Neuronale Lernverfahren für Graphen

Brijnesh J. Jain

TU Berlin, Fachbereich Elektrotechnik und Informatik
bjj@cs.tu-berlin.de

Abstract: Am Beispiel des Perzeptrons wird ein neues graphentheoretisches Resultat angewendet: Ein Differentialkalkül für Funktionen auf Graphen.

1 Einleitung

Dieser Beitrag erläutert den Grundgedanken neuronaler Lernverfahren zur Verarbeitung von Graphen am Beispiel des strukturellen Perzeptrons.

Die mathematische Grundlage neuronaler Lernverfahren ist ein neues graphentheoretisches Resultat. Es besagt, dass man einen Raum von Graphen mit genügend Struktur ausstatten kann, so dass sich Teile des mächtigen Differentialkalküls für strukturelle Funktionen auf Graphen übertragen lassen. Aus dem Blickwinkel dieses Resultats sind strukturelle neuronale Lernverfahren eine Anwendung des graphentheoretischen Differentialkalküls. Aus Sicht der künstlichen Intelligenz trägt die Erweiterung neuronaler Verfahren in der Domäne der Graphen zu zwei Forschungsrichtungen bei: (1) zur Integration des symbolischen und subsymbolischen Paradigmas; und (2) zur Kombination von struktureller und statistischer Mustererkennung.

Die Beschränkung auf ein einzelnes Perzeptron ist hilfreich, um die Grundideen und theoretischen Resultate anschaulich darzustellen, ohne dabei den zugrunde liegenden mathematischen Apparat zu bemühen. Neben neuronalen Lernverfahren für Graphen werden weitere potenzielle Anwendungsmöglichkeiten der neuen graphentheoretischen Resultate angedeutet.

1.1 Grundlegende Definitionen und Notationen

Es sei $\mathcal{A}$ eine nichtleere Menge von Knoten und Kantenattributen. Die Menge $\mathcal{A}$ enthalte ein ausgezeichnetes Symbol ε, genannt *void* Symbol. Wir setzen im Folgenden voraus, dass $\mathcal{A}$ ein Euklidischer Raum mit Skalarprodukt $\langle \cdot, \cdot \rangle$ ist. Ein *markierter Graph* ist ein Tripel $X = (V, E, \mu)$, bestehend aus einer endlichen Menge V von Knoten, einer Menge $E \subseteq V \times V$ von Kanten und einer Markierungsfunktion $\mu : V \times V \to \mathcal{A}$ mit $\mu(i, j) = \varepsilon$ genau dann, wenn $i \neq j$ und $(i, j) \notin E$. Für einen Graphen X bezeichne $V(X)$ die Menge der Knoten und $E(X)$ die Menge der Kanten. Ein markierter Graph X kann durch eine *markierte Adjazenzmatrix* $\boldsymbol{X} = (\boldsymbol{x}_{ij})$ beschrieben werden, wobei die Elemente $\boldsymbol{x}_{ij}$ der Matrix den Markierungen $\boldsymbol{x}_{ij} = \mu(i, j)$ entsprechen.

Ein *Morphismus* zwischen Graphen X und Y ist eine partielle Knotenabbildung

$$\phi : V(X) \to V(Y), \quad i \mapsto i^{\phi}.$$

Ein *Monomorphismus* ist ein injektiver Morphismus. Eine *Einbettung* ist ein Monomorphismus auf einer Teilmenge U von $V(X)$, die nicht zu einem Monomorphismus auf einer echt größeren Teilmenge $U' \subseteq V(X)$ erweitert werden kann. Gelegentlich nennen wir auch das Bild $X^{\phi} = \phi(X)$ Einbettung von X in Y entlang ϕ. Mit $\mathcal{E}(X, Y)$ bezeichnen wir die Menge aller Einbettungen von X in Y.

2 Klassifizierungslernen

Neuronale Netze können als eine Erweiterung konventioneller Techniken der statistischen Mustererkennung betrachtet werden. Typische Probleme der statistischen Mustererkennung sind Klassifizierungslernen, Funktionsregression und Clusteranalyse. In der folgenden Darstellung beschränken wir uns auf das Klassifizierungslernen.

