbar! $\rightarrow$ ein Modell soll die Wirkung bzw. den Verlauf von x_t erklären
- HMMs erfüllen die zwei Markow Eigenschaften:
    - 1. Markow-Eigenschaft: Der aktuelle Wert des ersten Zustands hängt ausschließlich von seinem letzten Wert ab
    - 2. Markow-Eigenschaft: Der aktuelle Wert des zweiten Zustands hängt ausschließlich vom aktuellen Wert des ersten ab

Definition

Definition

$\lambda = (S, V, A, B, \pi)$

Definition

$\lambda = (S, V, A, B, \pi)$

- $S = s_1, ..., s_n$ Menge der Zustände, die möglichen Werte der verborgenen (hidden) Zufallsvariablen X_t

Definition

$\lambda = (S, V, A, B, \pi)$

- $S = s_1, ..., s_n$ Menge der Zustände, die möglichen Werte der verborgenen (hidden) Zufallsvariablen X_t
- $V = v_1, ..., v_m$ Menge der möglichen Beobachtungen (beobachtete Knoten/Emissionen) Y_t

Definition

$\lambda = (S, V, A, B, \pi)$

- $S = s_1, ..., s_n$ Menge der Zustände, die möglichen Werte der verborgenen (hidden) Zufallsvariablen X_t
- $V = v_1, ..., v_m$ Menge der möglichen Beobachtungen (beobachtete Knoten/Emissionen) Y_t
- A Matrix der Übergangswahrscheinlichkeit von einem Zustand zum nächsten (Transition features)

Definition

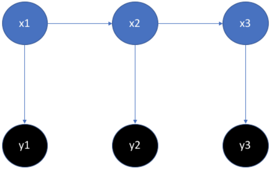
$\lambda = (S, V, A, B, \pi)$

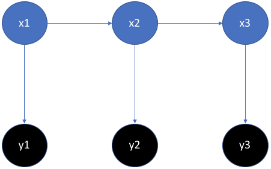
- $S = s_1, ..., s_n$ Menge der Zustände, die möglichen Werte der verborgenen (hidden) Zufallsvariablen X_t
- $V = v_1, ..., v_m$ Menge der möglichen Beobachtungen (beobachtete Knoten/Emissionen) Y_t
- A Matrix der Übergangswahrscheinlichkeit von einem Zustand zum nächsten (Transition features)
- B Beobachtungsmatrix mit $b_i(v_j)$, welche die Wahrscheinlichkeit angeben, in s_i v_j zu beobachten (State features)

Definition

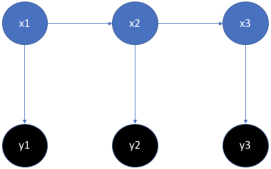
$\lambda = (S, V, A, B, \pi)$

- $S = s_1, ..., s_n$ Menge der Zustände, die möglichen Werte der verborgenen (hidden) Zufallsvariablen X_t
- $V = v_1, ..., v_m$ Menge der möglichen Beobachtungen (beobachtete Knoten/Emissionen) Y_t
- A Matrix der Übergangswahrscheinlichkeit von einem Zustand zum nächsten (Transition features)
- B Beobachtungsmatrix mit $b_i(v_j)$, welche die Wahrscheinlichkeit angeben, in s_i v_j zu beobachten (State features)
- π Anfangswahrscheinlichkeit, dass s_i der Startzustand ist: $\pi(i) = p(x_1 = x_i)$

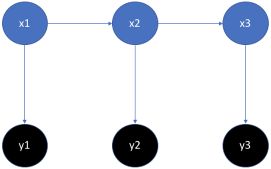

x1
x2
x3
y1
y2
y3



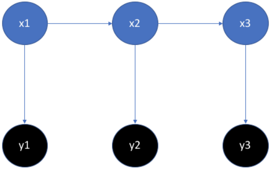
- Eine Beobachtung y_t hängt nur vom Zustand x_t ab $p(y_t \mid x_t)$



- Eine Beobachtung y_t hängt nur vom Zustand x_t ab $p(y_t \mid x_t)$
- Ein Zustand x_t hängt nur vom Vorgängerzustand ab $p(x_t \mid x_{t-1})$



- Eine Beobachtung y_t hängt nur vom Zustand x_t ab $p(y_t \mid x_t)$
- Ein Zustand x_t hängt nur vom Vorgängerzustand ab $p(x_t \mid x_{t-1})$
- Die Anfangswahrscheinlichkeit schreiben wir als $p(x_1 \mid x_0)$



- Eine Beobachtung y_t hängt nur vom Zustand x_t ab $p(y_t \mid x_t)$
- Ein Zustand x_t hängt nur vom Vorgängerzustand ab $p(x_t \mid x_{t-1})$
- Die Anfangswahrscheinlichkeit schreiben wir als $p(x_1 \mid x_0)$
- Die Wahrscheinlichkeit für die Zustandssequenz $\vec{x}$ und die Beobachtungssequenz $\vec{y}$ ist:

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

Drei wichtige Problemstellungen von HMMs

Drei wichtige Problemstellungen von HMMs

- **Filtern und Vorhersage**; Gegeben ist ein HMM und eine Beobachtungssequenz $o \in X$ der Länge T. Gesucht ist $P(o \mid \lambda)$. **Vorhersagen** ist ein wiederholtes Filtern $\rightarrow$ **Forward**-**Algorithmus**

Drei wichtige Problemstellungen von HMMs

- **Filtern und Vorhersage**; Gegeben ist ein HMM und eine Beobachtungssequenz $o \in X$ der Länge T. Gesucht ist $P(o \mid \lambda)$. **Vorhersagen** ist ein wiederholtes Filtern $\rightarrow$ **Forward-Algorithmus**
- **Dekodierung**; Ein HMM, sowie o ist gegeben. Die wahrscheinlichste Zustandsfolge aus X soll bestimmt werden, die eine vorgegebene Ausgabesequenz erzeugt haben könnte $\rightarrow$ **Viterbi-Algorithmus**

