X. Optimales Lernen und Verallgemeinern II

X.1 Kombinationen von neuronalen Netzwerken

Die Kombination verschiedener Modelle ist eine pragmatische und konzeptionell sehr einfache Methode zur Verbesserung der Modellierungs-, Klassifizierungs- oder Prognosegenauigkeit. Eine Verbesserung ergibt sich allerdings nur dann, wenn die verschiedenen Modelle auch unterschiedlich verallgemeinern, d.h. wenn nicht alle Modelle für die gleichen Testbeispiele schlechte bzw. gute Resultate liefern.

Neuronale Netzwerke eignen sich besonders gut zur Erzeugung von Modellen, die unterschiedlich verallgemeinern:

- Verwendung verschiedener Typen von neuronalen Netzwerken
- Variation der Anzahl Hidden Units
- Verschiedene Anfangsgewichte
- Verschiedene Trainingssets (z.B. durch "Resampling")
- etc.

X.1.1 Kontinuierliche Outputs

$x(\nu)$ = gewünschter Output für Testbsp. ν $[\nu = 1, \ldots, M]$

$x_i(\nu)$ = Output des Netzwerks i $[i = 1, \ldots, N]$

Mittlerer quadr. Outputfehler des Netzwerks i:

$$\sigma_i^2 = \frac{1}{M} \sum_{\nu=1}^{M} [x_i(\nu) - x(\nu)]^2 \equiv \langle (x_i - x)^2 \rangle$$

Einfachste Kombinationsregel:

$$y(\nu) = \frac{1}{N} \sum_{i=1}^{N} x_i(\nu) \qquad [\text{arithm. Mittelwert}]$$

Für den mittleren quadr. Outputfehler des kombinierten Systems ergibt sich dann:

$$\sigma_y^2 = \langle (y-x)^2 \rangle = \frac{1}{N^2} \left[\sum_{i=1}^{N} \sigma_i^2 + 2 \sum_{i>j} \rho_{ij} \sigma_i \sigma_j \right] ,$$

wobei

$$\rho_{ij} = \frac{\langle (x_i - x)(x_j - x) \rangle}{\sigma_i \sigma_j} \qquad \text{[Korrelationskoeffizienten]}$$

Da die Korrelationskoeffizienten ρ_{ij} zwischen -1 und $+1$ variieren, folgt

$$\sigma_y \leq \frac{1}{N} \sum_{i=1}^{N} \sigma_i ,$$

d.h. der mittlere Outputfehler des kombinierten Systems ist immer kleiner als der durchschnittliche Outputfehler der einzelnen Netzwerke (vorausgesetzt dass diese nicht alle perfekt korreliert sind, d.h. dass nicht alle $\rho_{ij} = +1$ sind).

σ_y^2 lässt sich auch wie folgt ausdrücken:

$$\sigma_y^2 = \frac{1}{N} \sum_{i=1}^{N} \sigma_i^2 - \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2$$

wobei

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

Diese Relation zeigt, dass der mittlere quadr. Outputfehler des kombinierten Systems umso kleiner wird, je grösser die Diskrepanz zwischen den einzelnen Netzwerken ist (vorausgesetzt, dass ihr mittlerer Fehler nicht entsprechend ansteigt).

Bootstrapping:

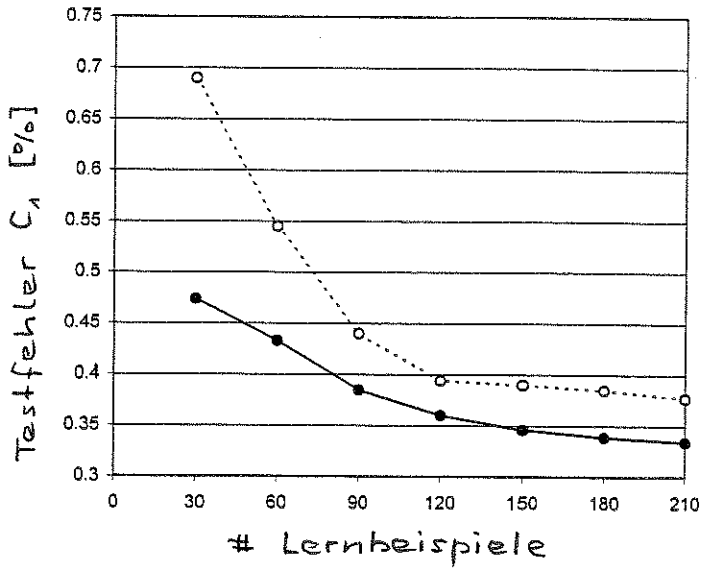
Eine der bekanntesten "Resampling"-Methoden ist Bootstrapping. Dabei werden neue Lernsets generiert, indem N Lernbeispiele zufällig aus den N zur Verfügung stehenden Beispielen ausgewählt werden (Wiederholungen erlaubt!).