Gegeben sei eine *Trainingsmenge*

$$\mathcal{Z} = \{(X_1, y_1) \ldots, (X_k, y_k)\} \subseteq \mathcal{X} \times \{\pm 1\},$$

bestehend aus Mustern $X_i \in \mathcal{X}$ und deren Klassenzugehörigkeiten $y_i \in \{\pm 1\}$. Das Lernproblem besteht darin, auf Grundlage der Trainingsdaten $\mathcal{Z}$ eine Funktion (*Klassifikationsregel*) $f : \mathcal{X} \to \{\pm 1\}$ aus einer vorgegebenen Menge $\mathcal{F}$ von Funktionen zu bestimmen, die bei Beobachtung eines neuen Musters X seine Klassenzugehörigkeit y mittels $f(X)$ *möglichst gut* vorhersagt. Dabei bedeutet *möglichst gut*, dass die Wahrscheinlichkeit einer Fehlklassifikation für beliebige Daten aus $\mathcal{X} \times \{\pm 1\}$ minimiert wird.

Die folgenden Beispiele veranschaulichen das Lernproblem.

- *Medizinische Diagnostik:* Es soll diagnostiziert werden, ob eine Person eine schweren Verletzung überlebt. Verletzte Personen können nach [McG94] durch Vektoren $\boldsymbol{x}^{\mathsf{T}} = (x_1, x_2, x_3)$ mit Merkmalen $x_1 =$ `Revised Trauma Score`, $x_1 =$ `Injury Severity Score` und $x_3 =$ `Alter` repräsentiert werden. Die Ausgangsvariable y kodiert die Klasse `Leben` mit $y = +1$ und die Klasse `Tod` mit $y = -1$.
- *Mutagenitätstest:* Mutagene sind chemische Substanzen, die sich erbgutverändernd auswirken. Da Mutationen von Körperzellen die Ursache von Krebs sein können, dienen Mutagenitätstests zur Detektion von potenziell krebserzeugenden Substanzen. Chemische Substanzen können durch markierte Graphen beschrieben werden, wobei Knoten für Atome und Kanten für Bindungen zwischen Atomen stehen. Markierungen beschreiben Atomart und Bindungstyp. Die Werte $y = +1$ und $y = -1$ kodieren die Klassen `mutagen` bzw. `nicht mutagen`.

Beschränkt man die Klasse $\mathcal{F}$ der Funktionen aus der man die Klassifikationsregel auswählt, so lässt sich das Lernproblem gemäß der statistischen Lerntheorie [Vap95] durch Minimie-

rung des *empirischen Risikos*

$$R_{emp}[f] = \frac{1}{k}\sum_{i=1}^{k}\frac{1}{2}|f(x_i) - y_i| \tag{1}$$

lösen. Es sei bemerkt, dass ohne Beschränkung von $\mathcal{F}$ das Klassifikationsproblem i.a. nicht erfolgreich gelöst werden kann. Ein Beispiel für eine Einschränkung von $\mathcal{F}$ ist die Klasse der linearen Funktionen, die im nächsten Abschnitt betrachtet werden.

3 Perzeptron

1958 erfand Rosenblatt [Ros58] das Perzeptron und ebnete damit den Weg für einen neuen Zugang der Mustererkennung. Mit dem Perzeptron ist außerdem der Grundstein für neuronale Lernverfahren [Hay99] und die Support-Vektor Maschine [SS02] gelegt. Das klassische Perzeptron nach Rosenblatt ist für Klassifikationsprobleme konzipiert, deren Muster durch Merkmalsvektoren dargestellt werden. Für eine ausführliche Darstellung sei auf [DHS01] verwiesen.

Das mathematische Model eines Perzeptrons ist von der Form

$$f : \mathbb{R}^n \to \mathbb{R}, \quad \boldsymbol{x} \mapsto f(\boldsymbol{x}) = \Phi(\boldsymbol{w}^\mathsf{T}\boldsymbol{x} + b),$$

wobei $\boldsymbol{w}$ der *Gewichtsvektor*, b der *Bias* und Φ die *Schwellenfunktion*

$$\Phi(u)\begin{cases}+1 & u \geq 0\\ -1 & u < 0\end{cases} \tag{2}$$

ist. Die gesuchte Klassifikationsregel ist somit eine Funktion f, die gemäß (1) die Anzahl der falsch klassifizierten Trainingsbeispiele minimiert.

3.1 Das Skalarprodukt

Die wichtigen Einsichten in die Funktionsweise des Perzeptrons beruhen auf den geometrischen und algebraischen Eigenschaften des Skalarprodukts.