Drei wichtige Problemstellungen von HMMs

- **Filtern und Vorhersage**; Gegeben ist ein HMM und eine Beobachtungssequenz $o \in X$ der Länge T. Gesucht ist $P(o \mid \lambda)$. **Vorhersagen** ist ein wiederholtes Filtern $\rightarrow$ **Forward-Algorithmus**
- **Dekodierung**; Ein HMM, sowie o ist gegeben. Die wahrscheinlichste Zustandsfolge aus X soll bestimmt werden, die eine vorgegebene Ausgabesequenz erzeugt haben könnte $\rightarrow$ **Viterbi-Algorithmus**
- **Lernproblem**; Gegeben ist die Ausgabesequenz o, die Parameter des HMM sollen bestimmt werden, die am wahrscheinlichsten o erzeugen $\rightarrow$ **Baum-Welch-Algorithmus**

Forward-Algorithmus

Forward-Algorithmus

- Nutzt sogenannte Forward-Variablen zur Berechnung von Wahrscheinlichkeiten von bestimmten Beobachtungen

Forward-Algorithmus

- Nutzt sogenannte Forward-Variablen zur Berechnung von Wahrscheinlichkeiten von bestimmten Beobachtungen
- Basiert auf dynamischer Programmierung

- Gegeben: HMM Modell $\lambda = (S, V, A, B, \pi)$ und eine Sequenz $o = o_1 o_2 ... o_T \in V$

- Gegeben: HMM Modell $\lambda = (S, V, A, B, \pi)$ und eine Sequenz $o = o_1 o_2 ... o_T \in V$
- Algorithmus berechnet $P(o \mid \lambda)$ (die Wahrscheinlichkeit im vorhandenen Modell tatsächlich die Beobachtung o zu machen)

- Gegeben: HMM Modell $\lambda = (S, V, A, B, \pi)$ und eine Sequenz $o = o_1 o_2 ... o_T \in V$
- Algorithmus berechnet $P(o \mid \lambda)$ (die Wahrscheinlichkeit im vorhandenen Modell tatsächlich die Beobachtung o zu machen)
- Forward-Variablen $a_t(i)$ beschreiben die Wahrscheinlichkeit zum Zeitpunkt $1 \leq t \leq T$ $o_1 o_2 ... o_t$ beobachtet zu haben und im Zustand $s_i \in S$ zu sein; $a_t(i) = P(o_1 o_2 ... o_t; q_t = s_i \mid \lambda)$

Initialisierung

$$a_1(i) = \pi \cdot b_i(o_1) \qquad 1 \leq i \leq |S|$$

Initialisierung

$$a_1(i) = \pi \cdot b_i(o_1) \qquad 1 \leq i \leq |S|$$

Rekursion

$$a_t(i) = a_t(i)\left(\sum_{j=1}^{|S|} a_{t-1}(j)a_{ji}\right) \cdot b_i(o_t) \qquad 1 < t \leq T, 1 \leq i \leq |S|$$

Initialisierung

$$a_1(i) = \pi \cdot b_i(o_1) \qquad 1 \leq i \leq |S|$$

Rekursion

$$a_t(i) = a_t(i) \left(\sum_{j=1}^{|S|} a_{t-1}(j) a_{ji} \right) \cdot b_i(o_t) \qquad 1 < t \leq T, 1 \leq i \leq |S|$$

Terminierung

$$P(o \mid \lambda) = \sum_{j=1}^{|S|} a_T(j)$$

Viterbi-Algorithmus

Viterbi-Algorithmus

- Gegeben: HMM Modell $\lambda = (S, V, A, B, \pi)$ und eine Sequenz $o = o_1 o_2 ... o_T \in V$

Viterbi-Algorithmus

- Gegeben: HMM Modell $\lambda = (S, V, A, B, \pi)$ und eine Sequenz $o = o_1 o_2 ... o_T \in V$
- Die wahrscheinlichste Zustandsfolge $q^* = q_1^* q_2^* ... q_T^* \in S^T$ soll berechnet werden

Viterbi-Algorithmus

- Gegeben: HMM Modell $\lambda = (S, V, A, B, \pi)$ und eine Sequenz $o = o_1 o_2 ... o_T \in V$
- Die wahrscheinlichste Zustandsfolge $q^* = q_1^* q_2^* ... q_T^* \in S^T$ soll berechnet werden
- q^* ist die Sequenz von verborgenen Zuständen, die unter allen Folgen q der Länge T den Wert von $P(q \mid o; \lambda)$ maximiert

Viterbi-Algorithmus

- Gegeben: HMM Modell $\lambda = (S, V, A, B, \pi)$ und eine Sequenz $o = o_1 o_2 ... o_T \in V$
- Die wahrscheinlichste Zustandsfolge $q^* = q_1^* q_2^* ... q_T^* \in S^T$ soll berechnet werden
- q^* ist die Sequenz von verborgenen Zuständen, die unter allen Folgen q der Länge T den Wert von $P(q \mid o; \lambda)$ maximiert
- Es gilt: $P(o; q^* \mid \lambda) = \max_{q \in S^T} P(o; q \mid \lambda)$

- Viterbi-Algorithmus verwendet zwei Variablen:

- Viterbi-Algorithmus verwendet zwei Variablen:
- $\vartheta_t(i)$ ist die **maximale Verbundwahrscheinlichkeit** zum Zeitpunkt $1 \leq t \leq T$ bei der Beobachtung von $o_1 o_2 ... o_t$ durch eine Zustandsfolge der Länge t gelaufen zu sein und im Zustand $s_i \in S$ zu enden: $\vartheta_t(i) = \max\limits_{\substack{q \in S^T \\ q_t = s_i}} P(o_1 o_2 ... o_t ; q_1 q_2 ... q_t \mid \lambda)$

- Viterbi-Algorithmus verwendet zwei Variablen:
- $\vartheta_t(i)$ ist die **maximale Verbundwahrscheinlichkeit** zum Zeitpunkt $1 \leq t \leq T$ bei der Beobachtung von $o_1 o_2 ... o_t$ durch eine Zustandsfolge der Länge t gelaufen zu sein und im Zustand $s_i \in S$ zu enden: $\vartheta_t(i) = \max_{\substack{q \in S^T \\ q_t = s_i}} P(o_1 o_2 ... o_t ; q_1 q_2 ... q_t \mid \lambda)$
- $\psi_t(i)$ speichert für jeden Zeitpunkt und jeden Zustand, welcher **Vorgängerzustand** an der Maximumsbildung beteiligt war

