Robustes Boosting durch konvexe Optimierung

Gunnar Rätsch

Research School for Information Sciences and Engineering
Australian National University, ACT 0200 Canberra, Australia
email: Gunnar.Raetsch@anu.edu.au
www: http://mlg.anu.edu.au/~raetsch

Dieser Beitrag stellt einige Grundgedanken des *Maschinellen Lernens* dar und faßt die wesentlichen Ergebnisse meiner Dissertation (Rät01) zusammen.

Lernmaschinen extrahieren Informationen aus einer gegebenen Menge von Trainingsbeispielen, so dass sie in der Lage sind, Eigenschaften von bisher ungesehenen Beispielen – z.B. eine Klassenzugehörigkeit – vorherzusagen. Wir betrachten den Fall, bei dem die resultierende *Klassifikations- oder Regressionsregel* aus einfachen Regeln – den *Basishypothesen* – zusammengesetzt ist. Die sogenannten *Boosting-Algorithmen* erzeugen iterativ eine gewichtete Summe von Basishypothesen, die gut auf ungesehenen Beispielen vorhersagen. In der Dissertation wurden folgende Sachverhalte behandelt:

- *Die zur Analyse von Boosting-Methoden geeignete Statistische Lerntheorie*
  Wir studieren lerntheoretische Garantien zur Abschätzung der Vorhersagequalität auf ungesehenen Beispielen. Als ein praktisches Ergebnis dieser Theorie haben sich sogenannte *Klassifikationstechniken mit großem Margin* herausgestellt – insbesondere Boosting und Support-Vektor-Maschinen (SVM). Ein großer Margin impliziert eine hohe Vorhersagequalität der Entscheidungsregel. Deshalb wird die Größe des Margins beim Boosting analysiert und ein verbesserter Algorithmus vorgeschlagen, der effizient Regeln mit *größtem* Margin erzeugt.

- *Etablierung des Zusammenhangs von Boosting und konvexer Optimierung.*
  Um die Eigenschaften der entstehenden Klassifikations- oder Regressionsregeln zu analysieren, ist es sehr wichtig zu verstehen, ob und unter welchen Bedingungen iterative Algorithmen wie Boosting konvergieren. Wir zeigen, dass solche Algorithmen benutzt werden können, um sehr große Optimierungsprobleme mit Nebenbedingungen zu lösen, deren Lösung sich gut charakterisieren läßt. Dazu werden Verbindungen zum Wissenschaftsgebiet der konvexen Optimierung aufgezeigt und diese ausgenutzt, um Konvergenzgarantien für eine Familie von Boosting-Algorithmen zu geben.

- *Kann man Boosting robust gegenüber Meßfehlern und Ausreißern machen?*
  Ein Problem bisheriger Boosting-Methoden ist die relativ hohe Sensitivität gegenüber Meßungenauigkeiten und Meßfehlern in der Trainingsdatenmenge. Um dieses Problem zu beheben, wird die sogenannte „Soft-Margin“ Idee, die beim Support-Vektor-Lernen schon benutzt wird, auf Boosting übertragen. Das führt zu theoretisch gut motivierten Algorithmen, die ein hohes Maß an Robustheit aufweisen.

- *Wie kann man die Anwendbarkeit von Boosting auf Regressionsprobleme erweitern?*
  Boosting-Methoden wurden ursprünglich für Klassifikationsprobleme entwickelt. Um die Anwendbarkeit auf Regressionsprobleme zu erweitern, werden die vorherigen Konvergenzresultate benutzt und neue Boosting-ähnliche Algorithmen zur Regression entwickelt.

- *Ist Boosting praktisch anwendbar?*
  Für die neu entwickelten Algorithmen wird gezeigt, dass sie in der Praxis tatsächlich gut funktionieren und direkt einsetzbar sind. Die Relevanz wird durch eine industrielle Anwendung in einem Stromverbrauch-Überwachungssystem illustriert.

In diesem Beitrag habe ich mich auf die Darstellung der wichtigsten Ergebnisse meiner Dissertation beschränkt. Es wird nur auf einige Verbindungen zur konvexen Optimierung eingegangen (Abschnitt 2), dann erklärt wie man Boosting robust machen kann (Abschnitt 3) und nur auf eine Anwendung in der Energietechnik eingegangen (Abschnitt 4). Abb. 1 zeigt die Zusammenhänge zwischen diesen Teilen. Die Kürze des Beitrags verlangte eine mathematisch weniger rigorose Darstellung der Sachverhalte; die Details finden sich in der Dissertation.

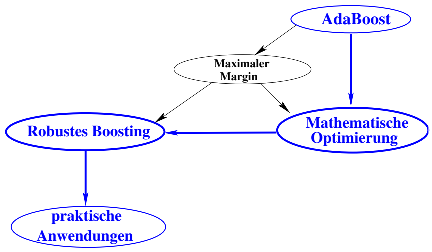

Abbildung 1: Themenüberblick

1 Grundgedanken der Statistischen Lerntheorie und Boosting

Lernen aus Beispielen Die *Statistische Lerntheorie* beschäftigt sich mit dem Lernen aus empirischen Daten, die durch eine Menge von Beobachtungen, den Trainingsbeispielen, gegeben sind:

$$(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_n) \in \mathbb{R}^d \times Y.$$

Die erste Größe, $\boldsymbol{x}$, beschreibt ein Objekt, die zweite, y, ist die für das Lernproblem relevante Beobachtung über das Objekt.

