

Abschätzung der Vertrauenswürdigkeit von Neuronalen Netzprognosen bei der Prozessoptimierung

Prof. Dr.-Ing. Peter Protzel⁺
Dipl.-Phys. Lars Kindermann⁺⁺
Dipl.-Inform. Michael Tagscherer⁺⁺
Dr. Achim Lewandowski⁺⁺

⁺Technische Universität Chemnitz, Institut für Automatisierung, 09107 Chemnitz
Tel. 0371 - 531 3442, peter.protzel@e-technik.tu-chemnitz.de

⁺⁺Bayerisches Forschungszentrum für Wissensbasierte Systeme (FORWISS)
Am Weichselgarten 7, 91058 Erlangen, www.forwiss.de/aknn

Zusammenfassung¹

Neuronale Netze als universale Funktionsapproximatoren liefern zu jeder beliebigen Kombination von Eingangsgrößen stets ein Resultat, unabhängig davon, wie erfolgreich das Training war. Auch wenn die aktuellen Eingangsgrößen in den vorhandenen Trainingsdaten überhaupt keine Repräsentation haben, wird das Netz ein beliebiges, bestenfalls zweifelhaftes Ergebnis anzeigen. Wir stellen einfache Methoden vor, die neben der Netzprognose eine parallele Ausgabe von zu erwartendem Fehler und der Vertrauenswürdigkeit liefern.

Einleitung

Eines der Haupteinsatzgebiete von Neuronalen Netzen ist die datengetriebene Modellbildung (Identifikation) von nichtlinearen Prozessen, für die kein analytisches Modell zur Verfügung steht. Mit Hilfe eines solchen Neuro-Modells kann dann z.B. die Reaktion eines Prozesses auf verschiedene Stellgrößen simuliert und vorab eine optimale Stellgrößensequenz ermittelt werden (Modellbasierte Prädiktive Regelung). Offensichtlich hängt die Güte einer solchen Prozessoptimierung entscheidend von der Güte des Modells ab. Die Probleme und Lösungsansätze, die sich hierbei durch den „Black-Box Charakter“ von Neuronalen Netzen ergeben, werden im folgenden am Beispiel von Multilayer Perzeptrons erörtert.

Multilayer Perzeptrons sind in der Anwendung wegen ihrer Robustheit und einfachen Implementierbarkeit einer der meistverwendeten Netzwerktypen [1]. MLPs liefern für beliebige Werte an den Eingängen ein Ergebnis, ob sinnvoll oder nicht. Außerhalb des Arbeitsbereiches, für den Trainingsdaten vorliegen, wird diese Netzprognose in der Regel jedoch falsch sein. Im Gegensatz zu mathematisch - physikalischen Modellen, bei denen man normalerweise eine Plausibilitätskontrolle und Fehlerrechnung mit implementiert, fehlen entsprechende Kontrollen bei Neuronalen Modellen meist. Man beschränkt sich bei der Gütebewertung auf die Angabe des mittleren Fehlers auf dem Trainings-, Test- oder Validierungsset, was jedoch nichts über die Vertrauenswürdigkeit eines einzelnen Ausgabewerts für einen bestimmten Eingangsvektor aussagt. Fehlerhafte Netzprognosen können verschiedene Ursachen haben, darunter:

1. Der Ausgangswert ist prinzipiell nicht aus den vorhandenen Eingangsdaten vorhersagbar

¹Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung, Wissenschaft, Forschung und Technologie unter dem Förderkennzeichen 01IN505B gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

wegen verrauschter, fehlender oder widersprüchlicher Trainingsdaten.

2. Das Netz ist nicht korrekt trainiert oder ist in seiner Struktur nicht geeignet, die Daten zu repräsentieren.
3. Das vorhandene Trainingsdatenmaterial repräsentiert nicht die aktuelle Prognoseaufgabe, z.B. weil in den Trainingsdaten kein ähnlicher Fall vorhanden ist.

