Informatik IV - Tutorium XII & XIII (SR -120)

Tut Nr. 3 – Optimierung & Neuronale Netze

David Münch

Universität Karlsruhe (TH)
Fakultät für Informatik
IBDS Prautzsch

15. Mai 2008

Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

Organisatorisches

Email: muenchdavid@gmail.com

https://www.stud.uni-karlsruhe.de/~uhbro/

Tutorium 12: Donnerstags 8:00 Uhr - Raum -120
Tutorium 13: Donnerstags 9:45 Uhr - Raum -120

Übungsblattabgabe Donnerstag.

Schein / Übungsblätter

Nur die mit (**K**) gekennzeichneten Aufgaben sind abzugeben.
Diese werden korrigiert und bewertet.
66% der Punkte aller mit (**K**) gekennzeichneten Aufgaben aller Übungsblätter sind notwendig, um einen Schein zu erhalten.

Schein / Übungsblätter

Nur die mit (**K**) gekennzeichneten Aufgaben sind abzugeben.
Diese werden korrigiert und bewertet.
66% der Punkte aller mit (**K**) gekennzeichneten Aufgaben aller Übungsblätter sind notwendig, um einen Schein zu erhalten.

Abgabe meistens Donnerstags.

Schein / Übungsblätter

Nur die mit (**K**) gekennzeichneten Aufgaben sind abzugeben.
Diese werden korrigiert und bewertet.
66% der Punkte aller mit (**K**) gekennzeichneten Aufgaben aller Übungsblätter sind notwendig, um einen Schein zu erhalten.

Abgabe meistens Donnerstags.

Abgabe in Zweiergruppe erlaubt und ausdrücklich erwünscht!

Schein / Übungsblätter

Nur die mit (**K**) gekennzeichneten Aufgaben sind abzugeben. Diese werden korrigiert und bewertet.
66% der Punkte aller mit (**K**) gekennzeichneten Aufgaben aller Übungsblätter sind notwendig, um einen Schein zu erhalten.

Abgabe meistens Donnerstags.

Abgabe in Zweiergruppe erlaubt und ausdrücklich erwünscht!

Um das Übungsteam zu unterstützen bitte folgendes Deckblatt verwenden:
`http://www.stud.uni-karlsruhe.de/~unbdh/deckblatt/index.php?course=5`

Literatur

Boehm, Prautzsch: Numerical Methods. AK Peters 1993. ISBN 3-528-06350-5
http://www.ubka.uni-karlsruhe.de/hylib-bin/suche.cgi?opacdb=UBKA_OPAC&fbt=7319953&nd=3204657

Ash: Information Theory. Dover 1990. ISBN 0-486-66521-6
http://www.ubka.uni-karlsruhe.de/hylib-bin/suche.cgi?opacdb=UBKA_OPAC&nd=9866904

Goos: Vorlesungen über Informatik. Bd. 4, Springer 1998. ISBN 3-540-60650-5
http://www.ubka.uni-karlsruhe.de/hylib-bin/suche.cgi?opacdb=UBKA_OPAC&fbt=9316367&nd=6568301

Was wollen wir heute erreichen?

Was wollen wir heute erreichen?

Was wollen wir heute erreichen?

- Genetische Algorithmen kennen lernen

Was wollen wir heute erreichen?

- Genetische Algorithmen kennen lernen
- Einführung in Künstliche Neuronale Netze

Was wollen wir heute erreichen?

- Genetische Algorithmen kennen lernen
- Einführung in Künstliche Neuronale Netze
- Mit dem einfachen Perzeptron arbeiten können

Definition: Optimierungsproblem

Ein Optimierungsproblem $P = (Q, c, l)$ besteht aus:

Definition: Optimierungsproblem

Ein Optimierungsproblem $P = (Q, c, l)$ besteht aus:

- einem Suchraum Q, in dem die Lösungen zu suchen sind.