Ziel des Lernens ist es, eine Regel zu erzeugen, mit der für bisher ungesehene Beobachtungen $\boldsymbol{x}^*$ eine gute Prognose y^* abgegeben werden kann. Als Vereinfachung wird angenommen, dass Trainingsbeispiele und ungesehene Beispiele jeweils unabhängig, aber von ein und demselben statistischen Prozess erzeugt werden, und dass es einen (statistischen) Zusammenhang zwischen den beiden Größen $\boldsymbol{x}$ und y gibt. Anderenfalls wären Vorhersagen unmöglich. Damit kann das Ziel des Lernens wie folgt formalisiert werden: Gesucht wird eine Funktion $f : \mathbb{R}^d \to Y$, die die „Kosten“ oder das *Risiko*

$$R[f] = \int l(f(\boldsymbol{x}), y)\, dP(\boldsymbol{x}, y)$$

der Vorhersage für ungesehene Beispiele minimiert. Dabei ist $P(\boldsymbol{x}, y)$ die Wahrscheinlichkeitsverteilung, die dem statistischen Prozess zugeordnet ist, und $l(\cdot, \cdot)$ ist eine Kostenfunktion, die definiert, wie „teuer“ falsche Vorhersagen sind.

In den oben definierten Rahmen lassen sich viele relevante Probleme einordnen: so z.B. das Problem der Erkennung einer bestimmten Krankheit einer Gruppe von Patienten. Der Patient (das o.g. Objekt) ist für das Lernproblem durch seine medizinischen Merkmale gegeben und es ist vorherzusagen, ob der Patient an einer Krankheit leiden wird ($y = +1$), oder nicht ($y = -1$, d.h. $Y = \{-1, +1\}$). In diesem speziellen Fall könnte durch die Kostenfunktion beispielsweise ausgedrückt werden, wie wichtig es ist, die Krankheit früh zu erkennen im Vergleich dazu wie schwerwiegend falsch-positive Ergebnisse sind.

Komplexität der Funktionenklasse Wie kann jedoch die Funktion f gefunden werden, die das Risiko minimiert? Die dem statischen Prozess zugrunde liegende Wahrscheinlichkeitsverteilung P ist im Allgemeinen nicht bekannt, sondern nur die von ihr erzeugten Trainingsbeispiele. Das erschwert die Aufgabe erheblich und macht sie zu einem interessanten Forschungsgegenstand.

Eines der wichtigsten Ergebnisse der statistischen Lerntheorie ist, dass es zwar notwendig, aber bei weitem nicht hinreichend ist, ein kleines *empirisches Risiko*

$$R_{\mathrm{emp}}[f, S] = \frac{1}{N} \sum_{n=1}^{N} l(f(\boldsymbol{x}_n), y_n)$$

zu erreichen. Für die Generalisierungsfähigkeit der erlernten Regel f ist es entscheidend, die Komplexität der Klasse der Funktionen F, aus der f ausgewählt wird, zu kontrollieren. Das zeigt sich z.B. in oberen Schranken für das Risiko, die vom empirischen Fehler und einem Komplexitätsmaß $\Omega[\mathsf{F}]$ abhängen und mit hoher Wahrscheinlichkeit gelten:

$$R[f] \leq R_{\mathrm{emp}}[f, S] + G(\Omega[\mathsf{F}], N), \tag{1}$$

wobei G vom hier gewählten Komplexitätsmaß und anderen Bedingungen abhängt. Der zweite Summand hängt sowohl von der Komplexität der Funktionenklasse $\Omega[\mathsf{F}]$ als auch von der Anzahl der Beispiele N ab: je mehr Beobachtungen verfügbar sind, desto komplexere Funktionen können zuverlässig gelernt werden. Lernalgorithmen minimieren oft die rechte Seite von (1), um eine Funktion mit kleinem Risiko zu erhalten.

Lernen mit Boosting Boosting ist eine Möglichkeit, Funktionen zur *Klassifikation mit zwei Klassen* durch Kombinieren einfacher Funktionen zu erzeugen, die einerseits konsistent mit den Trainingsdaten, aber auch möglichst einfach sind. Im bekanntesten Boosting-Algorithmus – *AdaBoost* – wird ein Basisalgorithmus benutzt, der Funktionen h aus einer Basishypothesenklasse H auswählt. Das könnten beispielsweise Entscheidungsbäume sein. Durch mehrmaliges Ausführen des Basisalgorithmus erhält man mehrere Funktionen $h_1, h_2, \ldots, h_T \in \mathsf{H}$, die dann linear kombiniert werden:

$$f(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x})$$

und das Vorzeichen von f wird zur Klassifikation benutzt. Es wird eine Gewichtung $\mathbf{d} = (d_1, \ldots, d_N)^\top$ auf den Trainingsbeispielen eingeführt, die eine Wahrscheinlichkeit festlegt mit der jedes Beispiel in der Trainingsmenge auftritt. In der ersten Iteration sind alle Gewichte gleich. Dann wird der Basisalgorithmus aufgerufen und der Boosting-Algorithmus erhöht die Gewichte der Beispiele, die von der generierten Basishypothese