Initialisierung

$$\vartheta_1(i) = \pi_i \cdot b_i(o_1)$$
$$\psi_1(i) = 0 \qquad 1 \leq i \leq |S|$$

Initialisierung

$$\vartheta_1(i) = \pi_i \cdot b_i(o_1)$$
$$\psi_1(i) = 0 \qquad 1 \leq i \leq |S|$$

Rekursion

$$\vartheta_t(i) = b_i(o_1) \cdot \max_{1 \leq j \leq |S|} (a_{ji} \cdot \vartheta_{t-1}(j))$$
$$\psi_t(i) = \underset{i \leq j \leq |S|}{argmax} (a_{ji} \cdot \vartheta_{t-1}(j)) \qquad 1 \leq i \leq |S|, 1 < t \leq T$$

Initialisierung

$$\vartheta_1(i) = \pi_i \cdot b_i(o_1)$$
$$\psi_1(i) = 0 \qquad 1 \leq i \leq |S|$$

Rekursion

$$\vartheta_t(i) = b_i(o_1) \cdot \max_{1 \leq j \leq |S|} (a_{ji} \cdot \vartheta_{t-1}(j))$$
$$\psi_t(i) = \underset{i \leq j \leq |S|}{argmax}(a_{ji} \cdot \vartheta_{t-1}(j)) \qquad 1 \leq i \leq |S|, 1 < t \leq T$$

Terminierung

$$P(o; q^* \mid \lambda) = \max_{1 \leq j \leq |S|} \vartheta_T(j)$$
$$q_T^* = \underset{1 \leq j \leq |S|}{argmax} \vartheta_T(j)$$

Initialisierung

$$\vartheta_1(i) = \pi_i \cdot b_i(o_1)$$
$$\psi_1(i) = 0 \qquad 1 \leq i \leq |S|$$

Rekursion

$$\vartheta_t(i) = b_i(o_1) \cdot \max_{1 \leq j \leq |S|} (a_{ji} \cdot \vartheta_{t-1}(j))$$
$$\psi_t(i) = \underset{i \leq j \leq |S|}{argmax} (a_{ji} \cdot \vartheta_{t-1}(j)) \qquad 1 \leq i \leq |S|, 1 < t \leq T$$

Terminierung

$$P(o; q^* \mid \lambda) = \max_{1 \leq j \leq |S|} \vartheta_T(j)$$
$$q_T^* = \underset{1 \leq j \leq |S|}{argmax} \vartheta_T(j)$$

Pfadermittlung

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad 1 \leq t < T$$

Baum-Welch-Algorithmus

Baum-Welch-Algorithmus

- Basierend auf dem Forward-Algorithmus und dem Backward-Algorithmus bekommt man den Baum-Welch-Algorithmus

Baum-Welch-Algorithmus

- Basierend auf dem Forward-Algorithmus und dem Backward-Algorithmus bekommt man den Baum-Welch-Algorithmus
- Berechnet die Maximalwahrscheinlichkeitsschätzungen (Maximum-Likelihood) jedes Zustands

Baum-Welch-Algorithmus

- Basierend auf dem Forward-Algorithmus und dem Backward-Algorithmus bekommt man den Baum-Welch-Algorithmus
- Berechnet die Maximalwahrscheinlichkeitsschätzungen (Maximum-Likelihood) jedes Zustands
- Berechnet daraufhin die Frequenz der Übergangs-Zustands-Paar-Werte

Baum-Welch-Algorithmus

- Basierend auf dem Forward-Algorithmus und dem Backward-Algorithmus bekommt man den Baum-Welch-Algorithmus
- Berechnet die Maximalwahrscheinlichkeitsschätzungen (Maximum-Likelihood) jedes Zustands
- Berechnet daraufhin die Frequenz der Übergangs-Zustands-Paar-Werte
- Spezialisierte Version des **EM Algorithmus** (Clusteranalyse)

Maximum Entropy Markov Model

Maximum Entropy Markov Model

- Maximum Entropy Markov Model (MEMM)

Maximum Entropy Markov Model

- Maximum Entropy Markov Model (MEMM)
- Eine Aufgabe für das Praktikum :)

Darstellung als Graphen

Darstellung als Graphen

- Ein Bild lässt sich als Graph definieren (Beispiel)

Darstellung als Graphen

- Ein Bild lässt sich als Graph definieren (Beispiel)
- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ mit Knoten $\mathcal{V} = (1, ..., N)$ (Pixelwerte) und einer Menge von ungerichteten Kanten $(i, j), i, j \in \mathcal{V}$

Darstellung als Graphen

- Ein Bild lässt sich als Graph definieren (Beispiel)
- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ mit Knoten $\mathcal{V} = (1, ..., N)$ (Pixelwerte) und einer Menge von ungerichteten Kanten $(i, j), i, j \in \mathcal{V}$
- *Hidden* variables werden Knoten zugewiesen; z.B: X_i mit $i = 0$ oder $i = 1$ (Background/Foreground segmentation)

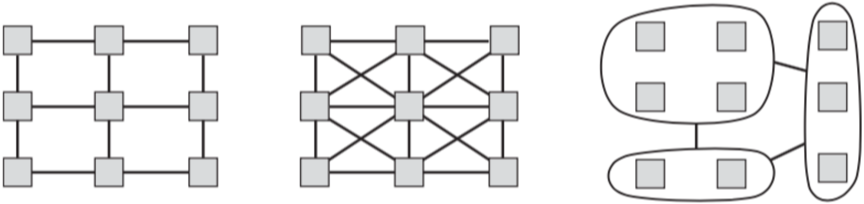


Abbildung: Graphen für visuelle Markow Modelle. Quelle: [Blake et al.(2011)Blake, Kohli, and Rother].