Definition: Optimierungsproblem

Ein Optimierungsproblem $P = (Q, c, l)$ besteht aus:

- einem Suchraum Q, in dem die Lösungen zu suchen sind.
- einer Bewertungs- bzw. Kostenfunktion $c : Q \rightarrow \mathbb{R}$

Definition: Optimierungsproblem

Ein Optimierungsproblem $P = (Q, c, l)$ besteht aus:

- einem Suchraum Q, in dem die Lösungen zu suchen sind.
- einer Bewertungs- bzw. Kostenfunktion $c : Q \rightarrow \mathbb{R}$
- einer Straffunktion $l : Q \rightarrow \mathbb{R}$, die mit $l(q) = 0$ eine Lösung charakterisiert.

Definition: Optimierungsproblem

Ein Optimierungsproblem $P = (Q, c, l)$ besteht aus:

- einem Suchraum Q, in dem die Lösungen zu suchen sind.
- einer Bewertungs- bzw. Kostenfunktion $c : Q \rightarrow \mathbb{R}$
- einer Straffunktion $l : Q \rightarrow \mathbb{R}$, die mit $l(q) = 0$ eine Lösung charakterisiert.

Die Bewertungsfunktion kann umformuliert werden zu:
$c'(q) = c(q) + \lambda l(q)$

Generische Algorithmen

Generische Algorithmen

- Binärer Merkmalsvektor k

Generische Algorithmen

- Binärer Merkmalsvektor k
- In jeder Generation erzeugen μ Eltern μ Nachkommen und nur diese überleben

Generische Algorithmen

- Binärer Merkmalsvektor k
- In jeder Generation erzeugen μ Eltern μ Nachkommen und nur diese überleben
- Im Gegensatz zu Evolutionsstrategien paaren sich starke Eltern häufiger (höhere Wahrscheinlichkeit) als schwächere

Fortpflanzung

Jedes Individuum q wird mit Wahrscheinlichkeit $W_q = \frac{fit(q)}{\sum_{q' \in P} fit(q_0)}$ Elter. Die μ Individuen der Population erzeugen μ Nachkommen (Klone), nur diese überleben.

Fortpflanzung

Jedes Individuum q wird mit Wahrscheinlichkeit $W_q = \frac{fit(q)}{\sum_{q' \in P} fit(q_0)}$ Elter. Die μ Individuen der Population erzeugen μ Nachkommen (Klone), nur diese überleben.

Kreuzung

Dann werden unter den μ Nachkommen $p_k \mu$ Individuen ausgesucht, die der Kreuzung unterzogen werden. Dabei ist $p_k \in [0, 1]$ ein beliebig vorgegebener Anteil. Die Kreuzungspaare werden zufällig festgelegt. Es werden ein oder mehrere Kreuzungspunkte j bestimmt.

Fortpflanzung

Jedes Individuum q wird mit Wahrscheinlichkeit $W_q = \frac{fit(q)}{\sum_{q' \in P} fit(q_0)}$ Elter. Die μ Individuen der Population erzeugen μ Nachkommen (Klone), nur diese überleben.

Kreuzung

Dann werden unter den μ Nachkommen $p_k \mu$ Individuen ausgesucht, die der Kreuzung unterzogen werden. Dabei ist $p_k \in [0, 1]$ ein beliebig vorgegebener Anteil. Die Kreuzungspaare werden zufällig festgelegt. Es werden ein oder mehrere Kreuzungspunkte j bestimmt.

Mutation

Mit einer Wahrscheinlichkeit p_m werden die Bits des Merkmalsvektors invertiert (p_m sehr klein).

Beispiel

Population sei $\mu = 4$

Merkmalsvektor $k = (x_1, x_2, x_3, x_4, x_5, x_6)$

Fitnessfunktion sei $fit(q) = x_1 + x_2 - x_3 + x_4 + x_5 - x_6$

Start-Population:

$q_1 = (1,0,0,0,1,0) \Rightarrow fit(q_1) = 2 \Rightarrow W_q 1 = \frac{2}{7}$

$q_2 = (1,1,0,1,0,0) \Rightarrow fit(q_2) = 3 \Rightarrow W_q 2 = \frac{3}{7}$

$q_3 = (1,0,1,1,0,0) \Rightarrow fit(q_3) = 1 \Rightarrow W_q 3 = \frac{1}{7}$

$q_4 = (1,1,0,0,0,1) \Rightarrow fit(q_4) = 1 \Rightarrow W_q 4 = \frac{1}{7}$

Fortpflanzung

Klonen mit der Auswahlwahrscheinlichkeit W_q

$$q_2' = (1,1,0,1,0,0) \Rightarrow fit(q_2') = 3$$
$$q_1' = (1,0,0,0,1,0) \Rightarrow fit(q_1') = 2$$
$$q_2'' = (1,1,0,1,0,0) \Rightarrow fit(q_2'') = 3$$
$$q_4' = (1,1,0,0,0,1) \Rightarrow fit(q_4') = 1$$

Kreuzung

Individuen zufällig auswählen
hier: $q'_2 = (1,1,0,1,0,0)$ und $q'_1 = (1,0,0,0,1,0)$
und Kreuzung durchführen $j = 3$
$q'*_2 = (1,1,0,0,1,0) \Rightarrow fit(q'*_2) = 3$
$q'*_1 = (1,0,0,1,0,0) \Rightarrow fit(q'*_1) = 2$

Mutation

Mit der Wahrscheinlichkeit p_m Bits invertieren:
$q'_4 = (1, 1, 0, 0, 0, 1)$
$q'^{+}_4 = (1, 1, 0, 0, 1, 1) \Rightarrow fit(q'^{+}_4) = 2$

Neue Population

$$q'*_2 = (1,1,0,0,1,0) \Rightarrow fit(q'*_2) = 3$$
$$q'*_1 = (1,0,0,1,0,0) \Rightarrow fit(q'*_1) = 2$$
$$q''_2 = (1,1,0,1,0,0) \Rightarrow fit(q''_2) = 3$$
$$q'^{+}_4 = (1,1,0,0,1,1) \Rightarrow fit(q'^{+}_4) = 2$$

- Art der Binärkodierung von Zahlen beeinflusst das Verhalten genetischer Algorithmen.

- Art der Binärkodierung von Zahlen beeinflusst das Verhalten genetischer Algorithmen.
- Informationen, die über die Fitness entscheiden oder die inhaltlich zusammengehören, sollten möglichst kompakt in der Bitfolge untergebracht werden, damit sie bei Kreuzungen möglichst nicht zerbrochen werden.

- Art der Binärkodierung von Zahlen beeinflusst das Verhalten genetischer Algorithmen.
- Informationen, die über die Fitness entscheiden oder die inhaltlich zusammengehören, sollten möglichst kompakt in der Bitfolge untergebracht werden, damit sie bei Kreuzungen möglichst nicht zerbrochen werden.

Schema-Theorem (Holland 1975)
Merkmale werden öfter reproduziert, wenn sie

- Art der Binärkodierung von Zahlen beeinflusst das Verhalten genetischer Algorithmen.
- Informationen, die über die Fitness entscheiden oder die inhaltlich zusammengehören, sollten möglichst kompakt in der Bitfolge untergebracht werden, damit sie bei Kreuzungen möglichst nicht zerbrochen werden.

Schema-Theorem (Holland 1975)

Merkmale werden öfter reproduziert, wenn sie

- durch weniger Bits dargestellt werden,

- Art der Binärkodierung von Zahlen beeinflusst das Verhalten genetischer Algorithmen.
- Informationen, die über die Fitness entscheiden oder die inhaltlich zusammengehören, sollten möglichst kompakt in der Bitfolge untergebracht werden, damit sie bei Kreuzungen möglichst nicht zerbrochen werden.

Schema-Theorem (Holland 1975)

Merkmale werden öfter reproduziert, wenn sie

- durch weniger Bits dargestellt werden,
- diese Bits eng zusammenstehen

- Art der Binärkodierung von Zahlen beeinflusst das Verhalten genetischer Algorithmen.
- Informationen, die über die Fitness entscheiden oder die inhaltlich zusammengehören, sollten möglichst kompakt in der Bitfolge untergebracht werden, damit sie bei Kreuzungen möglichst nicht zerbrochen werden.