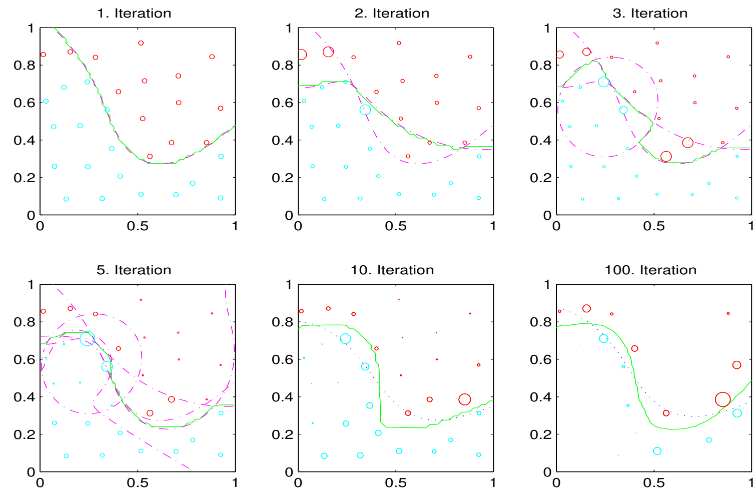

Abbildung 2: AdaBoost auf einem 2-dimensionalen Datensatz: Der Durchmesser der Punkte ist proportional zu den von AdaBoost verwendeten Gewichte während der ersten, zweiten, dritten, fünften, zehnten und hundertsten Iteration. Die Farbe gibt an, zu welcher Klasse der Punkt gehört. Die Strichpunkt-Linie zeigt die Entscheidungsgrenze der vom Basisalgorithmus generierten Hypothesen Die durchgezogene Linie zeigt die Entscheidungsgrenze der kombinierten Funktion. In den letzten beiden Graphen wurde zum Vergleich auch die Entscheidungsgrenze des verwandten Bagging Algorithmus (Bre96) angegeben (gepunktet). (Abb. entnommen aus ROM01).

falsch klassifiziert werden. Dieses Vorgehen wird wiederholt und es kommt zu einer Konzentration der Gewichte auf die besonders schwierig zu klassifizierenden Beispiele – die für das Lernproblem besonders wichtigen Trainingsbeispiele (siehe Abb. 2).

Für diesen Algorithmus wurde gezeigt (FS97), dass der empirische Fehler sehr schnell gegen Null konvergiert, wenn der Basislernalgorithmus in der Lage ist, Hypothesen zurückzuliefern, die etwas besser als zufällige Vorhersagen sind, d.h. der Fehler ist kleiner als $\frac{1}{2} - \frac{1}{2}\gamma$ für ein $\gamma > 0$. Dazu werden nur $\frac{2\log(N)}{\gamma^2}$ Iterationen (= Anzahl der Aufrufe des Basisalgorithmus) benötigt.

In (SFBL98) wurde eine Erklärung für Boosting-Methoden gefunden, die auf der Klassifikation mit *großem Margin* basiert. Dieses neue Ergebnis hat die folgende, vereinfachte Form: Angenommen, der empirische Fehler von f ist Null und es sei

$$\rho = \min_{n=1,\dots,N} y_n f(\boldsymbol{x}_n) \tag{2}$$

der sogenannte *Margin* von f, dann kann das Risiko von f mit hoher Wahrscheinlichkeit wie folgt abgeschätzt werden:

$$R[f] \leq \frac{1}{\rho} G(\Omega[\mathsf{H}], N), \tag{3}$$

wobei $\Omega[\mathsf{H}]$ ein Komplexitätsmaß der Basishypothesenklasse ist. Das zeigt, dass für die Generalisierungsfähigkeit der Margin ρ eine sehr wichtige Rolle spielt: je größer der Margin, desto kleiner wird die obere Schranke (3) an das Risiko. Wichtig ist, dass die Dimensionalität des Raumes der Beispiele nicht in die lerntheoretische Garantie (3) eingeht und damit beliebig groß sein kann.

2 Zusammenhänge zwischen Boosting und konvexer Optimierung

Der wichtigste theoretische Beitrag meiner Dissertation ist die Etablierung der Verbindung von Boosting-Methoden und Techniken der mathematischen Optimierung (siehe Abb. 1). Erst aus diesem Verständnis heraus konnten Boosting-Methoden in vollem Maße – insbesondere im Zusammenhang mit der Klassifikation mit großem Margin – verstanden werden. Damit können nun gezielt Veränderungen am Algorithmus vorgenommen werden, die sich dann theoretisch erklären lassen.

Merkmalsräume und große Margins Der Margin in (2) kann geometrisch verstanden werden: Dazu wird eine Abbildung $\Phi(\boldsymbol{x}) := (h_1(\boldsymbol{x}), \ldots, h_J(\boldsymbol{x}))^\top$ definiert, die von der Basishypothesenklasse $\mathsf{H} = \{h_j \mid j = 1, \ldots, J\}$ abhängt, mit der die Trainingsbeispiele in einen J-dimensionalen sogenannten Merkmalsraum $\mathbb{F}$ abgebildet werden. In $\mathbb{F}$ werden Hyperebenen mit (ℓ_1-normalisiertem) Normalenvektor $\boldsymbol{\alpha}$ betrachtet, die die Trainingspunkte linear trennen: $(\Phi(\boldsymbol{x}) \cdot \boldsymbol{\alpha})$ ist größer als 0, wenn die zugehörige Klasse $+1$ ist, anderenfalls kleiner als 0 (Abb. 3). In $\mathbb{F}$ können die vorzeichenbehafteten Abstände der Trainingsbeispiele zur Hyperebene durch $y_n f_{\boldsymbol{\alpha}}(\boldsymbol{x}_n) = y_n(\Phi(\boldsymbol{x}) \cdot \boldsymbol{\alpha})$ berechnet werden (RMSM02). Es stellt sich heraus, dass der oben definierte Margin ρ der minimale ℓ_∞-Abstand der in den Merkmalsraum abgebildeten Trainingsbeispiele zur Entscheidungebene ist (Abb. 3).

