FORSCHUNGSZENTRUM JÜLICH GmbH
Zentralinstitut für Angewandte Mathematik
D-52425 Jülich, Tel. (02461) 61-6402

Interner Bericht

Eine Einführung zu String-Kernen für die Sequenzanalyse mit Support-Vektor-Maschinen und anderen Kern-basierten Lernalgorithmen

Sebastian Schnitzler, Tatjana Eitrich

FZJ-ZAM-IB-2006-07

März 2006

(letzte Änderung: 28.3.2006)

Eine Einführung zu String-Kernen für die Sequenzanalyse mit Support-Vektor-Maschinen und anderen Kern-basierten Lernalgorithmen

Sebastian Schnitzler, Tatjana Eitrich

Zentralinstitut für Angewandte Mathematik
Forschungszentrum Jülich
http://www.fz-juelich.de

Zusammenfassung Support-Vektor-Maschinen (SVM's) sind ein Klassifikationsverfahren, das eine immer größere Bedeutung in der Analyse von Zeichenketten erlangt. In der Arbeit mit SVM's spielen Kerne eine bedeutende Rolle. In dieser Arbeit beschäfigen wir uns mit verschiedenen String-Kernen, die zur Analyse von Zeichenketten verwendet werden können. Wir stellen den Spektrum- und den Degree-Kern vor und beschäftigen uns mit speziellen Varianten, deren Unterschiede vor allem in einer unterschiedlichen Fehlertolerierung liegen. Zuletzt geben wir mögliche Anwendungsgebiete für SVM's mit String-Kernen an und dokumentieren bereits vorliegende Testergebnisse.

1 Support-Vektor-Maschinen und Kerne

Das Verfahren der Support-Vektor-Maschinen (SVM's) [1] ist ein mächtiges Instrument zur überwachten Klassifikation von Daten. Basierend auf der einfache Idee, eine trennende Hyperebene zur Klassentrennung zu erlernen, greifen Support-Vektor-Maschinen auf den Trick der Kernfunktionen [2] zurück, der es ermöglicht, nichtlineare Trennfunktionen zu generieren. Für die Klassifikation von Daten mit Support-Vektor-Maschinen ist die Arbeit mit derartigen Kernfunktionen ein grundlegender Bestandteil.

1.1 SVM-Hypothesenfunktion

Sei

$$\mathcal{T} := \left\{ \left(x^i, y_i\right) \in \mathcal{X} \times \{-1, 1\}, \quad i = 1, \ldots, t \right\} \tag{1}$$

ein Trainingsdatensatz mit $t \in \mathbb{N}_+$ Eingabe-Ausgabe-Punkten mit je $n \in \mathbb{N}$ Attribut-Werten. Das zentrale Optimierungsproblem bei der Anwendung von SVM's hat die Form

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^t} \quad \frac{1}{2} \sum_{i=1}^{t} \sum_{j=1}^{t} y_i y_j \alpha_i \alpha_j \langle x^i, x^j \rangle - \sum_{i=1}^{t} \alpha_i \tag{2}$$

unter den Nebenbedingungen $y^T\alpha = 0$ und $0 \leq \alpha \leq C$. Der Parameter $C > 0$ ist ein Parameter zur Fehlertolerierung. Im dargestellten Fall führt der Vektor α zu einer linearen Hypothese

$$h(x) = \operatorname{sgn}\left(\sum_{i=1}^{t} y_i \alpha_i \langle x^i, x\rangle + b\right) . \tag{3}$$

1.2 Kerne

Das zentrale Element in (2) und (3) ist das Skalarprodukt zwischen Punkten im Datenraum $\mathcal{X}$. Es erlaubt es uns, den Trick der Kerne zu verwenden. Die Idee dabei ist, dass die Hypothese (3) zu einer nichtlinearen Funktion wird, falls alle verwendeten Datenpunkte des Raumes $\mathcal{X}$ zunächst nichtlinear in einen hochdimensionalen Raum transformiert werden. Sei dazu

$$\Phi : \mathcal{X} \to \mathcal{F} \tag{4}$$

eine Abbildung vom Datenraum $\mathcal{X}$ in einen Hilbertraum $\mathcal{F}$ mit Dimension d, den sogenannten Merkmalsraum. Φ wird auch als Merkmalsabbildung bezeichnet. Dann wird (2) zu