Schema-Theorem (Holland 1975)

Merkmale werden öfter reproduziert, wenn sie

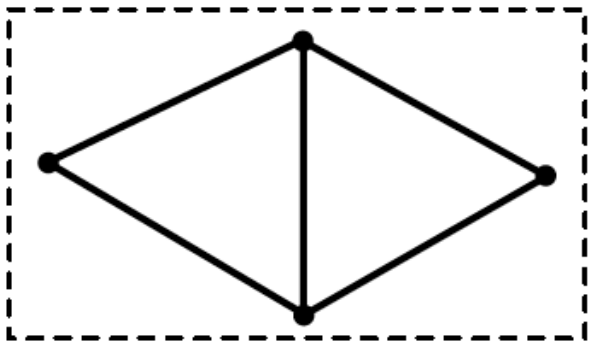
- durch weniger Bits dargestellt werden,
- diese Bits eng zusammenstehen
- und sie hohe Fitness bedeuten.

Aufgabe 18

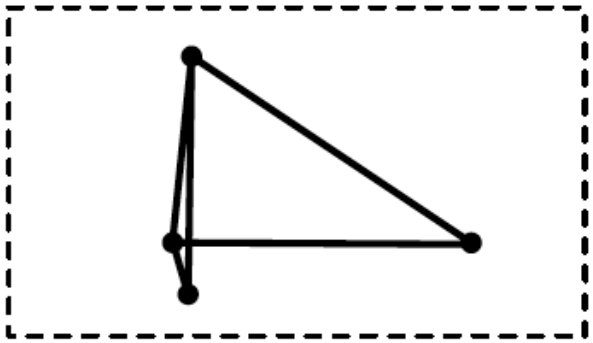
Ein Verfahren zum Zeichnen von Graphen (V, E) ist zu optimieren. Der Algorithmus nimmt die Knoten $V = \{v_1, ..., v_n\}$ und Kanten $E = \{(v_{i_1}, v_{j_1}), ..., (v_{i_m}, v_{j_m})\}$ als Eingabe und soll Positionen im $\mathbb{R}^2$, d. h. auf Bildschirm oder Papier (mit den Grenzen $0 \leq x \leq \sqrt{2}$ und $0 \leq y \leq 1$), also eine Konfiguration

$$K = (x_i, y_i)_{i=1}^{n},$$

für die v_i ausgeben. Kanten (v_{i_k}, v_{j_k}) werden als Strecken zwischen (x_{i_k}, y_{i_k}) und (x_{j_k}, y_{j_k}) gezeichnet.



ansehnlicher Graph



unansehnlicher Graph

1. Definieren Sie eine sinnvolle Bewertungsfunktion $BWF(K)$ für solche Konfigurationen K. Ihre Maximierung soll garantieren, daß ein ansehnlicher Graph entsteht:

2. Definieren Sie eine Straffunktion $PEN(K)$, die verhindern soll, daß Punkte zu nah an Kanten liegen (siehe Abbildung).

3. Definieren Sie zusätzlich eine Straffunktion $PEN(K)$, die verhindern soll, daß Kanten sich in der Zeichnung unnötig überschneiden.[1]

4. Wir wollen Verfahren der genetischen Optimierung auf unser Zeichenprogramm für Graphen (V, E) anwenden. Was sind gute Ideen für die Modellierung?