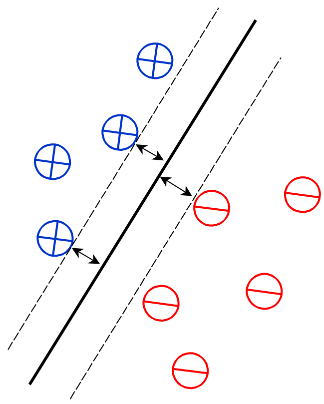

Abbildung 3: Lineare Trennung (durchgezogene Linie) zwischen positiven und negativen Beispielen: Die gestrichelten Linien markieren den Bereich des Margins.

Bestimmung der Ebene mit größtem Margin Um die Schranke für das Risiko in (3) zu minimieren, ist es notwendig den Margin ρ zu maximieren. In (Rät01; RW02) haben wir gezeigt, dass bisherige Boosting-Verfahren den Margin nicht maximieren – sie erreichen oft nur die Hälfte des größtmöglichen Margins. Wir haben deshalb eine Verbesserung des AdaBoost-Algorithmus vorgeschlagen, die den Margin effizient maximieren kann. Es werden nur $\mathcal{O}\left(\frac{\log(N)}{\epsilon^2}\log\left(\frac{1}{\epsilon}\right)\right)$ Iterationen dazu benötigt, wobei ϵ die erlaubte Abweichung vom größten Margin ist. *Das heißt, dass die Komplexität des Algorithmus zur Margin-Maximierung unabhängig von der Dimensionalität des Merkmalsraums ist, der u.U. auch ∞-dimensional sein kann.* Um das zu erreichen, wurde ein Parameter in AdaBoost eingeführt, der kontrolliert, welche Größe des Margins „angestrebt" wird. Falls diese Zielgröße erreichbar ist, dann liefert der modifizierte Algorithmus eine kombinierte Hypothese mit diesem Margin zurück und benötigt dazu nur $\mathcal{O}\left(\frac{\log(N)}{\epsilon^2}\right)$ Iterationen. Da die Größe des Margins vorher unbekannt ist, muß der Parameter durch eine binäre Suche gefunden werden. Das führt zum Faktor $\log\left(\frac{1}{\epsilon}\right)$ in der o.g. Rechenkomplexitätsabschätzung.

Barriere-Optimierung und Konvergenz Die Bestimmung der Hyperebene im Merkmalsraum mit größtem Margin kann durch Lösen eines linearen Optimierungsproblems (LP) erfolgen. Der oben beschriebene Algorithmus stellt somit eine neue Möglichkeit dar, solche LPs effizient zu lösen.

Dies wurde von uns im Zusammenhang mit den sogenannten *Barriere-Optimierungstechniken* untersucht, die Optimierungsprobleme mit Nebenbedingungen auf eine Sequenz von Problemen ohne Nebenbedingungen reduzieren. Wir haben als erste erkannt (RWM$^+$00), dass AdaBoost eine spezielle Implementation eines Barriere-Algorithmus ist, der die Exponentialfunktion zur Berücksichtigung der Nebenbedingungen nutzt. Um dieses Ergebnis verwerten zu können, mussten wir jedoch weitere Konvergenzeigenschaften Boosting-ähnlicher Methoden finden. *Es erwies sich als wichtig, Zusammenhänge zu koordinatenweisen Abstiegsverfahren zu erkennen (RMW02) und vorhandene Ergebnisse für den Fall von Boosting anzuwenden und weiterzuentwickeln.* Wir konnten zeigen, dass solche Algorithmen Optimierungsprobleme der folgenden Art effizient lösen: *Finde die Linearkombination $f_{\boldsymbol{\alpha}}$ von Basishypothesen aus* H*, die*

$$\underbrace{\sum_{n=1}^{N} l(f_{\boldsymbol{\alpha}}(\boldsymbol{x}_n), y_n)}_{\text{empirischer Fehler}} + \underbrace{C\|\boldsymbol{\alpha}\|_1}_{\text{Regularisierung}}, \tag{4}$$

minimiert. Dabei ist $l(\cdot,\cdot)$ eine strikt konvexe Kostenfunktion und die Norm der Kombinationskoeffizienten wird bestraft, um die Komplexität zu kontrollieren. Für solche Algorithmen konnten wir *lineare Konvergenz* zeigen, wenn der Basisalgorithmus gewissen, relativ leicht zu erfüllenden Anforderungen entspricht.

Bedeutung Mit diesem Verständnis sind wir nun in der Lage, fast beliebige konvexe Kostenfunktionen mit Boosting-Techniken zu minimieren. Dies stellt die Grundlage für das Design und die Analyse der im nächsten Schritt entwickelten *robusten Boosting* Algorithmen zur Klassifikation dar. Diesen Ansatz haben wir später auf den Fall der Regression erweitert (RDB02), bei dem es oft unendlich viele Basishypothesen gibt, und halbunendliche Optimierungprobleme betrachtet werden müssen. Zudem war es nun möglich, den Zusammenhang zwischen Boosting und den sogenannten Support-Vektor-Maschinen[1] besser zu verstehen und für die Entwicklung neuer Algorithmen ausnutzen (RMSM02).