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^t} \quad \frac{1}{2}\sum_{i=1}^{t}\sum_{j=1}^{t} y_i y_j \alpha_i \alpha_j \langle \Phi(x^i), \Phi(x^j)\rangle_{\mathcal{F}} - \sum_{i=1}^{t} \alpha_i \tag{5}$$

und die Hypothese hat die Form

$$h(x) = \operatorname{sgn}\left(\sum_{i=1}^{t} y_i \alpha_i \langle \Phi(x^i), \Phi(x)\rangle_{\mathcal{F}} + b\right) . \tag{6}$$

An dieser Stelle kann man Kernfunktionen zum Ersetzen der Skalarprodukte verwenden. Uns interessieren Funktionen $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, für die gilt [3]

$$K(x^i, x^j) = \langle \Phi(x^i), \Phi(x^j)\rangle_{\mathcal{F}} \quad \forall x^i, x^j \in \mathcal{X} . \tag{7}$$

Für zwei beliebige Punkte des Datenraumes soll der Wert der Kernfunktion genau dem Wert des Skalarproduktes der beiden transformierten Punkte im Merkmalsraum $\mathcal{F}$ entsprechen. Nicht jede Abbildung K mit dieser Eigenschaft ist auch tatsächlich ein Kern im Sinne der SVM-Theorie [4]. Eine symmetrische Funktion $K(x^i, x^j)$ ist genau dann eine Kernfunktion, wenn für einen beliebigen Trainingsdatensatz $\mathcal{T}$ sowie reelle Zahlen $\gamma_1, \ldots, \gamma_t$ die Ungleichung

$$\sum_{i=1}^{t}\sum_{j=1}^{t} \gamma_i \gamma_j K(x^i, x^j) \geq 0 \tag{8}$$

erfüllt ist [5]. Die Abbildung K muss also positiv semidefinit sein. Sei K eine Kernfunktion und seien $x^1 \ldots x^t \in \mathcal{X}$, so nennt man die $t \times t$ Matrix $\boldsymbol{K}$ mit $K_{ij} = K(x^i, x^j)\ \forall\ 1 \leq i, j \leq t$ Grammatrix oder Kernmatrix.

1.3 Standard-Kerne

Eine Reihe von Standard-Kernen ist bekannt [1]. Dazu zählen

- der Polynomial-Kern

$$K(x^i, x^j) := \langle x^i, x^j \rangle^d \quad (d \in \mathbb{N})$$

- der inhomogene Polynomial-Kern

$$K(x^i, x^j) := \left(\langle x^i, x^j \rangle + c\right)^d \quad (c \in \mathbb{R}_+, d \in \mathbb{N})$$

- der Gauß-Kern (auch RBF-Kern)

$$K(x^i, x^j) := e^{-\frac{\|x^i - x^j\|^2}{2\sigma^2}} \quad (\sigma \in \mathbb{R}_+)$$

- der Sigmoid-Kern

$$K(x^i, x^j) := \tanh\left(\kappa \langle x^i, x^j \rangle + \vartheta\right) \quad (\kappa \in \mathbb{R}_+, \vartheta \in \mathbb{R}_-) \ .$$

Diese Kerne werden erfolgreich zur Klassifikation numerischer Daten eingesetzt und stehen bei den frei verfügbaren Softwarepaketen zur Verfügung [6, 7]. In dieser Arbeit stellen wir einige weniger bekannter Kerne vor, welche zur Sequenzanalyse verwendet werden können. In den letzten Jahren haben sich die Anwendungsgebiete der Sequenzanalyse stark weiterentwickelt, sodass effiziente Methoden zur Klassifikation von großer Wichtigkeit sind. Das Verfahren der Support-Vektor-Maschinen setzt sich innerhalb der Verfahren des maschinellen Lernens immer stärker gegen andere Methoden durch. Aus diesem Grund ist die Bereitstellung und Untersuchung geeigneter Kerne für möglichst viele Formen der Datenausprägung eine wichtige Aufgabe.