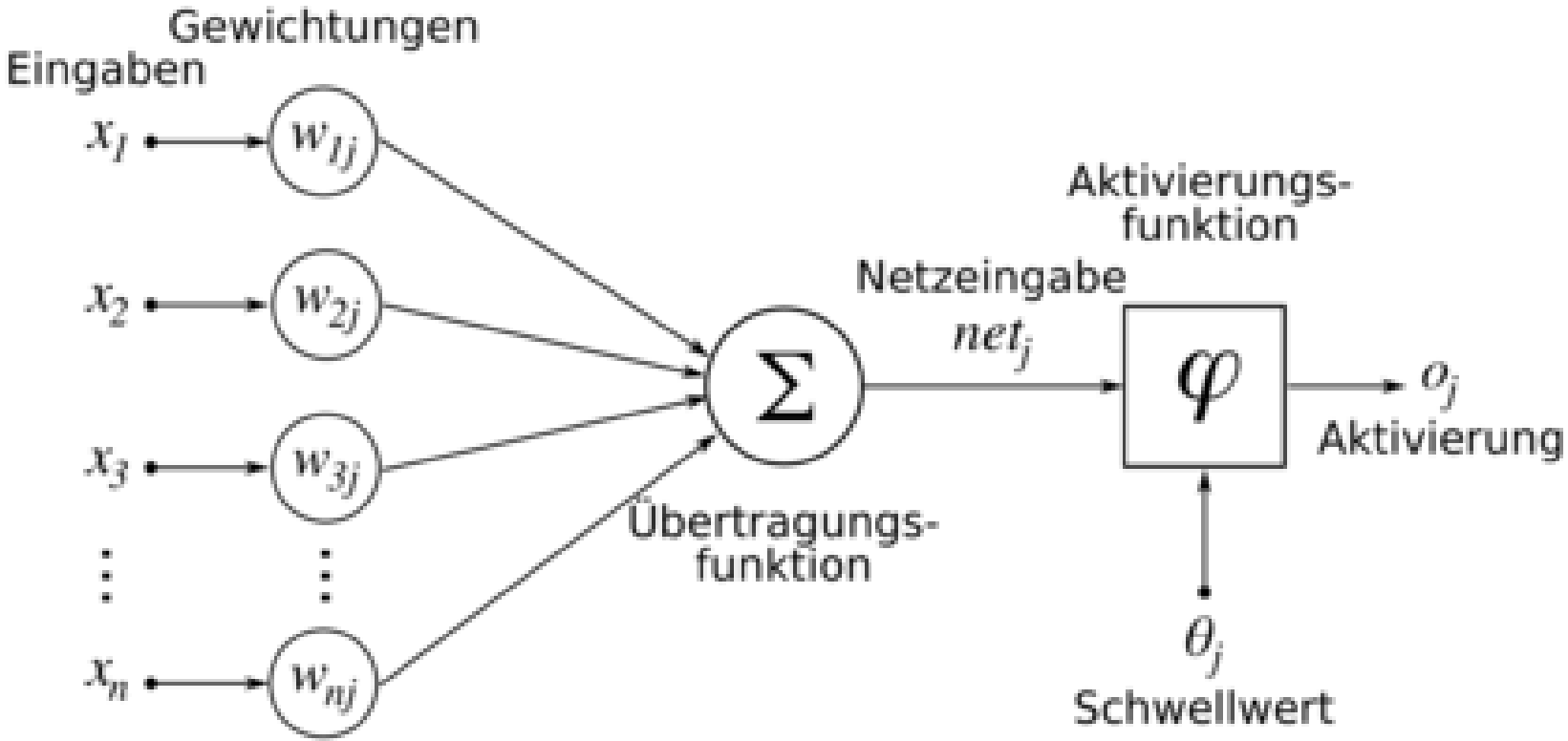
   i) Wie lassen sich sinnvoll Kreuzungen aus zwei Konfigurationen K_1, K_2 bilden?

   ii) Wie kann eine Konfiguration K sinnvoll mutiert werden?

Künstliche Neuronale Netze

Künstliche neuronale Netze basieren meist auf der Vernetzung vieler McCulloch-Pitts-Neuronen oder leichter Abwandlungen davon.
Die Topologie eines Netzes (die Zuordnung von Verbindungen zu Knoten) muss abhängig von seiner Aufgabe gut durchdacht sein.
Nach der Konstruktion eines Netzes folgt die Trainingsphase, in der das Netz lernt.
Seine besonderen Eigenschaften machen das KNN bei allen Anwendungen interessant, bei denen kein bzw. nur geringes explizites (systematisches) Wissen über das zu lösende Problem vorliegt. Dies sind z.B. die Texterkennung, Bilderkennung und Gesichtserkennung, bei denen einige Hunderttausend bis Millionen Bildpunkte in eine im Vergleich dazu geringe Anzahl von erlaubten Ergebnissen überführt werden müssen.