Geometrisch ist das Skalarprodukt $\boldsymbol{x}^\mathsf{T}\boldsymbol{y}$ eine Zahl, die Informationen über die Längen der Vektoren $\boldsymbol{x}$ und $\boldsymbol{y}$ sowie deren relative Lage zueinander vermöge

$$\boldsymbol{x}^\mathsf{T}\boldsymbol{y} = \|\boldsymbol{x}\|\|\boldsymbol{y}\|\cos\alpha \tag{3}$$

liefert. Hat $\boldsymbol{y}$ die Länge $\|\boldsymbol{y}\| = 1$, so ist $\boldsymbol{x}^\mathsf{T}\boldsymbol{y}$ die Länge der Orthogonalprojektion von $\boldsymbol{x}$ auf $\boldsymbol{y}$. Die geometrische Bedeutung des Skalarprodukt gibt Aufschluss darüber, was ein Perzeptron repräsentiert (siehe Abschnitt 3.2).

Algebraisch handelt es sich beim Skalarprodukt um eine positiv definite, symmetrische Bilinearform [Fis97]. Die algebraischen Eigenschaften werde benötigt, um zu zeigen, dass ein Perzeptron eine Hyperebene zur Trennung der Klassenregionen finden kann, sofern diese existiert (siehe Abschnitt 3.3).

3.2 Geometrische Interpretation

Die geometrische Anschauung des Skalarprodukts führt zu der folgenden geometrischen Interpretation des Perzeptrons.

Die zu $f(x)$ assoziierte lineare Gleichung $g(\boldsymbol{x}) = \boldsymbol{w}^\mathsf{T}\boldsymbol{x} + b$ definiert eine Hyperebene $\mathcal{H}(\boldsymbol{w}, b)$, die den Euklidischen Raum $\mathbb{R}^n$ in zwei Halbräume $\mathcal{R}_+$ und $\mathcal{R}_-$ zerlegt. Die Region $\mathcal{R}_+$ besteht aus allen Punkten für die f positiv ist. Entsprechend besteht die Region $\mathcal{R}_-$ aus allen Punkten für die f negativ ist. Somit repräsentieren $\mathcal{R}_+$ und $\mathcal{R}_-$ die Entscheidungsregionen der Klassen $y = +1$ und $y = -1$.

Der Gewichtsvektor $\boldsymbol{w}$ ist orthogonal zur Hyperebene $\mathcal{H}(\boldsymbol{w}, b)$ und legt somit die Richtung von $\mathcal{H}(\boldsymbol{w}, b)$ eindeutig fest. Durch $b/\|\boldsymbol{w}\|$ ist der Abstand von $\mathcal{H}(\boldsymbol{w}, b)$ zum Ursprung gegeben. Der Wert

$$r(\boldsymbol{x}) = \frac{g(\boldsymbol{x})}{\|w\|}$$

ist der vorzeichenbehaftete geometrische Abstand von $\boldsymbol{x}$ zur Hyperebene $\mathcal{H}(\boldsymbol{w}, b)$. Für alle Punkte $\boldsymbol{x}$ aus Region $\mathcal{R}_-$ ist r negativ. Entsprechend ist r positiv für alle Punkte aus $\mathcal{R}_+$. Der Abstand $r(\boldsymbol{x})$ kann somit als ein Maß dafür betrachtet werden, wie sicher wir uns sind, dass $\boldsymbol{x}$ zur Klasse $y = \Phi(r(\boldsymbol{x}))$ gehört.

3.3 Lernen

Nachdem wir nun wissen, das ein Perzeptron eine Hyperebene repräsentiert, stellt sich nun die Frage, wie man eine Trennhyperebene finden kann, die das empirische Risiko $R_{emp}[f]$ aus Abschnitt 2 minimiert.

Da das empirische Risiko $R_{emp}[f]$ als Treppenfunktion ungeeignet für gradientenbasierte Verfahren ist, approximiert der Perzeptron Algorithmus das Minimum von $R_{emp}[f]$ durch Gradientenabstieg der Fehlerfunktion

$$E(\boldsymbol{w}, b) = \sum_{i=1}^{k} E_i(\boldsymbol{w}, b) \tag{4}$$

wobei E_i die i-te Komponentenfunktion mit $E_i(\boldsymbol{w}, b) = \max\{0, -y \cdot g(\boldsymbol{x})\}$ ist. Für falsch klassifizierte Beispiele $(\boldsymbol{x}_i, y_i)$ ist der Term $-y \cdot g(\boldsymbol{x}) > 0$ und damit $E_i(\boldsymbol{w}, b)$ positiv. Somit ist auch $E(\boldsymbol{w}, b)$ stets nichtnegativ. Geometrisch ist $E(\boldsymbol{w}, b)$ proportional zur Summe der Abstände falsch klassifizierter Beispiele zur Hyperebene $\mathcal{H}(\boldsymbol{w}, b)$.