2 Einfache String-Kerne zur Sequenzanalyse

Kerne besitzen die vorteilhafte Eigenschaft, dass sie nicht ausschließlich auf die Auswertung einheitlicher Vektoren numerischer Daten beschränkt sind, sondern auch auf anderen Objekten arbeiten können. Im Folgenden beschreiben wir Kerne, die dazu verwendet werden können, Strings zu vergleichen. Diese Kerne können dann im Zusammenhang mit Support-Vektor-Maschinen zur Klassifikation von Sequenzen eingesetzt werden. Diese Verbindung ist noch sehr jung und stellt reichlichen Boden für Verbesserungen und Effizienzsteigerungen zur Verfügung. Diese Arbeit soll eine Einführung in das Thema bieten. Eine Anwendungsmöglichkeit von String-Kern basierten Support-Vektor-Maschinen stellt zum Beispiel die Analyse von DNA-Sequenzen dar. DNA-Sequenzen bestehen aus Folgen der Buchstaben A, C, G und T, wobei die Länge solcher Folgen nicht fest ist.

2.1 Grundbegriffe

Bevor wir einige interessante String-Kerne vorstellen, müssen zunächst noch die grundlegenden Begriffe geklärt werden. Die Definitionen kann man in [8] nachlesen.

- Ein Alphabet Σ ist eine endliche Menge von Symbolen, die als Buchstaben bezeichnet werden. Mit $|\Sigma|$ sei im Folgenden die Anzahl verschiedener Buchstaben im Alphabet Σ bezeichnet.
- Ein String der Länge l sei eine endliche Folge $u = (u_1, \ldots, u_l)$ von Buchstaben eines Alphabetes.
- Als k-Teilfolge eines Strings bezeichnen wir im Folgenden ein Teilstück eines l-elementigen Strings, welches die Länge k hat ($k \leq l$).

2.2 Spektrum-Kern

Der Spektrum-Kern wurde erstmalig in [9] vorgestellt. Sei $\mathcal{X}$ der Raum aller endlichen Folgen eines Alphabets Σ mit $|\Sigma| = l$. Das k-Spektrum einer Folge $u \in \mathcal{X}$ sei die Menge aller k-Teilfolgen, die sie enthält. Weiterhin bezeichne Σ^k die Menge aller möglichen k-Teilfolgen im Alphabet Σ. Es ist klar, dass Σ^k die Dimension l^k hat. Wir definieren nun eine Abbildung $\Phi_k : \mathcal{X} \to \mathbb{R}^{l^k}$ durch

$$\Phi_k(u) := (\phi_a(u))_{a \in \Sigma^k} \ , \tag{9}$$

in der $\phi_a(u)$ der Häufigkeit entspricht, in der die k-Teilfolge a im String u vorkommt. Demzufolge ist das Bild von u also eine gewichtete Darstellung seines k-Spektrums. Nun kann man den k-Spektrum-Kern K_k definieren als

$$K_k(u, v) := \langle \Phi_k(u), \Phi_k(v) \rangle_{\mathcal{F}} \quad (u, v \in \mathcal{X}) \ . \tag{10}$$

Da $\mathcal{F} = \mathbb{R}^{l^k}$ gilt, ist das Skalarprodukt in (10) als das übliche Skalarprodukt des $\mathbb{R}^{l^k}$ zu verstehen. Es sei erwähnt, dass dieser Kern keine Informationen über die Positionen der Teilfolgen in die Bewertung einfließen lässt.

