Maschinelles Lernen mit Prototypmethoden in der klinischen Proteomik

Frank-Michael Schleif

Die klinische Proteomik untersucht proteinbasierte Krankheitsprozesse durch Messung hochdimensionaler Spektren. Dies verlangt eine problemangepasste Vorverarbeitung und Algorithmik für die Erzeugung von statistischen Modellen. Prototypen basierte Algorithmen sind dafür besonders geeignet. In diesem Beitrag werden wesentliche Erweiterungen, wie Metrikadaptation, Fuzzy-Klassifikation und aktiven Lernens, für prototypen basierte Verfahren skizziert und auf klinischen Datensätzen getestet.

1 Einführung

Die klinische Proteomik untersucht proteinbasierte Krankheitsprozesse in klinischen Proben. Die Messung der Probe erfolgt dabei typischer Weise durch ein Massenspektrometer (siehe Abb. 1). Dabei entstehen hochdimensionale Spektren, die die Expressivität von bestimmten Proteinfragmenten anzeigen. Eine weitere Herausforderung ist die eher geringe Anzahl von Proben. Zudem ist die Güte und Interpretierbarkeit der Klassifikationsentscheidung von besonderer Bedeutung und die Adaptierbarkeit der generischen Klassifikationsmodelle bei Nachmessungen. Entsprechend werden die Spektren zur Weiterverarbeitung geeignet reduziert. Nach geeigneter Evaluierung, können diese für die Analyse und Diagnostik von Krankheitsprozessen in Frage kommen. Wir betrachten kurz die Aufbereitung der Spektren, nachfolgend werden Konzepte prototypischer Klassifikationsverfahren beschrieben und deren Erweiterungen für die klinische Proteomik skizziert. Im Ergebnisteil wird die entwickelte Algorithmik zur Bildung von Klassifikationsmodellen für verschiedenen klinische Datensätze eingesetzt und bewertet.

2 Massenspektrometrie

Die Massenspektrometrie ist eine Technik zur Messung von Masse zu Ladungsverhältnissen von Ionen in chemischen Strukturen (siehe auch [9] und Abb. 1). Die gemessenen Spektren bestehen aus mehreren 10000 Messpunkten, wobei lediglich Bereiche lokal höherer Intensität (Peaks) von Bedeutung sind, da diese auf das Vorhandensein bestimmter Fragmente in der Probe hinweisen. Zunächst werden die Spektren basislinienkorrigiert (Artefakt der Messung) und aligniert, so dass Peaks an korrespondierende Massenpositionen korrekt verglichen werden können (siehe [11]). Die Identifikation der Peaks (Peakpicking) geschieht dabei über eine lokale Maxima-Suche unter Berücksichtigung bestimmter Geräteeigenschaften. Danach ist jedes Spektrum als Linienspektrum (auf identischem Gitter) repräsentiert, welches nur noch aus verschiedenen Peaks besteht, aus denen dann einfache Merkmale (hier Flächen) abgeleitet werden. Die dadurch erhaltene Matrix von Merkmalen dient als Eingaberaum für die Klassifikationsmodelle. Wesentlich ist dabei, dass zu einem Merkmal stets die relative Massenposition im Originalspektrum erhalten bleibt um, die zugrunde liegende Masse weiter analysieren zu können. Diese Forderung

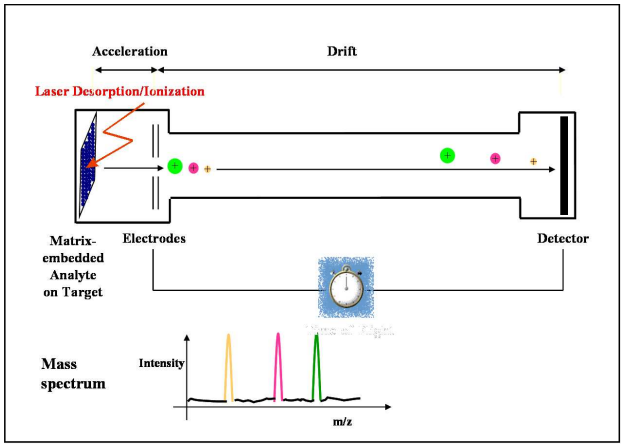

Figure 1: Wesentlicher Teile eines MALDI-TOF Massenspektrometers. Links die in Matrix eingebettete Probe, welche ionisiert und in Richtung des Detektors beschleunigt wird (Probe fragmentiert). Unterschiedliche molekulare Massen der Probe benötigen, bei gleicher zugeführter Energie, unterschiedlich lange, um die Flugstrecke zurückzulegen. Aus diesen Informationen kann die Struktur der Probe ermittelt werden.

begründet auch, warum bestimmte komplexere Klassifikationsmethoden (z.B. Kernelansätze) kaum anwendbar sind.