Alle diese Fehlerquellen können vom aktuellen Arbeitspunkt abhängen, Punkt drei mit Sicherheit. Also wäre es für den Anwender wünschenswert, Informationen über die zu erwartende Genauigkeit zu erhalten, die über den mittleren Fehler auf einem ganzen Datensatz hinausgehen. Für jeden Eingangswert sollte ein System nicht nur den modellierten Ausgangswert sondern auch ein Vertrauensmaß für die Gültigkeit/Verlässlichkeit berechnen können. Im Folgenden werden einfache Methoden vorgestellt, ein solches Vertrauensmaß für jeden Eingangswert zu ermitteln.

Vertrauenswürdigkeit von MLPs

Das Problem, den Fehler, den ein Modell macht, vorherzusagen, ist sehr ähnlich dem Problem das System selbst vorherzusagen. Eine Modellierung des vorzeichenbehafteten Fehlers ist natürlich nicht sinnvoll: Falls diese nicht Null ergäbe, wäre ja genausogut eine bessere Modellierung des Systems möglich. Allerdings wird auch ein optimal angepasstes Modell immer noch Fehler machen, die um Null herum verteilt sind. Dieser wird gewöhnlich absolut gemittelt über eine größere Testdatenmenge als Netzgüte angegeben. Um darüberhinausgehende lokalisierte Abschätzungen zu erhalten, wird ein zweites Netz, das die gleiche Struktur haben kann, mit den *gleichen Eingangswerten* aber dem absoluten oder quadratischen Fehler des Prognosenetzes als Zielfunktion trainiert. Um zu kleine Fehlerangaben durch Überanpassung auszuschließen, sollte dieses auf einem zweiten Validierungsdatensatz durchgeführt werden. Dieses Netz kann dann parallel zum ersten betrieben werden und liefert für jeden Eingangswert den mittleren zu erwartenden Fehler, abhängig von der Lage im Eingangsraum.

Allerdings versagt dieses Netz genauso für Eingangswerte, die außerhalb des Trainingsdatenbereiches liegen, da ja hier keinerlei Vorwissen vorhanden ist, auch nicht über die Fehler, die gemacht werden können. Diese dritte Fehlerquelle kann durch die Kenntnis der statistischen Verteilung der Eingangsdaten abgeschätzt werden [2,3]. Dieses kann ebenfalls durch ein weiteres, strukturell gleiches Netz erreicht werden. Dazu trainiert man es wiederum mit den *gleichen Eingängen*, jedoch mit einer Konstanten „1“ als Zielfunktion. Zusätzlich erzeugt man sich zufällige Eingänge aus dem gesamten zu erwartenden Arbeitsbereich und trainiert für diese mit dem Zielwert „0“. Dadurch ist dieses Netz in der Lage, die Verteilungsfunktion der Eingangswerte der Trainingsdaten zu modellieren: Es wird überall in Richtung Null gezogen, nur an Orten mit Trainingsdaten zur Eins hin. Durch Mittelung wird somit für jeden Bereich das Verhältnis von Zufallsvektoren und Trainingsdaten modelliert. Ein Ausgabewert nahe Null bedeutet also, dass man sich in unbekanntem Terrain bewegt und die Prognose des Prozessmodells nicht durch antrainiertes Vorwissen abgesichert ist. In den Abbildungen 1-5 wird dies anhand eines beispielhaften Funktionsverlaufes demonstriert [4].

Es hat sich als günstig herausgestellt, etwa 10-mal so viele Null-Vektoren zu erzeugen, wie Trainingsdaten vorhanden sind. Der Bereich sollte für jede Eingangsdimension klar über die maximal und minimal zu erwartenden Werte hinausgehen, um am Rand den Ausgang auf Null zu zwingen.

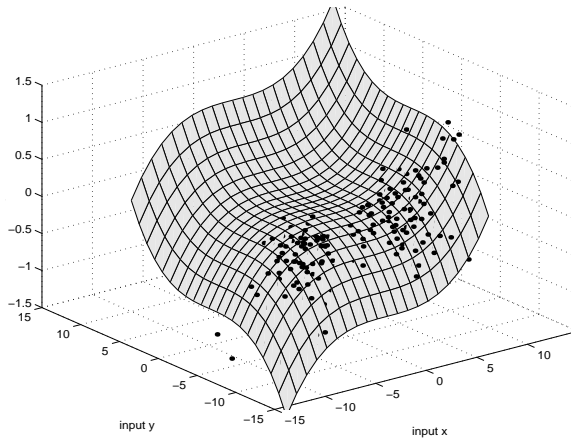


Abbildung 1: Ein Kennfeld $z = f(x,y)$ wird an diskreten Punkten (x,y) abgetastet, die einer nicht gleichmäßigen Verteilung entstammen. Diese dienen dann als Trainingsdaten für ein MLP.