3 Robustes Boosting

In der praktischen Anwendung von Lernalgorithmen ist es sehr wichtig, dass sie robust gegenüber Rauschen (Ausreißer, Meßfehler, etc.) sind. Eine zweite wichtige Innovation meiner Dissertation ist die Einführung des sogenannten *Soft-Margins*, um robustere Algorithmen zu erhalten (ROM01). Dieses Konzept wurde zwar schon vorher für die SVMs benutzt (CV95), konnte jedoch von uns in die „Boosting-Welt“ übertragen werden. Die *algorithmische Realisierung* dieser Idee baut stark auf den dargestellten Konvergenzresultaten und den Zusammenhängen zur Klassifikation mit großem Margin auf (siehe Abb. 1).

Man kann sich an einem einfachem Beispiel überlegen, dass die fehlerfreie Lösung mit dem größtem Margin nicht robust ist. Das wird in Abb. 4 [links und Mitte] verdeutlicht: wird nur ein einziger Punkt verschoben (z.B. durch Rauschen), kommt es zu einer starken Drehung der Trennebene und der Margin wird sehr viel kleiner.

[1] Support-Vektor-Maschinen sind eine andere wichtige Methode zur Klassifikation mit großem Margin (Vap95; Sch97). Sie nutzen Eigenschaften der Mercer-Kerne aus, um implizit in sehr hoch dimensionalen Räumen rechnen zu können.

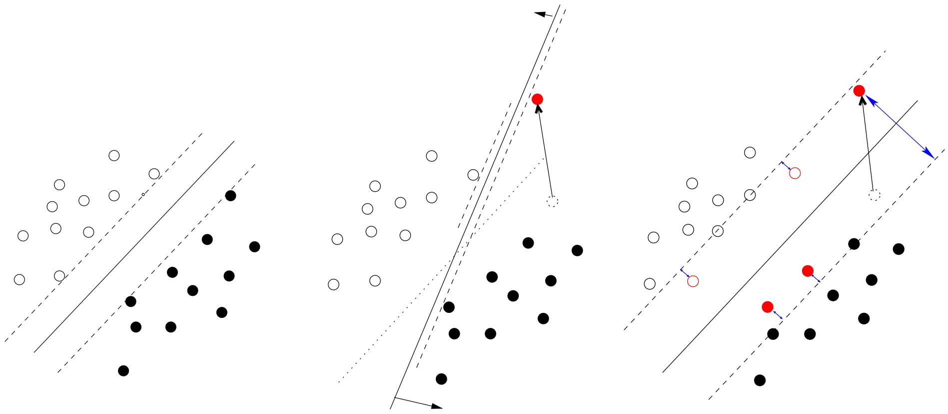

Abbildung 4: Finden der Trennebene mit größtem Margin: auf zuverlässigen Daten [links] und mit einem Ausreißer, der die Lösung sehr stark beeinflußt [Mitte]. Durch Einführung eines *Soft-Margins* [rechts] kann eine deutlich bessere Lösung gefunden werden. Die durchgezogene Linie zeigt die resultierende Trennebene und die gestrichelten Linien begrenzen den Margin-Bereich. (Abb. entnommen aus ROM01).

Die auf AdaBoost angewandte Lerntheorie beschäftigte bisher fast ausschließlich mit dem Fall trennbarer Probleme, bei dem der empirische Fehler Null ist. Die Theorie läßt sich jedoch auch für den nicht-trennbaren Fall anwenden und man erhält ein ähnliches Ergebnis wie in (3), bei dem der Margin-Fehler $R^{\rho}_{\text{emp}}[f, S]$ ein weiterer Summand ist:

$$R[f] \leq R^{\rho}_{\text{emp}}[f, S] + \frac{1}{\rho} G(\Omega[\mathsf{H}], N), \tag{5}$$

wobei $R^{\rho}_{\text{emp}}[f, S]$ angibt, welcher Anteil der Beispiele einen kleineren Margin als ρ haben.

Es gibt mehrere Wege, diese Idee algorithmisch umzusetzen: z.B. im Fall des Ausreißers in Abb. 4 könnte ρ größer gewählt werden, wenn dafür einige Margin-Fehler akzeptiert werden. Die Reduktion des Komplexitätsterms durch Vergrößerung des Margins würde größer sein, als die Anzahl der Margin-Fehler steigt (siehe Abb. 4 [rechts]). Eine zweite Strategie beruht auf der Beobachtung, dass sich die Gewichtungen in Boosting stark auf Ausreißer und andere, schwer zu klassifizierenden Beispiele konzentriert. Um zu verhindern, dass zu starkes Gewicht auf solche Beispiele gelegt wird, kann man die Gewichtung geeignet beschränken. Das führte zum Algorithmus AdaBoost$_{Reg}$ (ROM01).

Diese erste Idee läßt sich durch Einführung von Schlupfvariablen in das zugrundeliegende LP implementieren und führte zu zwei neuen Algorithmen – ν-Arc (RSS^{+}00) und C-Bar (Rät01). Für letzteren konnten wir Konvergenz zeigen. Das haben wir mit Hilfe der zuvor bewiesenen Konvergenzresultate Boosting-ähnlicher Methoden und des Zusammenhangs zu Barriere-Optimierungstechniken erreichen können. Außerdem haben wir gezeigt, dass die so erreichte Lösung in hohem Maße robust ist: eine weitere Verschiebung des Ausreißers senkrecht zur Hyperebene in Abb. 4 führt zu keiner Veränderung der Klassifikation.