Ein vereinfachter k-Spektrum-Kern entsteht, wenn in (9) folgende abgewandelte Funktion ϕ verwendet wird:

$$\phi_a(u) = \begin{cases} 1, \text{ falls } a \in u \\ 0, \text{ sonst} \end{cases} \ .$$

Jede Koordinate von Φ kann dann ausschließlich binäre Werte annehmen. Der Wert 1 wird angenommen, falls die Teilfolge a in u enthalten ist, die Anzahl der Treffer spielt jetzt keine Rolle mehr. Eine reale Gewichtung des k-Spektrums geht damit jedoch verloren.

Es ist nun zu beachten, dass der Merkmalsraum $\mathbb{R}^{l^k}$ sehr groß ist, aber die Vektoren im Allgemeinen nur sehr schwach besetzt sind. Die Anzahl der Koordinaten, die nicht 0 sind, ist nach oben durch $(|u| - (k-1))$ beschränkt. Diese Eigenschaft ist nützlich für eine effiziente Arbeitsweise des Kerns. Eine

effiziente Berechnung des Kerns kann mit Hilfe von Suffix-Bäumen erfolgen [9]. Suffix-Bäume werden beispielswiese in [10] vorgestellt. Der Wert des Kerns wird über das Durchlaufen eines Suffix-Baumes berechnet. Dank der vorteilhaften Eigenschaften des zugehörigen Suffix-Baumes geschieht das in linearer Zeit. Die Effizienz der Kernberechnungen spielt im Zusammenhang mit der Anwendung von Support-Vektor-Maschinen eine sehr wichtige Rolle. Die Auswertung des Kerns verbraucht bei großen Datensätzen bis zu 90% der Rechenzeit und gilt im Allgemeinen als der Flaschenhals des ganzen Verfahrens. Der hohe Zeitaufwand zur Berechnung der Kernmatrix stellt auch heute noch ein Hindernis bei der Nutzung von SVM's dar. Aus diesem Grund ist die schnelle Berechenbarkeit des k-Spektrum-Kerns als besonders positiv zu bewerten.

2.3 Gewichteter Degree-Kern

Der Spektrum-Kern hat die Eigenschaft, Teilfolgen an beliebigen Stellen zu suchen, ohne Informationen über deren genaue Position zu bewerten. Diese Tatsache kann sich in speziellen Anwendungen als Nachteil herausstellen. Eine passende Alternative stellt der gewichtete Degree-Kern [11] dar. Die Grundidee dieses Kerns besteht darin, das exakte Zusammentreffen von k-Teilfolgen an gleichen Stellen von Sequenzen zu finden. Der gewichtete Degree-Kern der Ordnung κ vergleicht zwei Folgen u und v der Länge l durch Summation aller Beiträge von gleichen Teilfolgen der Länge $k \in \{1, \ldots, \kappa\}$ gewichtet durch Koeffizienten β_k, formal

$$K_\kappa(u, v) := \sum_{k=1}^{\kappa} \beta_k \sum_{i=1}^{l-k+1} \mathbf{I}\left(a_{k,i}(u) = a_{k,i}(v)\right) \ . \tag{11}$$

Mit $a_{k,i}(u)$ bzw. $a_{k,i}(v)$ bezeichnen wir k-elementige Teilfolgen der Folgen u bzw. v, die bei der Position i starten. $\mathbf{I}$ ist die übliche Indikatorfunktion. Als Gewichte haben sich

$$\beta_k := 2(\kappa - k + 1)/(\kappa(\kappa + 1))$$

bewährt [11]. Teilfolgen sind also je nach Länge gewichtet. Es ist zu beachten, dass zwar $\beta_{k+1} < \beta_k$ für alle $k \in \{1, \ldots, \kappa - 1\}$ gelten, sich aber dennoch längere Übereinstimmungen stärker auswirken als kurze, da jede lange Übereinstimmung gleichzeitig auch verschiedene kürzere Übereinstimmungen impliziert. Grob gesehen kann man diesen Kern als Spektrum-Kern interpretieren.