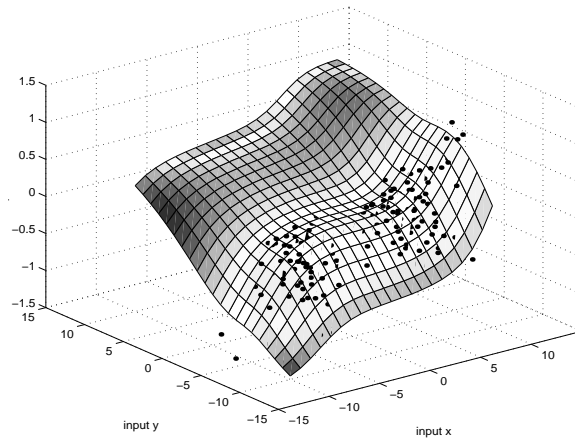


Abbildung 2: Nachdem das Netz mit diesen Werten trainiert wurde, liefert es dieses Kennfeld $f'(x,y)$: In Gebieten mit vielen Trainingspunkten ist die Approximation genau (hell), in leeren Gebieten macht es Fehler (dunkel).

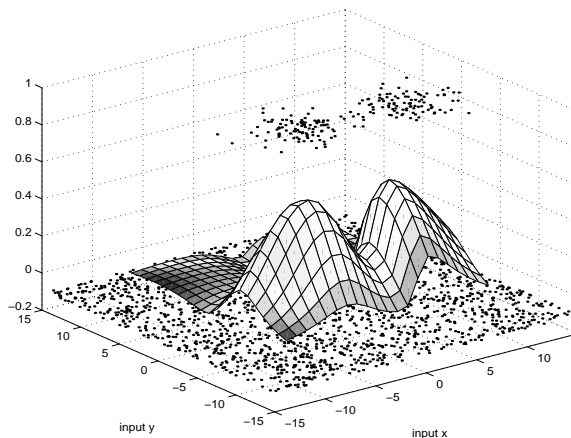


Abbildung 3 Ein anderes Netzwerk mit gleicher Topologie wurde mit den selben Eingangswerten (x,y) aber konstantem Target $z=1$ sowie zufällig generierten Inputs mit dem Target $z=0$ trainiert. Dieses Netz hat gelernt, die Verteilung der Trainingspunkte im Eingangsraum abzubilden.

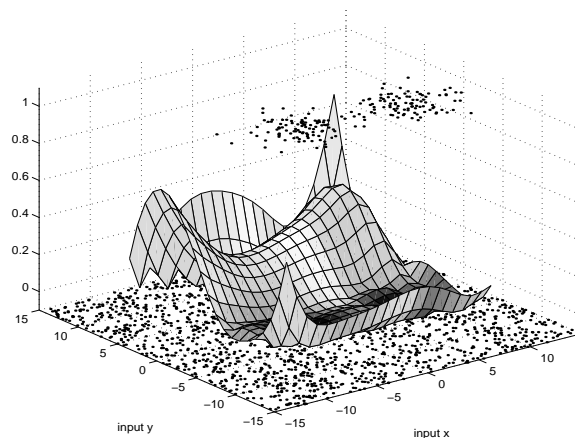


Abbildung 4 Der Absolutfehler $|f - f'|$ der Kennlinienapproximation. Dunkle Regionen markieren Gebiete mit hoher Datendichte, wie sie das zweite Netz erkannt hat.