In einer extensiven experimentellen Studie (ROM01; Rät01) wurden die neu vorgeschlagenen Algorithmen mit dem ursprünglichem AdaBoost-Algorithmus und anderen Algorithmen auf mehreren realen und künstlichen Datensätzen verglichen. *Wir konnten zeigen,* *dass die neu entwickelten Methoden deutlich robuster gegenüber Rauschen sind und deutlich bessere Klassifikatoren erzeugen. Im Vergleich zu anderen Methoden wie k-nächste-Nachbarn-Klassifikatoren, Entscheidungsbäumen und RBF-Netzen erreichen unsere Methoden exzellente Resultate.* Beim Vergleich zu Support-Vektor-Maschinen konnten keine signifikanten Unterschiede festgestellt werden. Es gibt jedoch Probleme, bei denen die neuen Boosting Algorithmen zu besseren Ergebnissen als SVMs führen (und andere, bei denen das andersherum ist). Das kann z.T. mit unterschiedlichen intrinsischen Eigenschaften der Algorithmen und Datensätze erklärt werden.

4 Eine Anwendung in der Energietechnik

In diesem Abschnitt wird die Anwendung unseres neu entwickelten, robusten Boosting Algorithmus – ν-Arc – in einem realen Problem in der Energietechnik diskutiert. Außer an diesem Problem haben wir noch an vielen anderen wissenschaftlich und wirtschaftlich interessanten Problemen gearbeitet: z.B. daran, vorherzusagen, ob oder wie stark ein Molekül an ein Protein bindet (Drug Design; siehe WRM$^+$02; RDB02); ob eine gegebene DNA- oder Aminosäure-Sequenz eine bestimmt biologische Eigenschaft hat (Translationsstart- und Spleißstellenerkennung; ZRM$^+$00; SRJM02; TKR$^+$02); welche Ziffer, Buchstaben oder Objekt ein Bild darstellt (Handschrift- und Objekterkennung; MRW$^+$00); wie sich eine chaotische Zeitreihe in der Zukunft verhält (Zeitreihenvorhersage MSR$^+$97; RDB02) und seit kurzem auch an der Erkennung von Unterschlagungsfällen in Firmen.

Solche Probleme sind besonders schwer zu analysieren, wenn es extrem wenige Trainingsbeispiele gibt. Die besten Methoden des maschinellen Lernens zeichnen sich dadurch aus, dass sie selbst dann sehr gute Leistungen erreichen. Das unterscheidet sie von herkömlichen Verfahren, die viele Trainingsbeispiele benötigen und nur mit niedrigdimensionalen Daten effizient arbeiten können. Deshalb habe ich für diesen Beitrag eine Anwendung gewählt, bei der es nur 36 Beispiele gibt, von denen nur 30 zum Training benutzt werden. Unsere Ergebnisse zeigen, dass selbst in diesem Fall exzellente Vorhersagen mit Boosting möglich sind.

Motivation Eines der schwierigsten Probleme der Energieindustrie sind kurzzeitige Spitzenbelastungen des Energienetzes. Um Ausfälle des Systems zu vermeiden, die bei Überbelastung auftreten, müssen heutzutage neue Kraftwerke gebaut werden. Man könnte sich aber auch ein Szenario in der Zukunft vorstellen, bei dem der Energieversorger dem Verbraucher mitteilt, dass z.B. 0.5% Energie gespart werden soll. Diese Möglichkeit könnte dann eingesetzt werden, wenn die Last des Netzes fast maximal ist und das System kurz vor dem Zusammenbruch steht. Die Reduzierung des Verbrauchs ist nicht bei allen Geräten möglich, aber fast 40% der Energie in Haushalten wird von Klimaanlagen und Kühlschränken verbraucht (1996 in Japan). Diese Geräte sind sehr oft mit einem Inverter-System ausgestattet, mit dem man den Energiebedarf stufenlos regeln kann (siehe Abb. 5).

Um ein solches Vorhaben zu realisieren, muß in einem ersten Schritt erkannt werden, welche Geräte in einem Haushalt eingeschaltet sind. Dazu ist es aber unpraktikabel, für jeden Verbraucher ein eigenes Meßgerät zur Verfügung zu stellen. Außerdem soll in den Haushalt möglichst nicht eingegriffen werden. Deshalb bleibt nur die Möglichkeit, den

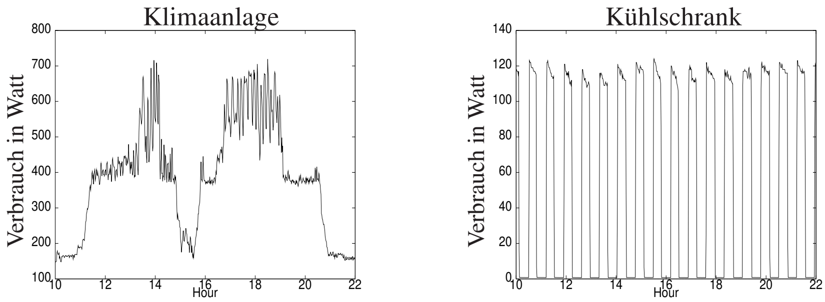

Abbildung 5: Verbrauchskurven einer Klimaanlage mit Inverter [links] und eines älteren Kühlschrank-Modells [rechts]. Die Kurve der Klimaanlage ist deutlich komplexer als die des Kühlschranks, der ein regelmäßiges An/Aus Verhalten zeigt. (Abb. entnommen aus ORM00).