Die verschiedenen Trainings-Sets haben also alle die gleiche Anzahl von Lernbeispielen wie der ursprüngliche Lernset, aber einige der Lernbeispiele fehlen und andere kommen mehrfach vor.

Anwendungsbeispiel:

Vorhersage der Konzentrationen von Komponenten einer Reinigungsflüssigkeit aus dem Infrarot-Spektrum.

[A. Ukil, J. Bernasconi, H. Brändle, ABB-interner Bericht]



····o···· Einzelnes neuronales Netzwerk

—●— Kombination (Mittelwert) von 70 Netzwerken (70 Lernsets mit Bootstrapping erzeugt)

X.1.2 Binäre Outputs (z.B. Klassifizierung)

$\longrightarrow \quad x(\nu) = \pm 1 \quad , \quad x_i(\nu) = \pm 1$

p_i = Wahrsch.keit, dass Netzwerk i den korrekten Output liefert ["Hitrate"]

p_{ij} = Wahrsch.keit, dass sowohl Netzwerk i als auch Netzwerk j den korrekten Output liefern

etc.

Einfachste Kombinationsregel:

$$y(\nu) = \text{sign}\left[\sum_{i=1}^{N} x_i(\nu)\right] \qquad \text{[Mehrheitsentscheid]}$$

Für $N = 3$ kann die Hitrate p_y des Mehrheitsentscheids wie folgt ausgedrückt werden:

$$p_y = p_{12} + p_{13} + p_{23} - 2\,p_{123} \; .$$

(p_y kann für beliebiges ungerades N explizit berechnet werden!)

Beispiel 1:

$p_1 = p_2 = p_3 = 0.6$

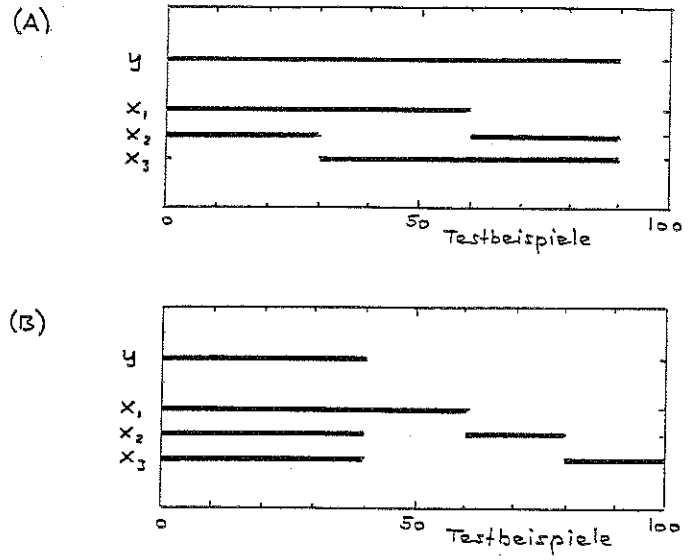
(A) $p_{12} = p_{13} = p_{23} = 0.3 \;, \; p_{123} = 0 \longrightarrow p_y = 0.9$

[beste Situation]

(B) $p_{12} = p_{13} = p_{23} = 0.4 \;, \; p_{123} = 0.4 \longrightarrow p_y = 0.4$

[schlechteste Situation]

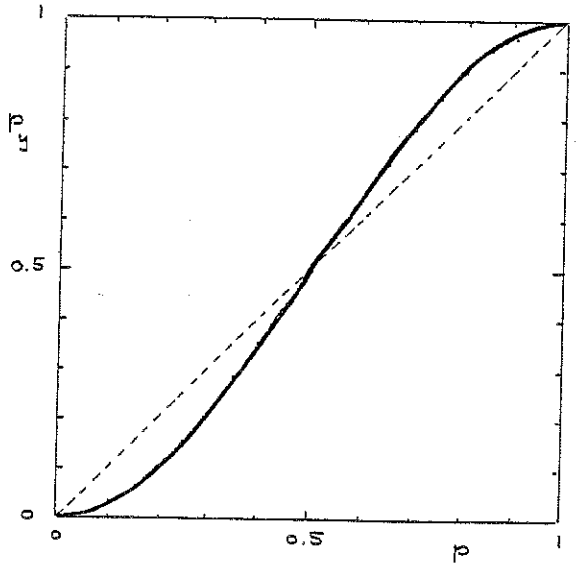
Illustration von Beispiel 1:



Beispiel 2:

x_1, x_2, x_3 unkorreliert $\longrightarrow$ $p_{12} = p_1 p_2$, $p_{123} = p_1 p_2 p_3$

$p_1 = p_2 = p_3 = p$ $\longrightarrow$ $p_y = 3p^2 - 2p^3$



Bemerkung:

p_y lässt sich auch schreiben als

$$p_y = p_{12} + q_3 (p_1 + p_2 - 2p_{12}) \quad *)$$

wobei

q_3 = Wahrsch.keit, dass x_3 korrekt ist, wenn x_1 und x_2 nicht übereinstimmen

$\longrightarrow$ Idee des Boosting-Algorithmus! (siehe S. X/6)

Der Boosting Algorithmus

[R. Schapire, Machine Learning 5, No. 2, 1990, pp. 197-227]
[H. Drucker et al., Int. J. of Pattern Recognition and Artif. Intelligence 7, No. 4, 1993, pp. 705-719]

1) Ein neuronales Netz wird mit M Lernbeispielen trainiert.

2) Mit Hilfe des trainierten Netzes wird ein zweites Lernset mit M Lernbeispielen gebildet, und zwar so, dass 50% der Beispiele vom 1. Netz richtig und 50% falsch klassifiziert werden.

3) Mit diesem Lernset wird ein zweites Netz trainiert.

4) Mit Hilfe der beiden Netze wird ein neues Lernset mit M Beispielen gebildet, welches nur solche Beispiele enthält, die von Netz 1 und Netz 2 unterschiedlich klassifiziert werden.

5) Mit diesem Lernset wird ein drittes Netz trainiert.

6) Die 3 neuronalen Netze werden gemäss Mehrheitsentscheid kombiniert.

Bei der Bildung des zweiten und dritten Lernsets fallen viele der vorhandenen Lernbeispiele weg, vor allem wenn die Erkennungsraten der Netze 1 und 2 hoch sind. Deshalb müssen geeignete Methoden zur Erzeugung von neuen Lernbeispielen verwendet werden.

Wenn die Netze 1 und 2 unabhängig sind und die gleiche Fehlerrate $e = 1-p$ haben, dann erhalten wir aus *) den folgenden Ausdruck für die Fehlerrate e_3 des kombinierten Systems:

$$e_3 = e\left[2 - e - 2q_3(1-e)\right] \quad ,$$

d.h. $e_3 < e$ falls $q_3 > \frac{1}{2}$, d.h. falls das dritte Netz mehr als die Hälfte der Beispiele richtig klassifiziert, bei denen die Netze 1 und 2 nicht einig sind.

ANWENDUNGSBEISPIELE:

(i) Interceptor Condition Problem LIC1

[D. Partridge and W.B. Yates
Neural Computation 8, pp. 869-893 (1996)]

Inputs: $I_i \quad (i = 1, \ldots, 5) \quad , \quad 0 \leq I_i \leq 1$

Output: $O = \begin{cases} 1 & \text{falls } \sqrt{(I_1 - I_2)^2 + (I_3 - I_4)^2} \leq I_5 \\ 0 & \text{sonst} \end{cases}$

5 neuronale Netze:

$$\langle p \rangle = 95.34\,\%$$

$$\langle p_y(3) \rangle = 98.49\,\%$$

$$\langle p_y(5) \rangle = 99.02\,\%$$

(ii) Vorhersage von Devisenkursänderungen [USD/DEM]

[J. Bernasconi and B. Bachmann
Proc. 3rd Int. Conf. on "Forecasting Financial Markets",
London, 27-29 March 1996]

5 neuronale Netzwerke:

	Hitrate	Profit
(a)	52.26 %	9.36 %
(b)	53.62 %	14.14 %
(c)	55.75 %	22.21 %

(a) Einzelvorhersagen
(b) Mehrheit von 3 Netzwerken
(c) Mehrheit aller 5 Netzwerke

X.2 ACTIVE LEARNING

Bei praktischen (z.B. industriellen) Problemen müssen die Lernbeispiele zum Modellieren einer Input/Output-Relation $y = f(\underline{x})$ oft mit Hilfe von aufwendigen, d.h. teuren Experimenten erzeugt werden. Dann ist man daran interessiert, bzw. darauf angewiesen, die notwendige Anzahl von Experimenten zu minimieren.