3 Datenrepräsentation

Prototypenbasierte Vektorquantisierungsmethoden [8] generieren robuste adaptive Klassifikationsmodelle [6, 12]. Die erste Variante, mit einer Kostenfunktion, ist der so genannte Generalized LVQ (GLVQ) [10] Algorithmus mit Erweiterungen zur Laufzeitoptimierung und Konvergenzverbesserung im Supervised Relevance Neural GAS (SRNG) [4]. Dabei sind Eingaben durch $\mathbf{v}$ mit Klassenbezeichner $c_{\mathbf{v}} \in \mathcal{L}$ gegeben. Sei $\mathcal{L}$ die Menge der Klassenbezeichner mit $\#\mathcal{L} = N_{\mathcal{L}}$ und $V \subseteq \mathbb{R}^{D_V}$. GLVQ/SRNG verwenden eine fixe Anzahl von Prototypen für jede Klasse. Sei $\mathbf{W} = \{\mathbf{w_r}\}$ die Menge von Prototypen und $c_{\mathbf{r}}$ die Klassenbezeichner von $\mathbf{w_r}$. Desweiteren, sei $\mathbf{W}_c = \{\mathbf{w_r} | c_{\mathbf{r}} = c\}$ die Untermenge der Prototypen, der Klasse $c \in \mathcal{L}$. Die Klassifikation wird dann durch die Abbildung Ψ in Form einer winner-take-all Regel

Methode	SNG	SNG	SNG	GLVQ	GLVQ	SNPC	SNPC	SNPC	FSNPC	FSNPC	FSNPC	QDA	LDA	SVM
Metrik	Euc	Euc_λ	Euc_{λ_r}	TM	MM	Euc	Euc_λ	Euc_{λ_r}	Euc	Euc_λ	Euc_{λ_r}	MM	MM	Poly
WDBC	95	97	96	96	89	89	95	97	93	91	95	94	96	87
Proteom_1	78	84	76	78	78	79	87	87	70	87	87	75	81	76
Proteom_2	87	93	85	65	90	82	87	91	92	93	91	71	81	85

Table 1: Modellgenauigkeiten in %, mit den Metriken Euklidisch(Euc), skaliert Euklidisch (Euc_λ), lokal skaliert Euklidisch (Euc_{λ_r}), Mahalanobismetrik(MM), Tanimotometrik (TM) und einem Polynomialkernel (Poly) mit $\gamma = \frac{1}{1000}, r = 1.0, d = 2.0$.

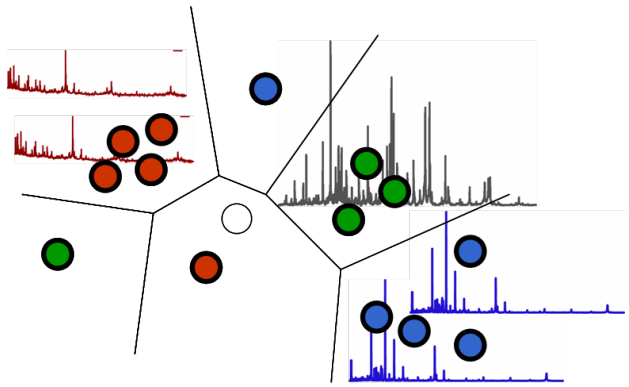

Figure 2: Prototypen basierter Klassifikation für 2d Daten mit multiplen Clustern. Die Prototypen sind jeweils durch einen zusätzlichen Kreis dargestellt und führen zu einer Zerlegung des Datenraums anhand der rezeptiven Felder (graue Netzstruktur).

realisiert, z.B. ein Eingabevektor $\mathbf{v} \in V$ wird auf den Prototypen abgebildet $\mathbf{s} \in A$ als Vektor $\mathbf{w}_s$ zu dem er am nächsten liegt $\mathbf{v}$,

$$\Psi_{\mathcal{V}\to\mathcal{A}} : \mathbf{v} \mapsto \mathbf{s}(\mathbf{v}) = \underset{\mathbf{r}\in A}{\text{argmin}}\, d(\mathbf{v}, \mathbf{w}_\mathbf{r}) \tag{1}$$