- Tendenzen von ähnlicher Struktur oder Farbe (Histogramme) müssen abgebildet werden

- Tendenzen von ähnlicher Struktur oder Farbe (Histogramme) müssen abgebildet werden
- Benachbarte Pixel haben **wahrscheinlich** die gleichen Label

- Tendenzen von ähnlicher Struktur oder Farbe (Histogramme) müssen abgebildet werden
- Benachbarte Pixel haben **wahrscheinlich** die gleichen Label
- Diese probabilistische Tendenz für $(i, j) \in \mathcal{E}$ führt zu gleichen Labeln X_i, X_j

- Tendenzen von ähnlicher Struktur oder Farbe (Histogramme) müssen abgebildet werden
- Benachbarte Pixel haben **wahrscheinlich** die gleichen Label
- Diese probabilistische Tendenz für $(i, j) \in \mathcal{E}$ führt zu gleichen Labeln X_i, X_j
- **Wichtig:** Es gibt keine explizite Verbindung zwischen solchen Pixeln, dies würde zu einem sehr dichten Modell führen mit sehr aufwendigen Algorithmen! Markow Modelle definieren nur die Beziehungen zwischen wenigen Paaren von Pixeln (definiert über geteilte Kanten in $\mathcal{E}$)

- Tendenzen von ähnlicher Struktur oder Farbe (Histogramme) müssen abgebildet werden
- Benachbarte Pixel haben **wahrscheinlich** die gleichen Label
- Diese probabilistische Tendenz für $(i, j) \in \mathcal{E}$ führt zu gleichen Labeln X_i, X_j
- **Wichtig:** Es gibt keine explizite Verbindung zwischen solchen Pixeln, dies würde zu einem sehr dichten Modell führen mit sehr aufwendigen Algorithmen! Markow Modelle definieren nur die Beziehungen zwischen wenigen Paaren von Pixeln (definiert über geteilte Kanten in $\mathcal{E}$)
- **Vorteil:** Markow Modelle haben die Eigenschaft, dass über die short-range Verbindungen long-range Korrelationen implizit geschaffen werden *(knock-on effect)*

Markov Random Fields

Markov Random Fields

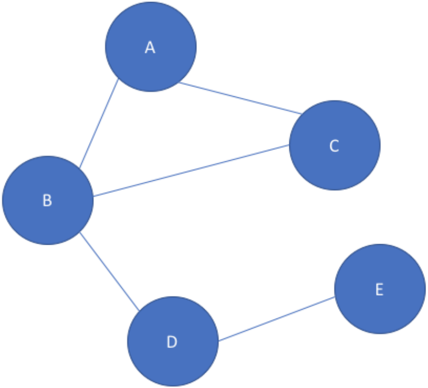


Abbildung: Beispiel eines MRF. Kanten repräsentieren Abhängigkeiten.

- Kurz: Ein Markov Random Field ist eine Menge von zufälligen Variablen, welche die Markov Eigenschaft erfüllen

- Kurz: Ein Markov Random Field ist eine Menge von zufälligen Variablen, welche die Markov Eigenschaft erfüllen
- Ungerichtetes graphisches Modell, welches Zyklen erlaubt

- Kurz: Ein Markov Random Field ist eine Menge von zufälligen Variablen, welche die Markov Eigenschaft erfüllen
- Ungerichtetes graphisches Modell, welches Zyklen erlaubt
- Vorteile gegenüber gerichteten Modellen:

- Kurz: Ein Markov Random Field ist eine Menge von zufälligen Variablen, welche die Markov Eigenschaft erfüllen
- Ungerichtetes graphisches Modell, welches Zyklen erlaubt
- Vorteile gegenüber gerichteten Modellen:
  - Symmetrisch, was besser geeignet ist für räumliche und relationale Daten

- Kurz: Ein Markov Random Field ist eine Menge von zufälligen Variablen, welche die Markov Eigenschaft erfüllen
- Ungerichtetes graphisches Modell, welches Zyklen erlaubt
- Vorteile gegenüber gerichteten Modellen:
  - Symmetrisch, was besser geeignet ist für räumliche und relationale Daten
- Nachteile:

- Kurz: Ein Markov Random Field ist eine Menge von zufälligen Variablen, welche die Markov Eigenschaft erfüllen
- Ungerichtetes graphisches Modell, welches Zyklen erlaubt
- Vorteile gegenüber gerichteten Modellen:
  - Symmetrisch, was besser geeignet ist für räumliche und relationale Daten
- Nachteile:
  - Parameter sind weniger interpretierbar und modular

- Kurz: Ein Markov Random Field ist eine Menge von zufälligen Variablen, welche die Markov Eigenschaft erfüllen
- Ungerichtetes graphisches Modell, welches Zyklen erlaubt
- Vorteile gegenüber gerichteten Modellen:
  - Symmetrisch, was besser geeignet ist für räumliche und relationale Daten
- Nachteile:
  - Parameter sind weniger interpretierbar und modular
  - Parameterbestimmung ist rechenaufwendiger

Definition:

Definition:

- Ein Markov Random Field (MRF) beschreibt eine Wahrscheinlichkeitsverteilung P über die Variablen $x_1, ..., x_n$ in einem **ungerichteten** Graphen G, in dem die Knoten mit den Variablen x_i korrespondieren

Definition:

- Ein Markov Random Field (MRF) beschreibt eine Wahrscheinlichkeitsverteilung P über die Variablen $x_1, ..., x_n$ in einem **ungerichteten** Graphen G, in dem die Knoten mit den Variablen x_i korrespondieren
- $P(x_1, ..., x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(x_c)$

Definition:

- Ein Markov Random Field (MRF) beschreibt eine Wahrscheinlichkeitsverteilung P über die Variablen $x_1, ..., x_n$ in einem **ungerichteten** Graphen G, in dem die Knoten mit den Variablen x_i korrespondieren
- $P(x_1, ..., x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(x_c)$
- C ist die Menge der **Cliquen** (vollständig verbundene Subgraphen von G)

Definition:

- Ein Markov Random Field (MRF) beschreibt eine Wahrscheinlichkeitsverteilung P über die Variablen $x_1, ..., x_n$ in einem **ungerichteten** Graphen G, in dem die Knoten mit den Variablen x_i korrespondieren
- $P(x_1, ..., x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(x_c)$
- C ist die Menge der **Cliquen** (vollständig verbundene Subgraphen von G)
- Partitionsfunktion: $Z = \sum_{x_1,\ldots,x_n} \prod_{c \in C} \phi_c(x_c)$

Ising Modell

- Beispiel: **Ising Modell** (Modell aus der statistischen Physik)