→ Active Learning (Active Data Selection):

Bestimmung der Inputwerte $\underline{x}$, bei denen das nächste Experiment durchgeführt werden soll, so dass der erwartete Informationsgewinn über den Zusammenhang $y = f(\underline{x})$ maximal wird.

Problemstellung:

- Modellierung des Zusammenhangs zwischen y und $\underline{x}$ mit einem neuronalen Netzwerk:

  $$y = f(\underline{x}\,;\,\underline{W})$$

- N Experimente durchgeführt
  → Lernbeispiele $\{\underline{x}_k, y_k\}$, $k = 1, \ldots, N$

- Bei welchem $\underline{x}$ soll das nächste Experiment durchgeführt werden?

X.2.1 "Active Learning" - Strategien

Space-Filling:

Das nächste Experiment wird bei demjenigen Input $\underline{x}^*$ durchgeführt, der am weitesten von den bereits vorhandenen Lernbeispielen entfernt ist (im interessierenden $\underline{x}$-Bereich):

$$\underline{x}^* = \operatorname{argmax}_{\underline{x}} d(\underline{x}) \;, \quad d(\underline{x}) = \min_k \| \underline{x} - \underline{x}_k \|$$

→ Sehr einfache Strategie, aber nicht immer besonders effizient.

Maximaler Informationsgewinn:

= maximale Entropieabnahme (in Bezug auf die Unsicherheit der Netzwerk-Gewichte)

$$\longrightarrow \quad \Delta S(\underline{x}) = \int dy \; \rho_N(y|\underline{x}) \left[S_N - S_{N+1}(\{\underline{x}, y\}) \right] = \max$$

$$S_N = - \int d\underline{W} \; \rho_N(\underline{W}) \log \rho_N(\underline{W})$$

$$S_{N+1}(\{\underline{x}, y\}) = - \int d\underline{W} \; \rho_{N+1}(\underline{W}|\{\underline{x}, y\}) \log \rho_{N+1}(\underline{W}|\{\underline{x}, y\})$$

$$\rho_N(y|\underline{x}) = \int d\underline{W} \; \rho_N(\underline{W}) \, \rho(y|\underline{x}; \underline{W})$$

$\rho_N(\underline{W})$ = Wahrsch.dichte der $\underline{W}$ nach N Experimenten

$\rho(y|\underline{x}; \underline{W})$ = Wahrsch.dichte, dass das neuronale Netz beim Input $\underline{x}$ den Output y vorhersagt

→ Strategie theoretisch fundiert, aber $\Delta S(\underline{x})$ nur für einfache Modelle effizient berechenbar.

Maximale Modellierungsunsicherheit:

Approximative Berechnung von $\Delta S(\underline{x})$:

$$\rightarrow \Delta S(\underline{x}) \approx \frac{1}{2} \log \left[1 + \frac{1}{\sigma^2} U(\underline{x}; \underline{W}_N) \right]$$

(unter der Annahme, dass das Modell $y = f(\underline{x}; \underline{W})$ die experimentellen Daten bis auf einen zufälligen Fehler σ darstellen kann)

$U(\underline{x}; \underline{W}_N)$ misst die Modellierungsunsicherheit nach N Experimenten und kann z.B. empirisch wie folgt abgeschätzt werden:

- Trainiere n Netzwerke mit den vorhandenen N Lernbeispielen (z.B. durch Resampling der Lerndaten)
- Bestimme die Varianz der verschiedenen Modelle als Funktion des Inputs $\underline{x}$:

$$s^2(\underline{x}) = \frac{1}{n} \sum_{i=1}^{n} \left[y_i(\underline{x}) - \bar{y}(\underline{x}) \right]^2$$

Das nächste Experiment wird dann bei demjenigen Input $\underline{x}$ durchgeführt, bei dem $s^2(\underline{x})$ am grössten ist.

In verschiedenen Tests hat diese Strategie zu sehr guten Resultaten geführt (z.B. [4]).