mit $d(\mathbf{v}, \mathbf{w})$ ein Ähnlichkeitsmaß. Der Prototyp $\mathbf{s}$ wird als Sieger bezeichnet. Die Untermenge die auf einen bestimmten Prototypen abgebildet wird (entsprechend (1)), bildet das rezeptive Feld des Prototypen. GLVQ-Training verändert die Prototypen im Datenraum, so dass für jede Klasse $c \in \mathcal{L}$, die entsprechenden Prototypen $\mathbf{W}_c$ die Klasse so genau wie möglich repräsentieren. Ziel ist die Generierung eines Modells welches die Daten per Prototypen sowohl in ihrer Datenverteilung (unüberwacht) als auch in ihrer Klassenstruktur (überwacht) repräsentiert. Soft Nearest Prototype Classification (SNPC) ist ein weiteres Prototypen basiertes Verfahren (eingeführt in [12]). Dabei werden sogenannte soft-assignments ($u_\tau(\mathbf{r}|\mathbf{v}_k)$) für Datenvektoren zu den Prototypen definiert, die eine statistische Interpretation als normalisierte Gausskernel über einer Metrik $d(\mathbf{v}_k, \mathbf{w}_\mathbf{r})$ besitzen. Im original SNPC betrachtet man:

$$E(\mathcal{S}) = \frac{1}{N_\mathcal{S}} \sum_{k=1}^{N_\mathcal{S}} \sum_{\mathbf{r}} u_\tau(\mathbf{r}|\mathbf{v}_k) \left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) \tag{2}$$

als eine Kostenfunktion mit $\mathcal{S} = \{(\mathbf{v}, c_\mathbf{v})\}$ als die Menge aller Eingabedaten, $N_\mathcal{S} = \#\mathcal{S}$. Die Klassenzuweisungsvariablen $\alpha_{\mathbf{r},c_{\mathbf{v}_k}} = 1$ wenn $c_{\mathbf{v}_k} = c_\mathbf{r}$ und 0 sonst (crisp). $u_\tau(\mathbf{r}|\mathbf{v}_k)$ ist die Wahrscheinlichkeit, dass ein Eingabevektor $\mathbf{v}_k$ dem Prototypen $\mathbf{r}$ zugewiesen wird. Eine crisp *winner-takes-all* Abbildung (1) durch verschwindende Kernelbreite, mit $c_{\mathbf{v}_k}$ und $d(\mathbf{v}_k, \mathbf{w}_\mathbf{r})$ fixiert, ergibt $u_\tau(\mathbf{r}|\mathbf{v}_k) = \delta(\mathbf{r} = \mathbf{s}(\mathbf{v}_k))$ (Details in [12]).

3.1 Relevance learning für SNPC

SNPC hängt erheblich von der verwendeten Metrik ab. Für hochdimensionale Daten, ist die Wahl der normalen euklidischen Metrik häufig ungünstig, da viele Eingabedimensionen, aus Sicht der Klassifikationsfragestellung, Rauschen kodieren. Relevanzlernen [5], bietet die Möglichkeit, die Metrikparameter mit zu lernen und wird für SNPC als SNPC-R eingeführt. Ein Parametervektor $\lambda = (\lambda_1, \ldots, \lambda_m)$ wird der verwendeten Metrik zugewiesen, nun bezeichnet als $d^\lambda(\mathbf{v}_k, \mathbf{w}_\mathbf{r})$. Ein häufiges Beispiel ist die skalierte euklidische Metrik $d^\lambda(\mathbf{v}_k, \mathbf{w}_\mathbf{r}) = \sum_{i=1}^{D_V} \lambda_i (\mathbf{v}_k^i - \mathbf{w}_\mathbf{r}^i)^2$ Parallel zur Optimierung der Prototypen werden nun auch die Relevanzparameter λ_j entsprechend des Klassifikationsproblems, adaptiert (Details siehe [11]). Werden die Metrikparameter pro Prototyp oder Klasse adaptiert erhält man den lokalen SNPC-R Algorithmus (siehe [11]). Im Falle der skalierten euklidischen Metrik können die λ_j in einem Relevanzprofil analysiert werden, welches die *Relevanz* jedes Merkmals bzw. Peaks für die Klassenseparation anzeigt. Damit kann nun eine klassifikationsbezogene Reduktion der Linienspektren erfolgen.

Wie in [2] gezeigt sind standard NPCs mit euklidischer Metrik, large margin Algorithmen mit dimensionsunabhängigen Generalisierungsschranken. In [4] konnte dieses Schemata für NPC-Algorithmen mit adaptiver Diagonalmetrik erweitert werden und in [11] wurde dies für NPCs mit lokaler Metrikparameter nachgewiesen.