Ein/Aus-Zustand der Geräte am Versorger-Haupteingang außerhalb der Wohnung/Hauses zu bestimmen. Dort kann man jedoch nur eine Überlagerung vieler verschiedener Verbrauchskurven messen. Daraus ergab sich die Aufgabe für unser Lernsystem: *Bestimme den Operationszustand einer Menge von verschiedenen Geräten anhand von Daten, die sich nicht-eingreifend erfassen lassen.* Ein ähnliches System wurde von (Car90) vorgestellt, das sich aber nicht für Invertersysteme verwenden läßt.

Ergebnisse In (ORM00) haben wir in einer Machbarkeitsstudie gezeigt, dass sich dieses Problem durch den Einsatz von Boosting Methoden und SVMs effizient lösen läßt (wurde zum Patent angemeldet). Dazu wurden 36 Kombinationen von Ein/Aus-Zuständen in einem Modell-Netz mit sechs Verbrauchern vermessen (siehe Tabelle 1; ein größeres Experiment ist in Vorbereitung). Für jede Kombination wurde eine Fourier-Analyse der gemessenen Verbrauchskurve durchgeführt und die Fourier-Koeffizienten als Merkmale für die Klassifikation benutzt. Für die Trainingsbeispiele ist bekannt, welche Geräte eingeschaltet sind. Dann haben wir verschiedene Lernverfahren (K-nächste-Nachbarn-Klassifikation (KNN), RBF-Netze, Support-Vektor-Maschinen und eine robuste Boosting-Methode: ν-Arc) auf dieses Problem angewandt und deren Generalisierungsfähigkeit durch Mittelung über mehrere zufällige Einteilungen in Trainings- und Testmenge bestimmt. Die Modellselektion wurde mit Kreuzvalidierung durchgeführt (Details in (ORM00)). Die Ergebnisse sind in Tabelle 1 dargestellt.

Wir können feststellen, dass die beste durchschnittliche Leistung von dem von uns vorgeschlagenem ν-Arc Algorithmus erreicht wird – gefolgt von den Support-Vektor-Maschinen (signifikanter Unterschied). Die anderen beiden Methoden schneiden deutlich schlechter

	KNN	RBF	ν-Arc	SVM
Klimaanlage	8.0±1.2	3.7±2.0	**1.9±0.5**	2.5±0.7
Kühlschrank mit Inv.	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
Kühlschrank ohne Inv.	35.7±1.4	26.9±2.5	**19.0±2.2**	24.7±1.3
Glühlampe	21.0±2.1	24.4±4.2	**11.5±1.1**	12.8±0.9
Leuchtstofflampe	35.8±1.5	16.4±3.6	12.9±1.1	**10.2±0.9**
Fernseher	3.3±0.5	2.8±1.4	1.3±0.5	**0.5±0.3**
Durchschnitt	17.3±1.1	12.2±2.3	**7.8±0.9**	8.5±0.6

Tabelle 1: Testfehler in % gemittelt über 20 zufällige Einteilungen in Trainings- und Testmenge.

ab. ν-Arc benutzte RBF-Netze als Basisalgorithmus und konnte das Ergebnis durch Kombination deutlich verbessern. Die SVM wurde mit RBF-Kernen trainiert und wir glauben, dass der kleine Unterschied zwischen ν-Arc und SVMs durch die nicht-adaptive Kernweite der SVM begründet werden kann. ν-Arc kann beide Vorteile verbinden: Klassifikation mit großem Margin und Sensitivität für Skaleninformation.

5 Zusammenfassung und Ausblick

In meiner Dissertation habe ich drei wichtige Themen bearbeitet (vgl. Abb. 1): (i) theoretische Grundlagen von Boosting und Zusammenhänge zur konvexen Optimierung, (ii) Verbesserung und Erweiterung von Boosting-Algorithmen, um sie robust zu machen und für Regressionsprobleme verwenden zu können und (iii) Anwendung der neuen Algorithmen in bisher nicht erschlossenen Anwendungsfeldern wie der Energietechnik.

Durch das Etablieren der Verbindungen von Boosting zu Optimierungsmethoden konnten vorhandene Algorithmen deutlich besser verstanden werden. Ich glaube, dass sich aus dieser Verbindung noch viele andere Ergebnisse und Konsequenzen für das Maschinelle Lernen ergeben werden (siehe z.B. MMZ02). In meiner Arbeit konnte ich das erarbeitete Verständnis nutzen, um neue Ideen für *robuste Methoden* zu implementieren und die Konvergenzeigenschaften neuer Algorithmen zu analysieren. Unsere Soft-Margin Idee für Boosting führte so zu Algorithmen, die in Simulationen mit exzellenten Ergebnissen abschneiden und die Verwendbarkeit von Boosting-Algorithmen auf Problemen mit viel Rauschen erst ermöglichen. Die in der Arbeit vorgestellten Resultate in realen Anwendungen zeigen, dass Boosting Algorithmen tatsächlich sehr gut funktionieren und ausgereift genug für weitere praktische Anwendungen sind.

Die weitere Entwicklung dieses faszinierenden Forschungsgebietes kann u.a. auf der von mir aufgebauten Web-Site `http://www.boosting.org` verfolgt werden.