Aufbau von künstlichen Neuronen



Definition: McCulloch-Pitts-Neuronen

1. $N_1, ..., N_n$ seien die Neuronen

Definition: McCulloch-Pitts-Neuronen

1. $N_1, ..., N_n$ seien die Neuronen
2. $w_{ij} \in \mathbb{R}$ das Gewicht der Verbindung $N_i N_j$

Definition: McCulloch-Pitts-Neuronen

1. $N_1, ..., N_n$ seien die Neuronen
2. $w_{ij} \in \mathbb{R}$ das Gewicht der Verbindung $N_i N_j$
3. $\sigma_i \in \mathbb{R}$ ein Schwellwert

Definition: McCulloch-Pitts-Neuronen

1. $N_1, ..., N_n$ seien die Neuronen
2. $w_{ij} \in \mathbb{R}$ das Gewicht der Verbindung $N_i N_j$
3. $\sigma_i \in \mathbb{R}$ ein Schwellwert
4. $q_i(t) \in \{0, 1\}$ Zustand von N_i zum Zeitpunkt $t \in \mathbb{N}$

Definition: McCulloch-Pitts-Neuronen

1. $N_1, ..., N_n$ seien die Neuronen
2. $w_{ij} \in \mathbb{R}$ das Gewicht der Verbindung $N_i N_j$
3. $\sigma_i \in \mathbb{R}$ ein Schwellwert
4. $q_i(t) \in \{0, 1\}$ Zustand von N_i zum Zeitpunkt $t \in \mathbb{N}$
5. $$q_j(t+1) := \begin{cases} 1, & \text{falls } \sum_{i=1}^{n} w_{ij} q_i(t) \geq \sigma_j \\ 0, & \text{sonst.} \end{cases}$$

Das einfache Perzeptron

Das Perzeptron ist die einfachste Art von Neuronalen Netzen.
Jedes Neuron j hat einen Schwellwert σ_j und die Neuronen arbeiten gleichzeitig in diskreten Zeitabständen.
Für die Zustandsmenge $Q = \{0, 1\}$ gilt zu einem Zeitpunkt t:

$$q_j(t+1) := \begin{cases} 1, & \text{falls } \sum_{i=1}^{n} w_{ij} q_i(t) \geq \sigma_j \\ 0, & \text{sonst.} \end{cases}$$

Das einfache Perzeptron hat nur Eingabe- und Ausgabeneuronen, aber keine versteckte Neuronen.

Das einfache Perzeptron

Jedes Ausgabeneuron zerlegt Q^m in 2 Halbräume. Man sagt auch, dass das einfache Perzeptron **linear separierend** ist.

Aufgabe

Betrachten Sie das Perzeptron-Modell für künstliche neuronale Netze.

a) Modellieren Sie folgende Funktionen durch ein Perzeptron mit minimaler Anzahl an Neuronen: AND, OR, XOR.

Aufgabe

Betrachten Sie das Perzeptron-Modell für künstliche neuronale Netze.

a) Modellieren Sie folgende Funktionen durch ein Perzeptron mit minimaler Anzahl an Neuronen: AND, OR, XOR.

b) Lässt sich die XOR Funktion mit einem Neuron realisieren? Beweisen Sie ihre Behauptung.

Quellen

Pajor - Informatik 4 Tutorium SS2007
Prautzsch - Skript Informatik 4 SS2008
Goos, Vorlesungen über Informatik Band 4
Wikipedia

Noch Fragen?

Vorschau

Vorschau

Vorschau

- Lernen von Neuronalen Netzen

Vorschau

- Lernen von Neuronalen Netzen
- Mehrschichtige Neuronale Netze

Vorschau

- Lernen von Neuronalen Netzen
- Mehrschichtige Neuronale Netze
- Erweiterte Konzepte von Neuronalen Netzen

Bis zum nächsten Mal