3 String-Kerne mit Fehlertolerierung

Die beiden bisher vorgestellten Kerne (10) und (11) haben den Nachteil, dass sie zu einem sehr großen Merkmalsraum führen, der jedoch zum größten Teil nicht genutzt wird. Deswegen gibt es Möglichkeiten zur Optimierung der Modelle mittels Modifizierungen der einfachen Kerne. Die Idee dabei ist, beim Vergleich von Strings einzelne Fehler zuzulassen. Fehler bedeuten in diesem Zusammenhang, dass einzelne Buchstaben in den Strings nicht übereinstimmen. Betrachtet man lange k-Teilfolgen, so läuft die Wahrscheinlichkeit, die gleiche Teilfolge in einer

weiteren Folge zu entdecken, sehr schnell gegen Null. Deswegen kann es von Vorteil sein, einige Fehlstände zuzulassen.

3.1 Spektrum-Kern mit Fehlständen

Sei a eine k-Teilfolge in Σ^k. Dann definieren wir die m-Umgebung $U_{m,k}$ von a als die Menge der k-Teilfolgen b, die sich von a an höchstens m Stellen unterscheiden. Die Merkmalsabbildung für a ist dann definiert als

$$\Phi_{m,k}(a) := (\phi_b(a))_{b \in \Sigma^k} \ , \tag{12}$$

wobei für ϕ die Vorschrift

$$\phi_b(a) := \begin{cases} 1, \text{ falls } b \in U_{m,k}(a) \\ 0, \text{ sonst} \end{cases}$$

gilt. Für einen beliebigen String u gilt dann

$$\Phi_{m,k}(u) := \sum_{\forall a \in u} \Phi_{m,k}(a) \ , \tag{13}$$

sodass der Kern berechnet wird als

$$K_{m,k}(u,v) := \langle \Phi_{m,k}(u), \Phi_{m,k}(v) \rangle \quad (u, v \in \mathcal{X}) \ . \tag{14}$$

Offensichtlich entspricht dieser Kern für $m = 0$ genau dem Spektrum-Kern.

3.2 Gewichteter Degree-Kern mit Fehlständen

Analog zum Spektrum-Kern lässt sich auch der gewichtete Degree-Kern so verändern, dass er Fehlstände berücksichtigt. Eine Methode für die Umsetzung dieses Kerns ist [11]

$$K_{m,\kappa}(u,v) = \sum_{k=1}^{\kappa} \sum_{j=0}^{m} \beta_{k,j} \sum_{i=1}^{l-k+1} \mathbf{I}(a_{k,i}(u) \neq_j a_{k,i}(v)) \ , \tag{15}$$

wobei $(a_{k,i}(u) \neq_j a_{k,i}(v))$ bedeutet, dass es genau j Fehlstände zwischen u und v gibt. Eine sinnvolle Wahl für $\beta_{k,j}$ ist [11]

$$\beta_{k,j} = \begin{cases} \beta_k / \left(\binom{k}{j} (\mid \Sigma \mid -1)^j \right), \text{ falls } k > j \\ 0 \text{ sonst} \end{cases} \ .$$

3.3 Lücken-Kern

Der sogenannte beschränkte (l,k)-Lücken-Kern nutzt den $|\Sigma|^k$-dimensionalen Merkmalsraum, der durch die Teilfolgen der Länge k bestimmt wird. In seiner Arbeitsweise unterscheidet er sich von den beiden bisher vorgestellten Kernen.

Sei $a = (a_1, \ldots, a_l)$ ein fester String der Länge $l \in \mathbb{N}$, wobei wie üblich $a_i \in \Sigma \; \forall \; i \in \{1, \ldots, l\}$ gilt. Sei $T_{(l,k)}(u)$ die Menge aller Teilfolgen b der Länge k, die in a vorkommen, so definieren wir eine Merkmalsabbildung als

$$\Phi_{l,k}(a) = (\phi_b(a))_{b \in \Sigma^k} \;\; , \tag{16}$$

wobei

$$\phi_b(a) := \begin{cases} 1, \text{ falls } b \in T_{(l,k)}(a) \\ 0, \text{ sonst} \end{cases} \quad . \tag{17}$$