Ising Modell

- Beispiel: **Ising Modell** (Modell aus der statistischen Physik)
- Modellierung des Verhaltens von Atomen; $y_s \in \{-1, +1\}$ definiert die Drehung eines Atoms

Ising Modell

- Beispiel: **Ising Modell** (Modell aus der statistischen Physik)
- Modellierung des Verhaltens von Atomen; $y_s \in \{-1, +1\}$ definiert die Drehung eines Atoms
- Ferro-Magneten; benachbarte Ausrichtungen *wollen* sich angleichen

Ising Modell

- Beispiel: **Ising Modell** (Modell aus der statistischen Physik)
- Modellierung des Verhaltens von Atomen; $y_s \in \{-1, +1\}$ definiert die Drehung eines Atoms
- Ferro-Magneten; benachbarte Ausrichtungen *wollen* sich angleichen
- Anti-Ferro-Magneten; benachbarte Ausrichtungen *wollen* sich unterscheiden

Ising Modell

- Beispiel: **Ising Modell** (Modell aus der statistischen Physik)
- Modellierung des Verhaltens von Atomen; $y_s \in \{-1, +1\}$ definiert die Drehung eines Atoms
- Ferro-Magneten; benachbarte Ausrichtungen *wollen* sich angleichen
- Anti-Ferro-Magneten; benachbarte Ausrichtungen *wollen* sich unterscheiden
- Lässt sich als MRF modellieren, ein Graph (Gitter) mit verbundenen, benachbarten Knoten bzw. Variablen

$$\psi_{st}(y_s, y_t) = \begin{pmatrix} e^{w_{st}} & e^{-w_{st}} \\ e^{-w_{st}} & e^{w_{st}} \end{pmatrix}$$

- w_{st} ist die Kupplungsstärke zwischen Knoten s und t, wenn keine Verbindung besteht dann $w_{st} = 0$
- Gewichtsmatrix W ist symmetrisch; $w_{st} = w_{ts}$
- Annahme, dass alle Kanten dieselbe Stärke haben $w_{st} = J$
- Wenn die Gewichte positiv sind $\rightarrow$ Ferro-Magneten, wenn die Gewichte negativ sind $\rightarrow$ Anti-Ferro-Magneten

Conditional Random Fields

Conditional Random Fields

- Was machen Conditional Random Fields anders? Wofür kann man sie einsetzen?

Conditional Random Fields

- Was machen Conditional Random Fields anders? Wofür kann man sie einsetzen?
- Ein Beispiel aus dem Bereich *Natural Language Processing (NLP)*

Conditional Random Fields

- Was machen Conditional Random Fields anders? Wofür kann man sie einsetzen?
- Ein Beispiel aus dem Bereich *Natural Language Processing (NLP)*
  - Ziel: Klassifikation einer Folge von Elementen (Wörter), d.h. die Zuordnung jedes Elements zu einer Klasse bzw. einem Label
  - Die Reihenfolge der Wörter eines Satzes ist sehr wichtig für die korrekte Klassifizierung
  - Beispiel: Das Wort 'zu' wird am Ende eines Satzes nie eine Präposition sein

Conditional Random Fields

- Was machen Conditional Random Fields anders? Wofür kann man sie einsetzen?
- Ein Beispiel aus dem Bereich *Natural Language Processing (NLP)*
  - Ziel: Klassifikation einer Folge von Elementen (Wörter), d.h. die Zuordnung jedes Elements zu einer Klasse bzw. einem Label
  - Die Reihenfolge der Wörter eines Satzes ist sehr wichtig für die korrekte Klassifizierung
  - Beispiel: Das Wort 'zu' wird am Ende eines Satzes nie eine Präposition sein
  - Aufgabenstellung die sich mittels Wahrscheinlichkeitsmodelle lösen lassen
    - HMM, MEMM und **Conditional Random Fields (CRF)**

- Supervised Lernverfahren: Mit Hilfe der annotierten Trainingsdaten wird eine Wahrscheinlichkeitsverteilung berechnet um das beste Label zu bestimmen

- Supervised Lernverfahren: Mit Hilfe der annotierten Trainingsdaten wird eine Wahrscheinlichkeitsverteilung berechnet um das beste Label zu bestimmen
- Kann eingesetzt werden für eine Reihe von Tasks in NLP:
  - Information Extraction (IE)

- Supervised Lernverfahren: Mit Hilfe der annotierten Trainingsdaten wird eine Wahrscheinlichkeitsverteilung berechnet um das beste Label zu bestimmen
- Kann eingesetzt werden für eine Reihe von Tasks in NLP:
  - Information Extraction (IE)
  - Shallow Parsing

- Supervised Lernverfahren: Mit Hilfe der annotierten Trainingsdaten wird eine Wahrscheinlichkeitsverteilung berechnet um das beste Label zu bestimmen
- Kann eingesetzt werden für eine Reihe von Tasks in NLP:
  - Information Extraction (IE)
  - Shallow Parsing
  - Named Entity Recognition

- Supervised Lernverfahren: Mit Hilfe der annotierten Trainingsdaten wird eine Wahrscheinlichkeitsverteilung berechnet um das beste Label zu bestimmen
- Kann eingesetzt werden für eine Reihe von Tasks in NLP:
  - Information Extraction (IE)
  - Shallow Parsing
  - Named Entity Recognition
  - Part-of-Speech Tagging

- Supervised Lernverfahren: Mit Hilfe der annotierten Trainingsdaten wird eine Wahrscheinlichkeitsverteilung berechnet um das beste Label zu bestimmen
- Kann eingesetzt werden für eine Reihe von Tasks in NLP:
  - Information Extraction (IE)
  - Shallow Parsing
  - Named Entity Recognition
  - Part-of-Speech Tagging
- Außerhalb des Bereichs NLP werden CRFs besonders im Bereich Bildverarbeitung angewendet

- HMMs modellieren viele Wahrscheinlichkeiten, die man eigentlich gar nicht benötigt

- HMMs modellieren viele Wahrscheinlichkeiten, die man eigentlich gar nicht benötigt
- HMMs modelliert die vollständige multivariate Wahrscheinlichkeitsverteilung von Beobachtungen und versteckten Zuständen und kann auch Beispiele dadurch generieren (**generatives** Modell)