Der Perzeptron Algorithmus arbeitet inkrementell, d.h. die Trainingsbeispiele werden dem Algorithmus nacheinander präsentiert. Bei Fehlklassifikation werden die Gewichte und der Bias gemäß einer Lernregel angepasst. Die Lernregel (s. *update*-Zeilen in Algorithmus 1) führt bei Fehlklassifizierung des i-ten Beispiels zu einem Gradientenabstieg der i-ten Komponentenfunktion $E_i(\boldsymbol{w}, b)$. Die Lernrate η bestimmt dabei die Schrittweite des Gradientenabstiegs. Algorithmus 1 beschreibt den Perzeptron Algorithmus.

Algorithm 1 (Perceptron Algorithm)

set $\boldsymbol{w} = \mathbf{0}$ and $b = 0$
repeat
  for all $i = 1$ **to** k **do**
    if $\Theta\left(\boldsymbol{w}^\mathsf{T}\boldsymbol{x}_i + b\right) \neq y_i$ **then**
      choose learning rate η
      update $\boldsymbol{w} = \boldsymbol{w} + \eta y_i \boldsymbol{x}_i$
      update $b = b + \eta y_i$
until some criterion is satisfied

Wir nehmen nun an, dass die Trainingsmenge *separierbar* ist, d.h.. es existiert eine Hyperebene, die alle Trainingsbeispiele korrekt trennt. Kann der Perzeptron Algorithmus eine Lösung finden, die alle Beispiele einer separierbaren Trainingsmenge korrekt klassifiziert? Die Antwort auf diese Frage liefert das *Perzeptron Konvergenz Theorem* [Ros62]: Bei konstanter Lernrate $\eta = 1$ findet der Perzeptron Algorithmus nach endlich vielen Iterationen einen Gewichtsvektor $\boldsymbol{w}^*$ und Bias b^*, so dass die Hyperebene $\mathcal{H}(\boldsymbol{w}^*, b^*)$ alle Beispiele einer separierbaren Trainingsmenge korrekt trennt.

4 Das Strukturelle Perzeptron

Der Perzeptron Algorithmus setzt voraus, dass die Eingabemuster Merkmalsvektoren fester Dimension sind. Werden Daten wie beim Mutagenitätstest durch Graphen repräsentiert, so ist der Perzeptron Algorithmus nicht anwendbar. In diesem Abschnitt erweitern wir das Perzeptron dahingehend, dass auch Graphen verarbeitet werden können.

Die Grundidee des strukturellen Perzeptrons besteht darin, den Gewichtsvektor $\boldsymbol{w}$ des klassischen Perzeptrons durch einen Gewichtsgraphen W und das Skalarprodukt $\boldsymbol{w}^\mathsf{T}\boldsymbol{x}$ von Vektoren durch ein verwandtes strukturelles Ähnlichkeitsmaß $W \bullet X$ für Graphen zu ersetzen. In Anlehnung an das klassische Perzeptron lässt sich dann das strukturelle Perzeptron mathematisch durch

$$f(X) = \Theta(W \bullet X + b)$$

beschreiben. Entscheidend für Einsichten in die Funktionsweise des strukturellen Perzeptrons und der Herleitung einer Lernregel ist eine geeignete Wahl des strukturellen Ähnlichkeitsmaßes.

4.1 Das Strukturelle Skalarprodukt

Das Ähnlichkeitsmaß $W \bullet X$ übernimmt beim strukturellen Perzeptron die Funktion des Skalarprodukts $\boldsymbol{w}^\mathsf{T}\boldsymbol{x}$ beim klassischen Perzeptron. Wie soll nun das strukturelle Ähnlichkeitsmaß beschaffen sein, um eine Lernregel zu formulieren aus der sich möglichst scharfe Konvergenzaussagen herleiten lassen? Der hier gewählte Ansatz besteht darin, ein Ähnlichkeitsmaß zu konstruieren, das ähnliche Eigenschaften wie das Skalarprodukt besitzt. Diesem Ansatz liegt die Annahme zugrunde, dass ein dem Skalarprodukt verwandtes struktu-

relles Ähnlichkeitsmaß sich ebenfalls leicht analysieren lässt.