Diese Abbildung kann man nun für Trainingsdaten u beliebiger Länge erweitern. Dazu wird die Merkmalsabbildung über alle l-Teilfolgen a des Strings u aufsummiert. Die neue Abbildung, die definiert ist als

$$\Phi_{l,k}^{\text{gap}}(u) := \sum_{\forall a \in u} \Phi_{l,k}(a) \; , \tag{18}$$

nennen wir Lücken-Merkmalsabbildung[1]. Für zwei Strings u und v wird der Kern analog zu den bisher vorgestellten Kernen als Skalarprodukt berechnet mittels

$$K_{\text{gap}}(u, v) := \langle \Phi_{(l,k)}^{\text{gap}}(u), \Phi_{(l,k)}^{\text{gap}}(v) \rangle \; . \tag{19}$$

Der Lücken-Kern zeichnet sich dadurch aus, dass er Unregelmäßigkeiten beim Matching toleriert, indem eine bestimmte Anzahl von Lücken möglich ist.

3.4 Substitutions-Kern

Der Substitutions-Kern baut auf einer speziellen Merkmalsabbildung auf, die ihrerseits eine sogenannte Mutationsumgebung verwendet. Dazu definieren wir eine Mutationsumgebung $M_{\sigma,k}$ für einen String $a = (a_1, \ldots, a_k)$ mittels

$$M_{\sigma,k}(a) := \left\{ b = (b_1, \ldots, b_k) \in \Sigma^k : -\sum_{i=1}^{k} \log P(a_i | b_i) < \sigma \right\} \; . \tag{20}$$

Die Definition von Nachbarschaft in (20) basiert also auf einem wahrscheinlichkeitstheoretischem Modell zur Substitution von Buchstaben. Um die maximale Größe der Mutationsumgebung zu kontrollieren, sollte der Parameter σ so gewählt werden, dass

$$\max_{a \in \Sigma^k} |M_{\sigma,k}(a)| < N$$

gilt, wobei N a priori gewählt werden muss [12]. Damit ist auch klar, dass die Wahl von σ, dessen Wert die Höhe möglicher Mutationen bestimmt, nicht trivial ist. Die Anwendung von Kreuzvalidierungsmethoden sollte in Betracht gezogen werden. Ein Nachteil dieses Kerns ist deshalb die teure Parameteranpassung.

[1] gap: engl. Lücke

Für beliebige Strings u hat die zum Substitutions-Kern führende Merkmalsabbildung die Form

$$\Phi_{\sigma,k}^{\text{sub}}(u) := \sum_{\forall a \in u} (\phi_b(a))_{b \in \Sigma^k} \ . \tag{21}$$

Dabei bezeichnen wir mit a – wie auch im Abschnitt 3.3 – eine k-Teilfolge von u. Die Abbildung ϕ ist dabei analog zu (17) definiert als

$$\phi_b(a) := \begin{cases} 1, \text{ falls } b \in M_{\sigma,k}(a) \\ 0, \text{ sonst} \end{cases} \ .$$

Sie enthält die Mutationsumgebung. Der Substitutions-Kern entsteht analog zu (19) über das Skalarprodukt im Merkmalsraum.

3.5 Joker-Kern

Der sogenannte Joker-Kern ermöglicht es, nicht vollkommen passende Teilfolgen als gleich zu klassifizieren, indem an bestimmten Stellen der Strings sogenannte Joker als Platzhalter deklariert werden können. Zu einem Joker passt dann jeder Buchstabe des Alphabets. Dazu wird das Standardalphabet Σ um ein Joker-Zeichen erweitert. Dieser neue Buchstabe wird mit $*$ bezeichnet. Der Merkmalsraum $\mathcal{F}$ wird nun bestimmt durch die k-Teilfolgen in $\Sigma \cup \{*\}$, wobei der Buchstabe $*$ darin mindestens m mal auftreten soll. Offensichtlich hat dieser Merkmalsraum die Dimension