- HMMs modellieren viele Wahrscheinlichkeiten, die man eigentlich gar nicht benötigt
- HMMs modelliert die vollständige multivariate Wahrscheinlichkeitsverteilung von Beobachtungen und versteckten Zuständen und kann auch Beispiele dadurch generieren (**generatives** Modell)
- Oft benötigt man nur ein **diskriminatives** Modell um die bedingten Wahrscheinlichkeiten der versteckten Attribute, gegeben einer Beobachtung (z.B. Text) zu modellieren

- HMMs modellieren viele Wahrscheinlichkeiten, die man eigentlich gar nicht benötigt
- HMMs modelliert die vollständige multivariate Wahrscheinlichkeitsverteilung von Beobachtungen und versteckten Zuständen und kann auch Beispiele dadurch generieren (**generatives** Modell)
- Oft benötigt man nur ein **diskriminatives** Modell um die bedingten Wahrscheinlichkeiten der versteckten Attribute, gegeben einer Beobachtung (z.B. Text) zu modellieren
  - Gegeben ist ein Text $e_{1:N}$

- HMMs modellieren viele Wahrscheinlichkeiten, die man eigentlich gar nicht benötigt
- HMMs modelliert die vollständige multivariate Wahrscheinlichkeitsverteilung von Beobachtungen und versteckten Zuständen und kann auch Beispiele dadurch generieren (**generatives** Modell)
- Oft benötigt man nur ein **diskriminatives** Modell um die bedingten Wahrscheinlichkeiten der versteckten Attribute, gegeben einer Beobachtung (z.B. Text) zu modellieren
    - Gegeben ist ein Text $e_{1:N}$
    - Ein bedingtes Modell findet die versteckte Zustandssequenz $X_{1:N}$ welche $P(X_{1:N} \mid e_{1:N})$ maximiert

- Markov Modelle besitzen die Unabhängigkeitsannahme, bei CRFs haben wir ein x_t welches abhängig ist von x_1

- Markov Modelle besitzen die Unabhängigkeitsannahme, bei CRFs haben wir ein x_t welches abhängig ist von x_1
- **Kurz:** Ein CRF modelliert die Wahrscheinlichkeitsverteilung einer Menge von Zielvariablen, gegeben einer Menge von beobachteten Variablen

- Markov Modelle besitzen die Unabhängigkeitsannahme, bei CRFs haben wir ein x_t welches abhängig ist von x_1
- **Kurz:** Ein CRF modelliert die Wahrscheinlichkeitsverteilung einer Menge von Zielvariablen, gegeben einer Menge von beobachteten Variablen
- CRFs kann viele Strukturen von Abhängigkeiten zwischen den Variablen modellieren

- Markov Modelle besitzen die Unabhängigkeitsannahme, bei CRFs haben wir ein x_t welches abhängig ist von x_1
- **Kurz:** Ein CRF modelliert die Wahrscheinlichkeitsverteilung einer Menge von Zielvariablen, gegeben einer Menge von beobachteten Variablen
- CRFs kann viele Strukturen von Abhängigkeiten zwischen den Variablen modellieren
- Häufige Variante: **Linear-chain CRF**

Linear-chain CRF

Linear-chain CRF

- Sei $e_{1:N}$ eine Beobachtung (z.B. Wörter in einem Dokument)

Linear-chain CRF

- Sei $e_{1:N}$ eine Beobachtung (z.B. Wörter in einem Dokument)
- Sei $x_{1:N}$ eine Sequenz von versteckten Zuständen

Linear-chain CRF

- Sei $e_{1:N}$ eine Beobachtung (z.B. Wörter in einem Dokument)
- Sei $x_{1:N}$ eine Sequenz von versteckten Zuständen
- Ein **linear-chain CRF** definiert die bedingte Wahrscheinlichkeitsverteilung:

Linear-chain CRF

- Sei $e_{1:N}$ eine Beobachtung (z.B. Wörter in einem Dokument)
- Sei $x_{1:N}$ eine Sequenz von versteckten Zuständen
- Ein **linear-chain CRF** definiert die bedingte Wahrscheinlichkeitsverteilung:

$$\mathbf{P}(\mathbf{x}_{1:N} \mid \mathbf{e}_{1:N}) = \alpha e^{\left[\sum_{i=1}^{N} F(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i)\right]}$$

Linear-chain CRF

- Sei $e_{1:N}$ eine Beobachtung (z.B. Wörter in einem Dokument)
- Sei $x_{1:N}$ eine Sequenz von versteckten Zuständen
- Ein **linear-chain CRF** definiert die bedingte Wahrscheinlichkeitsverteilung:

$$\mathbf{P}(\mathbf{x}_{1:N} \mid \mathbf{e}_{1:N}) = \alpha e^{\left[\sum_{i=1}^{N} F(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i)\right]}$$

- α : Normalisierungsfaktor

Linear-chain CRF

- Sei $e_{1:N}$ eine Beobachtung (z.B. Wörter in einem Dokument)
- Sei $x_{1:N}$ eine Sequenz von versteckten Zuständen
- Ein **linear-chain CRF** definiert die bedingte Wahrscheinlichkeitsverteilung:

$$\mathbf{P}(\mathbf{x}_{1:N} \mid \mathbf{e}_{1:N}) = \alpha e^{\left[\sum_{i=1}^{N} F(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i)\right]}$$

- α : Normalisierungsfaktor
- F : Feature Funktion definiert als die gewichtete Summe einer Sammlung von k Komponenten Feature Funktionen

$$F(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \sum_k \lambda_k f_k(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i)$$

$$F(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \sum_k \lambda_k f_k(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i)$$

- Die λ_k Parameter Werte werden mit dem MAP Verfahren gelernt (Erinnerung letzte VL)

$$F(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \sum_k \lambda_k f_k(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i)$$

- Die λ_k Parameter Werte werden mit dem MAP Verfahren gelernt (Erinnerung letzte VL)
- Die Feature Funktionen f sind die wichtigsten Elemente eines CRF

$$F(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \sum_k \lambda_k f_k(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i)$$