Das gesuchte Ähnlichkeitsmaß lässt sich nicht über die algebraischen Eigenschaften des Skalarprodukts konstruieren, solange unklar ist, wie man eine wohldefinierte Addition von zwei Graphen formuliert. Aus diesem Grund betrachten wir die geometrischen Eigenschaften des Skalarprodukts, um das *strukturelle Skalarprodukt* einzuführen.

Wir beginnen die Darstellung mit einer vorbereitenden Definition. Seien X und Y Graphen und $\phi \in \mathcal{E}(X, Y)$ eine Einbettung. Die Zahl

$$\sigma(\phi, X, Y) = \sum_{i,j} \langle \boldsymbol{x}_{ij}, \boldsymbol{y}_{i^\phi j^\phi} \rangle,$$

misst den *Grad der Übereinstimmung* einer Einbettung X^ϕ mit Y.

Das *strukturelle Skalarprodukt* von X und Y ist durch

$$X \bullet Y = \max_{\phi \in \mathcal{I}(X,Y)} \sigma(\phi, X, Y).$$

definiert. Bei $X \bullet Y$ handelt es sich also um den maximal möglichen Grad der Übereinstimmung von X^ϕ und Y, den man bei beliebiger Wahl von $\phi \in \mathcal{E}(X, Y)$ erhalten kann. Zu bemerken ist, dass die zum strukturellen Skalarprodukt zugehörige Einbettung nicht eindeutig bestimmt ist. Aus diesem Grund zeichnen wir die Einbettungen gesondert aus, die zum maximalen Grad der Übereinstimmung zweier Graphen führen. Wir nennen eine Einbettung ϕ *optimale Einbettung*, wenn $\sigma(\phi, X, Y) \geq \sigma(\psi, X, Y)$ für alle Einbettungen ψ. In diesem Fall gilt $X \bullet Y = \sigma(\phi, X, Y)$.

Wir skizzieren nun die Herleitung der geometrischen Eigenschaften des strukturellen Skalarprodukts. Zunächst zeigt man, dass

$$\|X\| = \sqrt{X \bullet X}$$

die Eigenschaften einer Norm erfüllt. Das ist bemerkenswert, weil $X \bullet Y$ selbst nicht den algebraischen Eigenschaften eines Skalarprodukts genügt. Geometrisch handelt es sich bei der Norm um die *Länge* eines Graphen. Das strukturelle Skalarprodukt zusammen mit der von ihr induzierten Norm erfüllt die strukturelle Variante der *Cauchy-Schwarz Ungleichung*

$$|X \bullet Y| \leq \|X\|^2 \|Y\|^2,$$

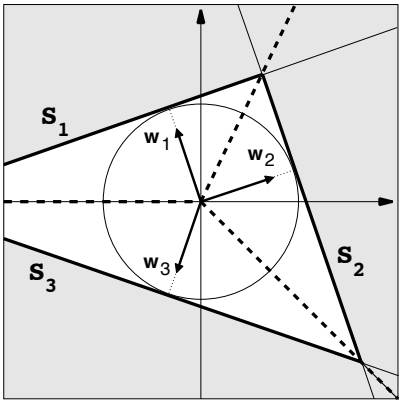
die uns in Analogie zu Gleichung (3) mittels

$$X \bullet Y = \|X\| \|Y\| \cos \alpha$$

die geometrische Interpretation des strukturellen Skalarprodukts liefert.

4.2 Geometrische Interpretation

Für die geometrische Interpretation des strukturellen Perzeptrons nehmen wir an, dass alle hier betrachteten Graphen die gleiche Anzahl von n Knoten besitzen. Für den allgemeinen Fall sei auf die Arbeit [Jai05] verwiesen.



Wir nehmen an, dass der Gewichtsgraph W die drei Vektorrepräsentationen $\boldsymbol{w}_1$, $\boldsymbol{w}_2$ und $\boldsymbol{w}_3$ besitzt. Der fett gezeichnete Polygonzug mit den Segmenten $\mathcal{S}_i = \mathcal{S}(\boldsymbol{w}_i, b)$ stellt die Trennfläche $\mathcal{T}(W, b)$ des strukturellen Perzeptrons dar. Die schattierte Fläche repräsentiert Region $\mathcal{R}_+$, die weiße Fläche $\mathcal{R}_-$. Die Segmente $\mathcal{S}_i$ sind orthogonal zu den Vektoren $\boldsymbol{w}_i$ und tangential zum Kreis mit Mittelpunkt $\mathbf{0}$ und Radius $b/\|W\|$. Die Begrenzungen der Kegel $\mathcal{K}(\boldsymbol{w}_i)$ sind durch die gestrichelten Linien gekennzeichnet.