3.2 Fuzzy Klassifikation für SNPC-R

Für den *Fuzzy gelabelten* SNPC (FSNPC) werden nun fuzzy Werte (d.h. unsichere Entscheidungen) für $\alpha_{\mathbf{r},c}$ zugelassen, die anzeigen, in wie weit ein Datenpunkt einem rezeptiven Feld zugewiesen ist so dass $0 \leq \alpha_{\mathbf{r},c} \leq 1$ im Gegensatz zum Crisp-Fall und unter der Normalisierungsbedingung $\sum_{c=1}^{N_\mathcal{L}} \alpha_{\mathbf{r},c} = 1$. Die Prototypklassenbezeichner werden automatisch während des Trainings adaptiert. Dabei ändert sich die Lerndynamik des Standard SNPC. Allerdings kann eine ähnliche Lerndynamik hergeleitet werden

$$\triangle \mathbf{w}_\mathbf{r} = -\frac{T}{2\tau^2} \cdot \frac{\partial d_\mathbf{r}}{\partial \mathbf{w}_\mathbf{r}} \quad T = u_\tau(\mathbf{r}|\mathbf{v}_k) \cdot \left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}} - lc(\mathbf{v}_k, c_{\mathbf{v}_k})\right) \tag{3}$$

Parallel können die fuzzy labels $\alpha_{\mathbf{r},c_{\mathbf{v}_k}}$ optimiert werden $\frac{\partial lc(\mathbf{v}_k, c_{\mathbf{v}_k})}{\partial \alpha_{\mathbf{r},c_{\mathbf{v}_k}}}$: $\triangle \alpha_{\mathbf{r},c_{\mathbf{v}_k}} = -u_\tau(\mathbf{r}|\mathbf{v}_k)$ gefolgt von einer Normalisierung. Im SNPC wird eine sogenannte Fensterregel verwendet, um den Algorithmus stabil zu halten. Wie in [11] gezeigt wurde kann auch für den FSNPC eine ähnliche Fensterregel abgeleitet werden. Auch hier kann Relevanzlernen integriert werden [13]. Durch die Verwendung von Fuzzy-Labeln werden nun auch die

Label dynamisch adaptiert und zudem werden schlecht gelernte Rezeptive Felder durch unsichere Klassenentscheidungen offenbar. Beide Aspekte sind hilfreich im Kontext der klinischen Proteomik, da einerseits die Anzahl der Prototypen pro Klasse nicht vorgegeben werden muss und zum anderen Zuordnungen von Datenpunkten zu überlappenden Bereichen identifiziert werden können.

3.3 Aktive Lernstrategien

Die Margin Analyse aus [2], [3] und [11] motiviert elegante Schemata für aktive Lernstrategien, die zur Konvergenzverbesserung dienen. Offensichtlich, hängt die Generalisierungsfähigkeit von GRLVQ-Algorithmen lediglich von Punkten mit zu kleinem Margin ab. Somit müssen lediglich, extreme Marginwerte beschränkt werden und eine Einschränkung entsprechender Updates sollte nur für extreme Paare von Prototypen erfolgen. Diese Argumentation führt zur Entwicklung von aktiven Lernschemata wenn eine gegebene statische Menge von Trainingsdaten vorliegt. Dazu definiert man eine monoton abfallende, nicht-negative Selektionsfunktion (0 - keine Selektion, 1 - garantierte Selektion) $L^c : \mathbb{R} \to \mathbb{R}$ und selektiert aktiv Trainingsdatenpunkte von einer gegebenen Beispieldatenmenge (Details und positive Ergebnisse in [11]).

4 Anwendungen

Die vorgestellten Methoden wurden auf klinischen Proteomdaten (Proteom I, Proteom II, WDBC aus [11],[1]) angewandt und mit Standardverfahren (Support Vector Machine (SVM), Diskriminanzanalyse (LDA /QDA) sowie GLVQ und SNG) verglichen (Details in [11],[7], Ergebnisse in Tabelle1). Nahezu alle Klassifikatoren zeigen gute Ergebnisse von über $> 90\%$ für den WDBC Datensatz [1]. Der Proteom$_1$ Datensatz ist der komplizierteste Datensatz und wird am besten durch einen Klassifikator mit Relevanzlernen modelliert. Der Proteom$_2$ Datensatz wird durch mehrere Klassifikatoren korrekt modelliert allerdings, führt auch hier Relevanzlernen zu einer weiteren Verbesserung. Klassifikationsverfahren mit Fuzzy-Labeling zeigen ähnlich gute Ergebnisse führen aber zu einer einfacheren Modellgenerierung im Bezug auf die Netzwerkstruktur. Die lokale Metrikadaptation hat für die betrachteten Datensätze nur leichte Vorteile.