- Die λ_k Parameter Werte werden mit dem MAP Verfahren gelernt (Erinnerung letzte VL)
- Die Feature Funktionen f sind die wichtigsten Elemente eines CRF
- f_k hat Zugriff auf die benachbarten Zustände $\mathbf{x}_{i-1}$ und $\mathbf{x}_i$, die gesamte Beobachtung $\mathbf{e}$ und die Position in der Sequenz, i

- Die Aufgabe beim Modellieren liegt darin, zu einem bestimmten Problem bestimmte Feature Funktionen zu definieren

- Die Aufgabe beim Modellieren liegt darin, zu einem bestimmten Problem bestimmte Feature Funktionen zu definieren
- Beispiel:

- Die Aufgabe beim Modellieren liegt darin, zu einem bestimmten Problem bestimmte Feature Funktionen zu definieren
- Beispiel:
  - Feature Funktion generiert den Wert 1, wenn das Wort **ANDREW** ist und der aktuelle Zustand **SPEAKER** ist

- Die Aufgabe beim Modellieren liegt darin, zu einem bestimmten Problem bestimmte Feature Funktionen zu definieren
- Beispiel:
  - Feature Funktion generiert den Wert 1, wenn das Wort **ANDREW** ist und der aktuelle Zustand **SPEAKER** ist

$$f_1(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \begin{cases} 1 & \text{if } \mathbf{x}_i = \text{SPEAKER and } \mathbf{e}_i = \text{ANDREW} \\ 0 & \text{otherwise} \end{cases}$$

- Die Wirkung der Features ist abhängig vom dazugehörigen Gewicht

- Die Wirkung der Features ist abhängig vom dazugehörigen Gewicht
- Wenn $\lambda_1 > 0$ und f_1 wahr ist, ist die Wahrscheinlichkeit der versteckten Zustandssequenz $\mathbf{x}_{1:N}$ erhöht, d.h. dass das Modell den Zielzustand **SPEAKER** für das Wort **ANDREW** bevorzugen soll

- Die Wirkung der Features ist abhängig vom dazugehörigen Gewicht
- Wenn $\lambda_1 > 0$ und f_1 wahr ist, ist die Wahrscheinlichkeit der versteckten Zustandssequenz $\mathbf{x}_{1:N}$ erhöht, d.h. dass das Modell den Zielzustand **SPEAKER** für das Wort **ANDREW** bevorzugen soll
- Falls $\lambda_1 < 0$ wird das CRF Modell die Assoziation versuchen zu vermeiden

- Die Wirkung der Features ist abhängig vom dazugehörigen Gewicht
- Wenn $\lambda_1 > 0$ und f_1 wahr ist, ist die Wahrscheinlichkeit der versteckten Zustandssequenz $\mathbf{x}_{1:N}$ erhöht, d.h. dass das Modell den Zielzustand **SPEAKER** für das Wort **ANDREW** bevorzugen soll
- Falls $\lambda_1 < 0$ wird das CRF Modell die Assoziation versuchen zu vermeiden
- Bei $\lambda_1 = 0$ wird das Feature ignoriert

$$f_2(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \begin{cases} 1 & \text{if } \mathbf{x}_i = \text{SPEAKER and } \mathbf{e}_{i+1} = \text{SAID} \\ 0 & \text{otherwise} \end{cases}$$

$$f_2(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \begin{cases} 1 & \text{if } \mathbf{x}_i = \text{SPEAKER and } \mathbf{e}_{i+1} = \text{SAID} \\ 0 & \text{otherwise} \end{cases}$$

- Für einen Satz wie **'Andrew said ...'** hält f_1 und f_2

$$f_2(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \begin{cases} 1 & \text{if } \mathbf{x}_i = \text{SPEAKER and } \mathbf{e}_{i+1} = \text{SAID} \\ 0 & \text{otherwise} \end{cases}$$

- Für einen Satz wie **'Andrew said ...'** hält f_1 und f_2
- In diesem Fall überlappen sich die Features und verstärken die Annahme $\mathbf{x}_1 = \text{SPEAKER}$

$$f_2(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \begin{cases} 1 & \text{if } \mathbf{x}_i = \text{SPEAKER and } \mathbf{e}_{i+1} = \text{SAID} \\ 0 & \text{otherwise} \end{cases}$$

- Für einen Satz wie '**Andrew said ...**' hält f_1 und f_2
- In diesem Fall überlappen sich die Features und verstärken die Annahme $\mathbf{x}_1 = \text{SPEAKER}$
- Aufgrund der Unabhängigkeitsannahme wäre so etwas in HMMs nicht möglich

$$f_2(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \begin{cases} 1 & \text{if } \mathbf{x}_i = \text{SPEAKER and } \mathbf{e}_{i+1} = \text{SAID} \\ 0 & \text{otherwise} \end{cases}$$

- Für einen Satz wie '**Andrew said ...**' hält f_1 und f_2
- In diesem Fall überlappen sich die Features und verstärken die Annahme $\mathbf{x}_1 = \text{SPEAKER}$
- Aufgrund der Unabhängigkeitsannahme wäre so etwas in HMMs nicht möglich
- Features in CRFs können jeden Teil der Sequenz $\mathbf{e}_{1:N}$ nutzen

$$f_2(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \begin{cases} 1 & \text{if } \mathbf{x}_i = \text{SPEAKER and } \mathbf{e}_{i+1} = \text{SAID} \\ 0 & \text{otherwise} \end{cases}$$

- Für einen Satz wie **'Andrew said ...'** hält f_1 und f_2
- In diesem Fall überlappen sich die Features und verstärken die Annahme $\mathbf{x}_1 = \text{SPEAKER}$
- Aufgrund der Unabhängigkeitsannahme wäre so etwas in HMMs nicht möglich
- Features in CRFs können jeden Teil der Sequenz $\mathbf{e}_{1:N}$ nutzen
- Features können ebenfalls bei Transitionen zwischen Zuständen verwendet werden

$$f_2(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{e}, i) = \begin{cases} 1 & \text{if } \mathbf{x}_i = \text{SPEAKER and } \mathbf{e}_{i+1} = \text{SAID} \\ 0 & \text{otherwise} \end{cases}$$