Abbildung 1: Geometrische Interpretation

Die geometrische Anschauung beruht auf der Darstellung von Graphen als Mengen von Vektoren im Euklidischen Vektorraum $\mathcal{A}^N$, wobei $N = n^2$. Die Vektorrepräsentation $\boldsymbol{x}$ eines Graphen X erhalten wir dadurch, dass wir die Spalten der Adjazenzmatrix $\boldsymbol{X}$ von links nach rechts zu einem N-dimensionalen Vektor mit Elementen aus $\mathcal{A}$ verketten. Permutieren wir die durch die Matrix $\boldsymbol{X}$ festgelegte Reihenfolge der Knoten, so erhalten wir einen zu X strukturell äquivalenten (*isomorphen*) Graphen X'. Die Adjazenzmatrix $\boldsymbol{X'}$ und damit auch die Vektorrepräsentation $\boldsymbol{x'}$ von X' können sich von der Matrix $\boldsymbol{X}$ und zugehörigen Vektor $\boldsymbol{x}$ unterscheiden. Da sowohl X als auch X' das gleiche Objekt repräsentieren, betrachten wir die Menge $\mathcal{V}_X$ aller möglichen Vektorrepräsentationen, die man durch Umnummerierung der Knoten aus X ableiten kann. Die Menge $\mathcal{V}_X$ ist dann die Darstellung von X im Euklidischen Raum.

Die zu $f(X)$ assoziierte Gleichung $g(X) = W \bullet X + b = 0$ definiert eine Trennebene $\mathcal{T}(W, b)$, die sich stückweise aus Hyperebenensegmenten zusammensetzt (s. Abb. 1). Die Anzahl der Hyperebenensegmente entspricht der Anzahl der verschiedenen Vektorrepräsentationen von W. Jedes Element $\boldsymbol{w} \in \mathcal{V}_W$ ist orthogonal zur Hyperebene $\mathcal{H}(\boldsymbol{w}, b)$, wobei $b/\|W\|$ der Abstand von $\mathcal{H}(\boldsymbol{w}, b)$ zum Ursprung ist. Somit liegen sämtliche Hyperebenen, die durch die Elemente aus $\mathcal{V}_W$ definiert werden, tangential zu einer Hyperkugel mit Radius $b/\|W\|$.

Wir bestimmen nun, welche Abschnitte der Hyperebenen $\mathcal{H}(\boldsymbol{w}, b)$ Segmente $\mathcal{S}(\boldsymbol{w}, b)$ der Trennfläche $\mathcal{T}(W, b)$ sind. Die Vektorrepräsentationen $\boldsymbol{w} \in \mathcal{V}_W$ definieren verallgemeinerte Hyperkegel $\mathcal{K}(\boldsymbol{w})$ mit der Eigenschaft, dass jeder Punkt innerhalb $\mathcal{K}(\boldsymbol{w})$ den kleinsten Winkel zu $\boldsymbol{w}$ und größeren Winkel zu den anderen Vektorrepräsentationen hat. Das gesuchte Segment $\mathcal{S}(\boldsymbol{w}, b)$ für eine Hyperebene $\mathcal{H}(\boldsymbol{w}, b)$ erhalten wir durch den Schnitt von $\mathcal{H}(\boldsymbol{w}, b)$ mit dem Kegel $\mathcal{K}(\boldsymbol{w})$.

Dadurch, dass wir Segmente aus den Hyperebenen zur Konstruktion der Trennfläche herausschneiden, ist der Betrag von

$$r(X) = \frac{g(X)}{\|W\|}$$

des vorzeichenbehafteten geometrischen Abstands von X zur nächstgelegenen Hyperebene $\mathcal{H}(W, b)$ eine untere Schranke des Abstands von X zur Trennfläche $\mathcal{T}(W, b)$.

4.3 Lernen

Durch Substitution von $\boldsymbol{w}$ und $\boldsymbol{w}^\mathsf{T}\boldsymbol{x}$ durch W und $W \bullet X$ in Gleichung (4) erhalten wir die strukturelle Fehlerfunktion

$$E(W, b) = \sum_{i=1}^{k} E_i(W, b)$$

mit $E_i(W, b) = \max\{0, -y \cdot g(X)\}$ als i-te Komponentenfunktion.