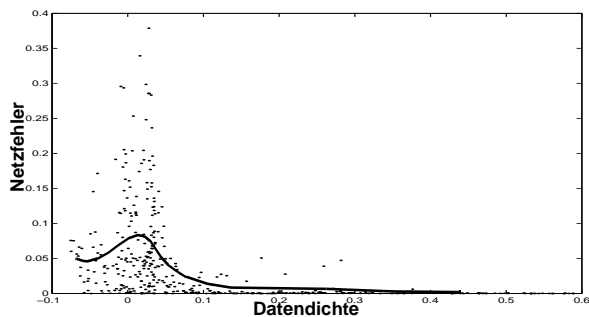


Abbildung 5: Auf einem Testdatensatz erkennt man erkennt deutlich den Zusammenhang zwischen dem Vorhersagefehler für das Kennfeld an einem Punkt im Arbeitsraum und der Trainingsdatendichte an dieser Stelle. Das Dichteanimationsnetz kann somit zur Berechnung einer Vertrauenskenngroße verwendet werden.

Anwendung bei der Optimierung von Prozesseinstellungen

Ein verbreiteter Anwendungsbereich für Neuronale Prozessmodelle ist die Optimierung von einstellbaren Parametern. Dabei werden durch Simulation am Modell die variierbaren Einstellungen gesucht, die bei gegebenen fixen Eingangsgrößen eine ebenfalls gegebene Zielgröße bestmöglich annähern oder den Ausgangswert maximieren oder minimieren. Dazu variiert ein Optimierungsalgorithmus die variablen Eingänge auf der Suche nach dem bestmöglichen Ausgang. Ein großes Problem dabei ist, dass bei Neuronalen Netzen Extremwerte oft weit außerhalb des Trainingsbereiches liegen. Benutzt man z.B. ein Gradientenabstiegsverfahren zur Einstellung der Eingänge, wird es oft auf Werte weit außerhalb des Bereiches konvergieren, der durch Trainingsdaten erfasst ist. Man muss manuell zusätzliche Randbedingungen und Einschränkungen einführen, um sinnvolle Ergebnisse zu erhalten.

Durch die Einbeziehung der beschriebenen Fehlernetze in die Optimierung (Abbildung 6) kann dieses Problem entschärft werden:

1. Es sind keine Schritte erlaubt, die in Bereiche führen, in denen die Ausgabe des „Verteilungsnetzes“ (NN3) auf nahe Null zurückgeht, also ins „Terra Incognita“. Dieses entspricht der automatischen Einschränkung auf den Bereich der vorhandenen Trainingsdaten.
2. Schritte in Richtungen, in denen zwar das Ergebnis des Modells besser wird, jedoch der Fehler stark steigt, werden schwächer berücksichtigt als Änderungen deren Auswirkung „sicher“ bekannt ist. Um die mittlere Abweichung vom Zielwert so klein wie möglich zu halten, muss man sogar die Summe aus quadratischer Abweichung von Ziel- und Istwert aus NN1 plus den erwarteten quadratischen Fehler aus NN2 minimieren.