- Für einen Satz wie **'Andrew said ...'** hält f_1 und f_2
- In diesem Fall überlappen sich die Features und verstärken die Annahme $\mathbf{x}_1 = \text{SPEAKER}$
- Aufgrund der Unabhängigkeitsannahme wäre so etwas in HMMs nicht möglich
- Features in CRFs können jeden Teil der Sequenz $\mathbf{e}_{1:N}$ nutzen
- Features können ebenfalls bei Transitionen zwischen Zuständen verwendet werden
- Feature Funktionen müssen nicht binär sein, sondern können beliebige reelle Zahlen annehmen

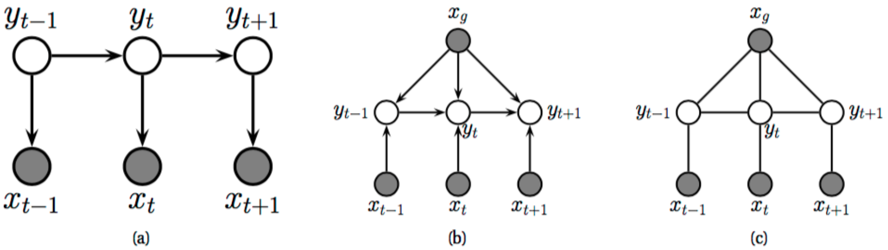


Abbildung: Verschiedene Modelle für sequentielle Daten. (a) HMM (b) MEMM (c) CRF. Quelle: [Blake et al.(2011)Blake, Kohli, and Rother]

Beispiele in der Anwendung

(a) (b) (c) (d) (e)

- Beispiel: Handschrift erkennen
- Buchstaben können sehr ähnlich aussehen, wie Bild (b) und (d) zeigen
- Der Kontext verringert die Fehlerrate

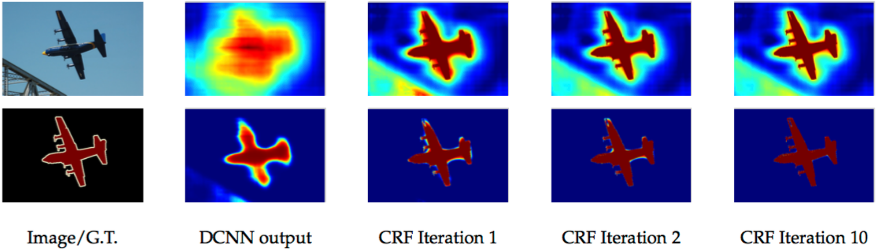


Abbildung: CRF angewendet innerhalb eines CNN Modells als Post-Processing Verfahren. Quelle:
[Chen et al.(2016)Chen, Papandreou, Kokkinos, Murphy, and Yuille]

Zusammenfassung

Zusammenfassung

- CRFs sind MRFs meist vorzuziehen, weil:

Zusammenfassung

- CRFs sind MRFs meist vorzuziehen, weil:
  - Modellieren direkter das Vorhersage-Problem $P(Y \mid X = x)$, wodurch sie genauer sind

Zusammenfassung

- CRFs sind MRFs meist vorzuziehen, weil:
  - Modellieren direkter das Vorhersage-Problem $P(Y \mid X = x)$, wodurch sie genauer sind
  - Aber: MRFs kann mit fehlenden Werten in der Eingabe x umgehen sowie eine Vorhersage von X $(P(X \mid Y = y))$

Zusammenfassung

- CRFs sind MRFs meist vorzuziehen, weil:
  - Modellieren direkter das Vorhersage-Problem $P(Y \mid X = x)$, wodurch sie genauer sind
  - Aber: MRFs kann mit fehlenden Werten in der Eingabe x umgehen sowie eine Vorhersage von X $(P(X \mid Y = y))$
- CRF sind weit verbreitet in der Anwendung von Deep Learning Applikationen

Zusammenfassung

- CRFs sind MRFs meist vorzuziehen, weil:
  - Modellieren direkter das Vorhersage-Problem $P(Y \mid X = x)$, wodurch sie genauer sind
  - Aber: MRFs kann mit fehlenden Werten in der Eingabe x umgehen sowie eine Vorhersage von X $(P(X \mid Y = y))$
- CRF sind weit verbreitet in der Anwendung von Deep Learning Applikationen
- Eignen sich besonders gut als Post-Processing Methode für Bild Segmentierung

Zusammenfassung

- CRFs sind MRFs meist vorzuziehen, weil:
  - Modellieren direkter das Vorhersage-Problem $P(Y \mid X = x)$, wodurch sie genauer sind
  - Aber: MRFs kann mit fehlenden Werten in der Eingabe x umgehen sowie eine Vorhersage von X $(P(X \mid Y = y))$
- CRF sind weit verbreitet in der Anwendung von Deep Learning Applikationen
- Eignen sich besonders gut als Post-Processing Methode für Bild Segmentierung
- CRFs werden ebenfalls end-to-end in RNNs eingesetzt

Zusammenfassung

- CRFs sind MRFs meist vorzuziehen, weil:
  - Modellieren direkter das Vorhersage-Problem $P(Y \mid X = x)$, wodurch sie genauer sind
  - Aber: MRFs kann mit fehlenden Werten in der Eingabe x umgehen sowie eine Vorhersage von X $(P(X \mid Y = y))$
- CRF sind weit verbreitet in der Anwendung von Deep Learning Applikationen
- Eignen sich besonders gut als Post-Processing Methode für Bild Segmentierung
- CRFs werden ebenfalls end-to-end in RNNs eingesetzt
- Es existieren viele Frameworks/Implementierungen wie z.B. CRF++ oder crfsuite für Python um CRFs zu erstellen, trainieren und testen

Vielen Dank für Eure Aufmerksamkeit!

Quellen

Andrew Blake, Pushmeet Kohli, and Carsten Rother.
Markov Random Fields for Vision and Image Processing.
MIT Press, Cambridge, 2011.
ISBN 978-0-262-01577-6.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille.
Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs.
CoRR, abs/1606.00915, 2016.
URL http://arxiv.org/abs/1606.00915.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira.
Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.
ISBN 1-55860-778-1.
URL http://dl.acm.org/citation.cfm?id=645530.655813.

Stuart Jonathan Russell and Peter Norvig.
Artificial Intelligence - A Modern Approach.
Prentice Hall, London, 2010.
ISBN 978-0-136-04259-4.