In Übereinstimmung mit der klassischen Fehlerfunktion (4) gilt: Für falsch klassifizierte Beispiele (X_i, y_i) ist der Term $-y \cdot g(X)$ und damit auch $E_i(W, b)$ positiv. Somit ist auch $E(W, b)$ stets nichtnegativ.

Die geometrische Interpretation der strukturellen Fehlerfunktion stimmt jedoch nicht mehr mit dem klassischen Fall überein: Geometrisch liefert $E(W, b)/\|W\|$ eine obere Schranke der Summe der Abstände falsch klassifizierter Beispiele zur Trennfläche $\mathcal{T}(W, b)$. Es kann aber gezeigt werden, dass der strukturelle Fehler $E(W, b) = 0$ ist genau dann, wenn alle Beispiele der Trainingsmenge korrekt klassifiziert werden. Folglich ist die strukturelle Fehlerfunktion konsistent.

Der Ablauf des strukturellen Perzeptron Algorithmus ist prinzipiell der gleiche wie der des klassischen Perzeptron Algorithmus (siehe Algorithmus 2). Lediglich die Lernregel des strukturellen Perzeptron muss modifiziert werden. Dazu führen wir eine Addition und Skalarmultiplikation von Graphen ein. Beide algebraischen Operationen werden komponentenweise über die Adjazenzmatrizen definiert. Die Addition ist graphentheoretisch unsinnig, doch es kann gezeigt werden, dass Sie bei Anwendung der Lernregel zum gewünschten Resultat führt. Sei nun (X_i, y_i) ein falsch klassifiziertes Beispiel. Die Lernregel bestimmt zuerst eine optimale Einbettung $\phi \in \mathcal{E}(X, Y)$. Anschließend wird die Lernregel lokal auf die gleiche Weise angewendet wie beim klassischen Perzeptron. Dabei übernimmt der Graph $y \cdot X^\phi$ eine ähnliche Rolle wie der Gradient $y \cdot \boldsymbol{x}_i$ im klassischen Fall.

Algorithm 2 (Structural Perceptron Algorithm)

set $W = 0$ and $b = 0$
repeat
  for all $i = 1$ **to** k **do**
    if $\Theta(\boldsymbol{w}^\mathsf{T}\boldsymbol{x}_i + b) \neq y_i$ **then**
      choose learning rate η
      determine optimal embedding $\phi \in \mathcal{E}(X; Y)$
      update $W = W + \eta y_i X_i^\phi$
      update $b = b + \eta y_i$
until some criterion is satisfied

Wir betrachten wieder eine separierbare Trainingsmenge und interessieren uns dafür, ob Algorithmus 2 eine Trennfläche finden kann, die alle Beispiele der Trainingsmenge korrekt separiert. Da das strukturelle Skalarprodukt nicht bilinear ist, können die Beweise für das Perzeptron Konvergenz Theorem nicht übertragen werden. Man kann jedoch über einen anderen Weg zu schwächeren Konvergenzaussagen kommen. Wir unterscheiden hierbei zwei Fälle:

1. Konstante Lernrate η: Bei konstanter Lernrate kann auch nach unendlich vielen Iterationen keine Konvergenz zu einer separierbaren Trennfläche garantiert werden. Man kann aber durch Wahl einer beliebig kleinen Lernrate den strukturellen Fehler $E(W, b)$ beliebig verringern.
2. Abnehmende Lernrate η: Algorithmus 2 garantiert nach unendlich vielen Iterationen Konvergenz zu einer Lösung, die alle Trainingsbeispiele korrekt klassifiziert, wenn die Lernrate nach dem so genannten Prinzip der stochastischen Approximation schrittweise verringert wird.

Vom praktischen Standpunkt handelt es sich bei den Konvergenzbeweisen scheinbar um akademische Resultate. Die theoretische Bedeutung der Konvergenzbeweise liegt darin, dass Algorithmus 2 im separierbaren Fall nicht ein unlösbares Problem zu lösen versucht. In Experimenten konnte gezeigt werden, dass der strukturelle Perzeptron Algorithmus durchaus in der Lage ist, separierbare Probleme bereits nach wenigen Iterationen zu lösen.