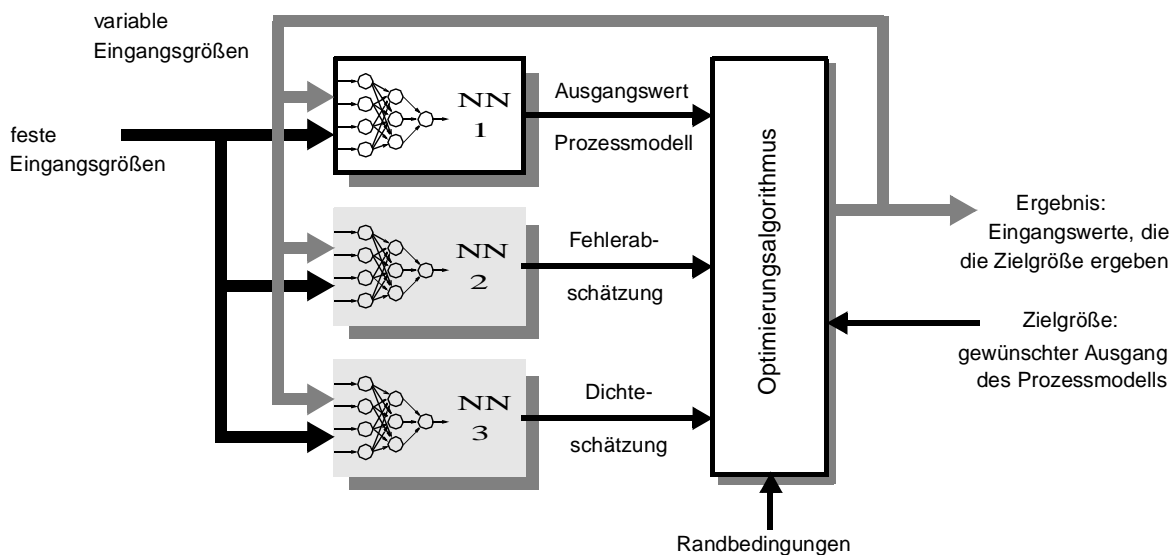


Abbildung 6: Die Kombination von 3 strukturell gleichen Netzen ermöglicht einem Optimierungsalgorithmus bei seiner Suche nach optimalen variablen Parametern zur Erreichung einer Zielgröße, Einstellungen zu vermeiden, für die das Prozessmodell NN1 nicht vertrauenswürdig ist: Das Modell macht große Fehler (NN2) oder hat noch keine entsprechenden Trainingsdaten gesehen (NN3). Dazu wird NN2 mit dem Absolut- oder quadratischen Fehler von NN1 trainiert und NN3 zur Approximation der Datendichte im Eingangsraum der Trainingsdaten verwendet.

Diskussion

Die hier vorgeschlagene Methode zur Fehlermodellierung und Plausibilitätskontrolle von MLPs ermöglicht bei sehr geringem zusätzlichem Aufwand (man verwendet ja die gleiche Netzstruktur 3-fach parallel) häufig auftretende Probleme im industriellen Einsatz zu erkennen und zu vermeiden.

Bisherige Methoden zur Bestimmung von Dichten oder Verteilungen nutzen vor allem statistische Modellierungen durch Gaußsche Verteilungen [5] oder verwenden Bayesian Netzwerke [6]. Damit ist eine zwar eine exaktere Modellierung der Verteilungseigenschaften möglich (z.B. das Integral normiert sich automatisch zu Eins), der benötigte Aufwand übersteigt aber häufig den zur eigentlichen Funktionsapproximation.

Verwendet man zur Funktionsapproximation lokale Verfahren wie Radiale Basisfunktionsnetze oder Lokale Lineare Karten, kann man Vertrauenswürdigkeitsmaße auch direkt aus den Aktivierungen der jeweiligen lokalen Modelle ableiten, ohne dass man parallele Netze einführen muss [7].

Literatur

- [1] D.E. Rumelhart and J. E. McClelland (1986). *Parallel Distributed Processing*, MIT Press, Cambridge Mass.
- [2] M. Rosenblatt, (1956), *Remarks on some Nonparametric Estimation of a Density Function*, Ann Math. Statist., 27 823-825.
- [3] E. Parzen, (1962), *On Estimation of a Probability Density Function and Mode*. Ann of Math. Statist., 33 1065-1076.
- [4] L. Kindermann, A. Lewandowski, M. Tagscherer and P. Protzel (1999). *Computing Confidence Measures and Marking Unreliable Predictions by Estimating Input Data Densities with MLPs*. Proceedings of the Sixth International Conference on Neural Information Processing, Perth, Australia. Vol I: 91-94.
- [5] V. Tresp, S. Ahmad and R. Neumeier (1994), *Training neural networks with deficient data*, in J.D. Cowan, G. Tesauro and J. Alspector (eds), *Advances in Neural Information Processing Systems 6*, Morgan Kaufmann, San Mateo, California.
- [6] D.J.C. MacKay (1995) *Bayesian neural Networks and density networks*, Nuclear Instruments and Methods in Physics Research, Section A 354(1): 73-80.
- [7] M. Tagscherer, L. Kindermann, A. Lewandowski and P. Protzel: (1999) *Overcome neural limitations for real world applications by providing confidence values for network predictions*. Proceedings of the Sixth International Conference on Neural Information Processing, Perth, Australia. Vol II: 520-525.