Danksagung Herzlich danke ich den Betreuern der Arbeit: Klaus-Robert Müller, Manfred K. Warmuth und Kristin K. Bennett.

Literatur

[Bre96] L. Breiman. Bagging Predictors. *Machine Learning*, 26(2):123–140, 1996.

[Car90] J. Carmichael. Non-intrusive Appliance Load Monitoring System. EPRI journal, 1990.

[CV95] C. Cortes and V.N. Vapnik. Support Vector Networks. *Mach. Learn.*, 20:273ff, 1995.

[FS97] Y. Freund and R.E. Schapire. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. *J. of Comp. and System Sc.*, 55(1):119–139, 1997.

[MMZ02] S. Mannor, R. Meir, and T. Zhang. The Consistency of Greedy Algorithms for Classification. In *Proc. COLT*, Sydney, Springer Verlag, 2002. Im Druck.

[MRW+00] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, A.J. Smola, and K.-R. Müller. Invariant Feature Extraction and Classification in Kernel Spaces. *NIPS 12*, 526–532. MIT, 2000.

[MSR+97] K.-R. Müller, A.J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V.N. Vapnik. Predicting Time Series with Support Vector Machines. In *Artificial Neural Networks — ICANN'97*, pages 999–1004, Berlin, 1997. Springer LNCS 1327.

[ORM00] T. Onoda, G. Rätsch, and K.-R. Müller. A Non-Intrusive Monitoring System for Household Electric Appliances with Inverters. In *Proc. of NC'2000*, Berlin, 2000. ICSC.

[ORM00a] T. Onoda, G. Rätsch, and K.-R. Müller. Applying Support Vector Machines and Boosting to a Non-Intrusive Monitoring System for Household Electric Appliances with Inverters. In *Proceedings of NC'2000*, 2000.

[Rät01] G. Rätsch. *Robust Boosting via Convex Optimization*. PhD thesis, University of Potsdam, October 2001. http://www.boosting.org/papers/thesis.ps.gz.

[RDB02] G. Rätsch, A. Demiriz, and K. Bennett. Sparse Regression Ensembles in Infinite and Finite Hypothesis Spaces. *Machine Learning*, 48(1-3):193–221, 2002. Special Issue on New Methods for Model Selection and Model Combination.

[RMSM02] G. Rätsch, S. Mika, B. Schölkopf, and K.-R. Müller. Constructing Boosting Algorithms from SVMs: an Application to One-Class Classification. *IEEE PAMI*, 24(9), 2002.

[RMW02] G. Rätsch, S. Mika, and M.K. Warmuth. On the Convergence of Leveraging. In *NIPS 14*, MIT Press, 2002. Im Druck.

[ROM01] G. Rätsch, T. Onoda, and K.-R. Müller. Soft Margins for AdaBoost. *Machine Learning*, 42(3):287–320, 2001.

[RSS+00] G. Rätsch, B. Schölkopf, A.J. Smola, S. Mika, T. Onoda, and K.-R. Müller. Robust Ensemble Learning. In *Adv. in Large Margin Classifiers*, 207–219. MIT Press, 2000.

[RW02] G. Rätsch and M.K. Warmuth. Maximizing the Margin with Boosting. In *Proc. COLT'02, Sydney*, Springer Verlag, 2002. Im Druck.

[RWM+00] G. Rätsch, M. Warmuth, S. Mika, T. Onoda, S. Lemm, and K.-R. Müller. Barrier Boosting. In *Proc. COLT'00*, pages 170–179, 2000. Morgan Kaufmann.

[Sch97] B. Schölkopf. *Support vector learning*. Oldenbourg Verlag, Munich, 1997.

[SFBL98] R.E. Schapire, Y. Freund, P.L. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, October 1998.

[SRJM02] S. Sonnenburg, G. Rätsch, A. Jagota, and K.-R. Müller. New Methods for Splice-Site Recognition. In *Proc. ICANN*, 2002. Im Druck.

[TKR+02] K. Tsuda, M. Kawanabe, G. Rätsch, S. Sonnenburg, and K.R. Müller. A New Discriminative Kernel from Probabilistic Models. *Neural Computation*, 2002. Im Druck.

[Vap95] V.N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, 1995.

[WRM+02] M.K. Warmuth, G. Rätsch, M. Mathieson, J. Liao, and C. Lemmen. Active Learning in the Drug Discovery Process. In *NIPS 14*, MIT Press, 2002. Im Druck.

[ZRM+00] A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. Lengauer, and K.-R. Müller. Engineering Support Vector Machine Kernels That Recognize Translation Initiation Sites. *BioInformatics*, 16(9):799–807, 2000.

Gunnar Rätsch, geboren 1973, studierte Informatik und Physik in Potsdam und Zürich. Er erhielt den Jacob-Jacobi-Preis für den besten Abschluß in der naturwissenschaftlichen Fakultät (1998) in Potsdam. Nach dem Diplom in Informatik (1998) erstellte er an der Universität Potsdam und am Fraunhofer Institut für Rechnerarchitektur und Software-Technik seine Dissertation (2001). Er hat Forschungsaufenthalte an der Universität von Kalifornien in Santa Cruz, der Australian National University in Canberra und am Forschungsinstitut der japanischen Energieindustrie in Tokyo verbracht. Derzeit ist er an der Australian National University tätig, an der er seine wissenschaftlichen Interessen im Bereich des Maschinellen Lernens, der Mathematischen Optimierung und Bioinformatik weiter verfolgt.