Die abschliessende Analyse zeigt, dass die durchgeführten Erweiterungen erfolgreich für reale Datensätze aus der klinischen Proteomik einsätzbar sind. Dabei ist Metrikadaptation nützlich, um potentielle Biomarkerkandidaten zu identifizieren sowie, um die Vorhersagegenauigkeit zu verbessern. Lokales Relevanzlernen verbesserte die Ergebnisse im allgemeinen nicht, aber gestattete eine spezifischere Interpretation der relevanten Peaks. Der FSNPC Algorithmus gestattet Sicherheitsbewertungen der Klassifikationsentscheidungen und verbesserte die Vorhersagegenauigkeit. Dies ist vorallem auf eine flexiblere Modellierung des Klassifikators zurückzuführen, da die Klassenbezeichner der Prototypen nicht länger fest vorgegeben sind und somit nur die allgemeine Anzahl der Prototypen spezifiziert werden muss. Die Tanimoto- und Mahalanobisdistanz sowie LDA und QDA sind für Daten der klinischen Proteomik nur nach geeigneter Vorselektion der Merkmale anwendbar. Im Vergleich zum bekannten SVM-Algorithmus sind die Ergebnis bei Verwendung von Prototypenverfahren ähnlich gut oder zum Teil auch besser, aber mit dem zusätzlichen Vorteil verbesserter Interpretierbarkeit.

References

[1] C. Blake and C. Merz. UCI repository of machine learning databases., (last visit 01.11.2006). available at: http://www.ics.uci.edu/ mlearn/MLRepository.html.

[2] K. Crammer, R. Gilad-Bachrach, A.Navot, and A.Tishby. Margin analysis of the lvq algorithm. In *Proc. NIPS 2002* http://www.nips.cc (last visit 01.05.2007).

[3] B. Hammer, M. Strickert, and Th. Villmann. On the generalization ability of GRLVQ networks. *NPL*, 21(2):109–120, April 2005.

[4] B. Hammer, M. Strickert, and Th. Villmann. Supervised neural gas with general similarity measure. *NPL*, 21(1):21–44, 2005.

[5] B. Hammer and Th. Villmann. Generalized relevance learning vector quantization. *Neural Netw.*, 15(8-9):1059–1068, 2002.

[6] B. Hammer and Th. Villmann. Mathematical aspects of neural networks. In M. Verleysen, editor, *ESANN'2003*, pages 59–72, Brussels, Belgium, 2003. d-side.

[7] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

[8] Teuvo Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Heidelberg, 1995. (2nd Ext. Ed. 1997).

[9] Daniel C. Liebler. *Introduction to Proteomics*. Humana Press, 2002.

[10] A. S. Sato and K. Yamada. Generalized learning vector quantization. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in NIPS*, volume 7, pages 423–429. MIT Press, 1995.

[11] Frank-Michael Schleif. *Prototype based Machine Learning for Clinical Proteomics*. Technical University Clausthal, 2006. Dissertation.

[12] S. Seo, M. Bode, and K. Obermayer. Soft nearest prototype classification. *IEEE TNN*, 14:390–398, 2003.

[13] T. Villmann, F.-M. Schleif, and B. Hammer. Prototype-based fuzzy classification with local relevance for proteomics. *Neurocomp. Letters*, pages 2425–2428, 2006.

Kontakt

Dr. Frank-Michael Schleif
Universität Leipzig
Medizinische Fakultät
AG Computational Intelligence
Karl-Tauchnitzstr. 25,04107 Leipzig
Tel.: +49 (0)341-9718896
Fax : +49 (0)341-9718849
Email: schleif@informatik.uni-leipzig.de

Bild

Frank-Michael Schleif studierte bis 2002 Angewandte Informatik an der Universität Leipzig und war dort nachfolgend als wissenschaftlicher Mitarbeiter tätig. Ab 2003 begann er eine Industriepromotion bei der Bruker Bioscience Corp., mit Betreuung an der TU Clausthal, die 2006 als beste Informatik-Promotion der TU Clausthal zum GI-Dissertationspreis vorgeschlagen wurde. Während der Arbeit bei Bruker war er in verschiedenen Forschungsprojekten und der Software-Entwicklung für den Bereich klinische Proteomik tätig. Seit Dezember 2006 arbeitet er an der Universität Leipzig in der AG Computational Intelligence in Projekten der Datenanalyse. Seine Forschungsinteressen liegen im Bereich Mustererkennung, statistische Datenanalyse und Algorithmen Entwicklung.