5 Strukturelle Differentialrechnung und Anwendungen

Die Lernregel beim strukturellen Perzeptron verwendet das zentrale Resultat der Arbeit [Jai05]: *Der Raum markierter Graphen kann mit genügend Struktur ausgestattet werden, so dass sich Teile des Differentialkalküls auf strukturelle Funktionen übertragen*. Insbesondere kann gezeigt werden, dass der Gradient einer glatten Funktion auf Graphen ein wohldefinierter Graph ist, der in Richtung des steilsten Anstiegs zeigt. Ausgangspunkt der Theorie ist dabei das strukturelle Skalarprodukt.

Das Differentialkalkül fließt nicht nur in den Konvergenzbeweisen des strukturellen Perzeptron Algorithmus ein. Es können nun komplexere neuronalen Architekturen für Graphen konstruiert werden, wie beispielsweise mehrlagige neuronale Netze für überwachte oder Kohonen Netze für unüberwachte Lernprobleme. Denn alle neuronalen Lernverfahren haben eines gemeinsam, sie minimieren eine Fehlerfunktion als Funktion der Gewichte. Durch das Differentialkalkül in der Domäne der Graphen besitzen wir nun ein mathematisches Werkzeug, strukturelle Fehlerfunktionen mit kontinuierlichen Methoden zu lösen.

Anwendungen der graphentheoretischen Resultate beschränken sich nicht nur auf die Konstruktion von neuronalen Lernverfahren für Graphen. Durch das strukturelle Skalarprodukt und das darauf aufbauende strukturelle Differentialkalkül kann nun für Probleme der folgenden Bauart ein allgemeine mathematische Grundlage gegeben werden: Gesucht ist eine Struktur aus einer Menge von Strukturen, die eine bestimmte Eigenschaft P erfüllt. Der kreative Prozess bei der Lösung besteht darin, das Problem auf geeignete Weise als diskretes Optimierungsproblem zu formulieren und dann mittels Relaxation in ein kontinuierli-

ches Problem zu transformieren. Ein einfaches Beispiel dafür ist die Bestimmung eines Mittelwertes einer Menge von Graphen, deren praktische Bestimmung zuvor unklar war. Dieses Beispiel ist deswegen interessant, weil es die praktische Lösung mit einer theoretischen Weiterentwicklung verbindet. Zwar ist der Mittelwert einer Menge von Graphen nicht eindeutig definiert. Jedoch lässt sich aufbauend auf den neuen graphentheoretischen Resultaten die Hypothese formulieren, dass mit einer immer größer werdenden Stichprobe der Mittelwert von Graphen gegen den eindeutigen Erwartungswert einer unbekannten Verteilung konvergiert. Diese Erkenntnisse finden ihre Anwendung beispielsweise in k-means clustering.

6 Schluss

Durch das strukturelle Skalarprodukt und das Differentialkalkül für strukturelle Funktionen wird in dieser Arbeit ein mathematisches Werkzeug bereitgestellt, um ungelöste diskrete Probleme neu zu formulieren und mit kontinuierlichen Methoden zu lösen. Die Konstruktion und mathematische Analyse neuronaler Lernverfahren für Graphen ist dabei nur eine von mehreren möglichen Anwendungen der hier vorgestellten graphentheoretischen Resultate und wurde ausführlich in der Arbeit [Jai05] behandelt.

Danksagung: Ich bedanke mich bei F. Wysotzki, S. Wrobel und O. Hellwich.

Literatur

[DHS01] R.O. Duda, P.E. Hart und D.G. Storck. *Pattern Classification*. J. Wiley & Sons, 2001.

[Fis97] G. Fischer. *Lineare Algebra*. Vieweg, 1997.

[Hay99] S. Haykin. *Neural Networks*. Prentice Hall, Inc., 2nd. Auflage, 1999.

[Jai05] B.J. Jain. *Structural Neural Learning Machines*. Dissertation, TU Berlin, 2005.

[McG94] M. McGonigal. A New Technique for Survival Prediction in Trauma Care Using a Neural Network. In *Proc. World Conference on Neural Networks*, Seiten pp. 3495–3498, 1994.

[Ros58] F. Rosenblatt. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[Ros62] F. Rosenblatt. *Principles of Neurodynamics*. Spartan Book, 1962.

[SS02] A. Smola und B. Schölkopf. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, 2002.

[Vap95] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

Brijnesh Johannes Jain studierte Mathematik und Informatik in Göttingen und Berlin. Anschließend arbeitete er als Wissenschaftlicher Mitarbeiter im Fachbereich Künstliche Intelligenz der TU Berlin und erstellte dort seine Dissertation.