$$d = \sum_{i=0}^{m} \binom{k}{i} |\Sigma|^{k-i} \ .$$

Sein nun a eine k-Teilfolge. Dann passt a zu einer anderen k-Teilfolge b ($a \rightsquigarrow b$), falls

$$a_i = b_i \quad \forall b_i \neq *$$

gelten. Alle Buchstaben in b, die keine Joker sind, müssen also zu den entsprechenden Einträgen in a passen. Die zugehörige Merkmalsabbildung ist natürlicherweise definiert als

$$\Phi_{m,\lambda,k}^{*}(u) := \sum_{a \in u} (\phi_b(a))_{b \in \mathcal{F}} \tag{22}$$

mit

$$\phi_b(a) = \begin{cases} \lambda, \text{ falls } a \rightsquigarrow b \\ 0 \text{ sonst} \end{cases} \ ,$$

wobei $\lambda \in (0, 1]$ ein Gewichtungsparameter ist. Es existieren Ansätze, bei denen das Gewicht λ nicht fest, sondern variabel ist [12].

3.6 Fazit

Wir haben eine Reihe von Kernen zur Sequenzanalyse vorgestellt. Ausgehend von zwei einfachen Kernen, die auf genauen Übereinstimmungen von Teilfolgen basieren, wurden Kerne gezeigt, welche die wichtige Funktionalität der Mutation einzelner Buchstaben zulassen. Für reale Anwendungen spielen derartige Phänomene eine große Rolle [13].

Alle in dieser Arbeit vorgestellten String-Kerne zeichnen sich dadurch aus, dass die Merkmalsabbildungen Φ explizit bekannt sind. Diese Eigenschaft erlaubt es, die SVM-Hypothesenfunktion h zu interpretieren. Insbesondere kann der Normalenvektor w der trennenden Hyperebene im Merkmalsraum direkt berechnet werden als

$$w = \sum_{i=1}^{t} y_u \alpha_i \Phi(x^i). \tag{23}$$

Dieser kann dann beispielsweise dazu verwendet werden, zu untersuchen, welche Abstände einzelne Trainingspunkte zur Hyperebene aufweisen. Bei den Standardkernen, wie wir sie auf Seite 3 vorgestellt haben, ist der Merkmalsraum nicht bekannt und die Hyperebene kann nur abstrakt über die Werte der Kerne angegeben werden. Aus diesem Grund werden die SVM's oft als Black-Box-Verfahren bezeichnet.

4 Anwendungsgebiete

String-Kerne haben vielfältige Anwendungsgebiete in den unterschiedlichsten Forschungsbereichen. In diesem Abschnitt geben wir einen kurzen Überblick zu einigen konkreten Anwendungsbeispielen, bei denen die in dieser Arbeit vorgestellten Kerne erfolgreich eingesetzt worden sind.

Der einfache Spektrum-Kern sowie seine verallgemeinerte Form mit Fehlertolerierung sind in [9, 13] zum Vergleich von Proteinsequenzen und zur Klassifikation von Proteinen eingesetzt worden. Es hat sich gezeigt, dass der generalisierte Kern für kleine Werte von m, genauer gesagt für $m = 1$, den einfachen Spektrum-Kern schlagen konnte. Eine Support-Vektor-Maschine mit dem gewichteten Degree-Kern wurde in [11] zur Erkennung von sogenannten Splice-Sites in der DNA verwendet. Die Kerne aus den Abschnitten 3.3-3.5 wurden in [12] für einen Benchmark-Datensatz der Datenbank $SCOP$[2] getestet. Als besonders positiv konnte dabei die Geschwindigkeit der Algorithmen bewertet werden.