$\rightarrow$ Wesentlich schnellere Abnahme des Modellierungsfehlers (als Funktion der Anzahl Lernbeispiele) als bei zufälligen oder Space-Filling Strategien.

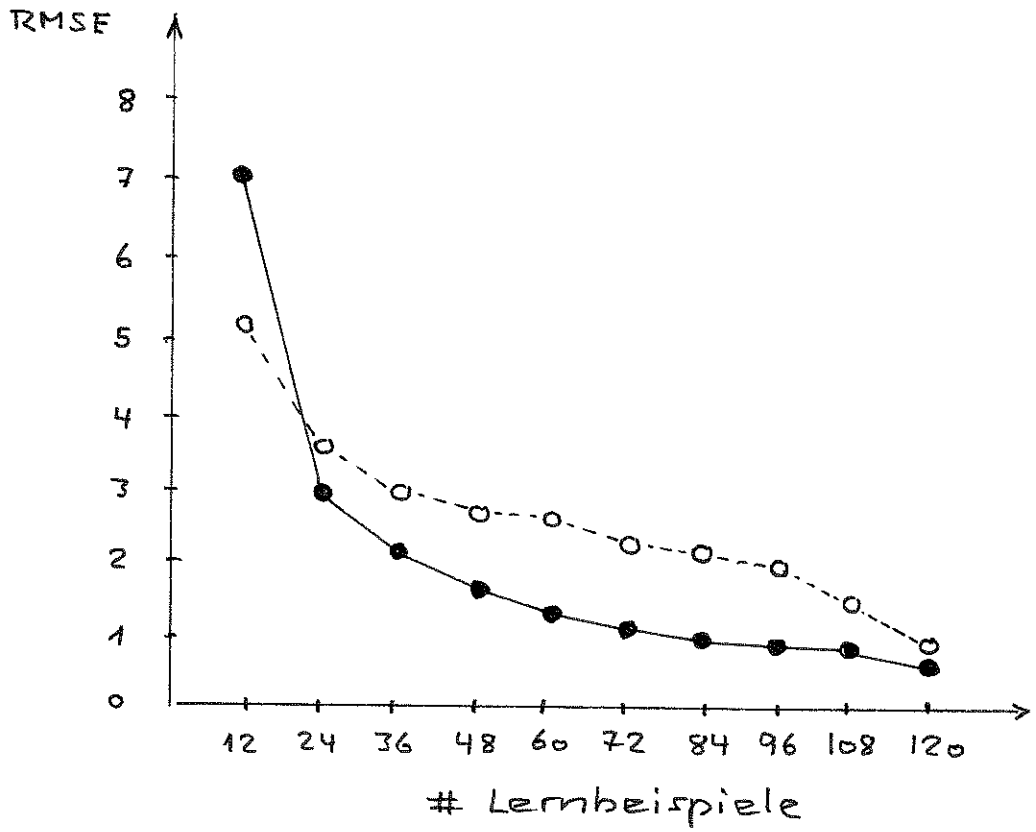
X.2.2 Anwendungsbeispiel:

Vorhersage des Fettgehalts von Fleischproben aus dem Infrarot-Absorptionsspektrum

[Sem.arbeit Philippe Bourquin, WS 2004/05]

→ Modellierung mit Multilayer-Perceptron:

- Inputs = 12 wichtigste Principal Components des Infrarot-Spektrums
- Output = Fettgehalt [%]



RMSE = Root Mean Squared Error für Fettgehalt [%]

---o--- zufällige Wahl der nächsten Lernbeispiele

—●— Max. Modellierungsunsicherheit eines Ensembles von neuronalen Netzen

X.2.3 Literatur zu Active Learning

[1] J. Bernasconi and F. Greuter
Adaptive Design of Experiments
Informatik - Informatique 1, 1998, pp. 18-20

[2] D. J. C. MacKay
Information-Based Objective Functions for Active Data Selection
Neural Computation 4, 1992, pp. 590-604

[3] A. Krogh and J. Vedelsby
Neural Network Ensembles, Cross-Validation, and Active Learning
In: G. Tesauro et al, eds., Advances in Neural Information Processing Systems 7, pp. 231-238, MIT Press, 1997

[4] J. Poland and A. Zell
Different Criteria for Active Learning in Neural Networks: A Comparative Study
ESANN 2002 Proceedings - European Symposium on Artificial Neural Networks, pp. 119-124