Zusätzlich zu den Applikationen innerhalb der Bioinformatik, findet man weitere Anwendungen der String-Kerne in der Kategorisierung von Texten [14], bei der Filterung von Spam-Nachrichten und in der Spracherkennung. Da herkömmliche Methoden nicht ausreichen, verwendet [15] String-Kerne zur Erkennung der Sprache in der Web-Sites verfasst sind. Die erzielten Ergebnisse sind sehr vielversprechend. Diese Anwendungsgebiete zeichnen sich alle dadurch aus, dass sie mit

[2] http://scop.berkeley.edu

Strings unbekannter Länge über einem bekannten Alphabet arbeiten. Die Analogie zur Klassifikation von DNA- und Proteinsequenzen ist damit offensichtlich.

Wie weitverbreitet mittlerweile die Anwendung von Support-Vektor-Maschinen mit String-Kernen ist, verdeutlicht eine etwas exotischere Anwendung, die in [16] zu finden ist. Dort wird gezeigt, wie man berühmte Interpreten anhand deren unterschiedlicher Arten zu musizieren automatisch erkennen kann. Fachleute sind über die hohen Erkennungsraten von über 90% sehr erstaunt [17] und nutzen die gewonnenen Erkenntnisse unter anderem dazu, dass Computer eigenständig lernen, möglichst ausdrucksvolle Musik zu komponieren oder zu spielen.

Literatur

1. Cristianini, N., Shawe-Taylor, J.: An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press (2000)
2. Schölkopf, B.: The kernel trick for distances. In: NIPS. (2000) 301–307
3. Schölkopf, B., Smola, A.J.: Learning with kernels. MIT Press (2002)
4. Mercer, J.: Functions of positive and negative type and their connection with the theory of integral equations. Philosophical Transactions of the Royal Society, London A **209** (1909) 415–446
5. Genton, M.G.: Classes of kernels for machine learning: a statistics perspective. Journal of Machine Learning Research **2** (2001) 299–312
6. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. (2001) Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.
7. Celis, S., Musicant, D.R.: Weka-parallel: machine learning in parallel. Computer Science Technical Report 2002b, Carleton College (2002)
8. Herbrich, R.: Learning kernel classifiers - theory and algorithms. MIT Press (2002)
9. Leslie, C., Eskin, E., Noble, W.S.: The spectrum kernel: a string kernel for SVM protein classification. In: Proceedings of the 7. Pacific Symposium on Biocomputing (PSB 2002). (2002) 564–575
10. Universität Dortmund Fachbereich Informatik, L.: Suffix Arrays – Eine Datenstruktur für Stringalgorithmen. (2006) http://ls11-www.cs.uni-dortmund.de/lehre/AE/Suffix1.pdf.
11. Sonnenburg, S., Rätsch, G., Schölkopf, B.: Large scale genomic sequence SVM classifiers. In: Proceedings of the 22. International Conference on Machine Learning. (2005)
12. Leslie, C., Kuang, R.: Fast kernels for inexact string matchings. In: Proceedings of the 16. Conference on Computational Learning Theory and 7. Kernel Workshop (COLT/Kernel'2003). Volume 2777 of LNCS., Springer (2003) 114–128
13. Leslie, C., Eskin, E., Cohen, A., Weston, J., Noble, W.S.: Mismatch string kernels for discriminative protein classification. Bioinformatics **20**(4) (2004)
14. Lodhi, H., Saunders, C., Shawe-Taylor, J., Christiani, N., Watkins, C.: Text classification using string kernels. Journal of Machine Learning Research **2** (2002) 419–444
15. Kruengkrai, C., Srichaivattana, P., Sornlertlamvanich, V., Isahara, H.: Language identification based on string kernels. In: Proceedings of the 5th International Symposium on Communications and Information Technologies (ISCIT-2005). (2005)
16. Saunders, C., Hardoon, D.R., Shawe-Taylor, J., Widmer, G.: Using string kernels to identify famous performers from their playing style. In: 15th European Conference on Machine Learning (ECML). (2004) 384–395

17. Widmer, G.: Musikalisch intelligente Computer Anwendungen in der klassischen und populären Musik. Informatik Spektrum **28**(5) (2005) 363–368