Projekt *Learning and Softcomputing*

Sommersemester 2015: Lernversuche zur automatischen Zeichenerkennung mit einem Backpropagation-Netz

1 Aufgabenstellung

Ein neuronales Netz soll so realisiert und trainiert werden, dass es die 26 Großbuchstaben (Versalien, Majuskeln) A, B, ... Z erkennt. Gleiche Großbuchstaben können unterschiedliche Form haben. Sie treten in unterschiedlichen Fonts und auch handgeschrieben auf.

Ziel der Projektaufgabe ist, die Thematik der *Neuronalen Netze* mit *Backpropagation* Lern-Algorithmus zu durchdringen, sich in ein Software-Framework zur Realisierung neuronaler Netze einzuarbeiten und dessen Software-Komponenten zur Lösung der gestellten Aufgabe einzusetzen.

1. Verschaffen Sie sich bitte einen guten Überblick über Backpropagation-Netze. Das Literaturverzeichnis am Ende dieser Aufgabenstellung gibt Ihnen einige Ankerpunkte an denen Sie Ihre Recherchen beginnen können.

2. Wählen Sie bitte ein Software-Framework zur Realisierung Ihres neuronalen Netzes aus. Wir wollen mehrlagige neuronale Netze mit Backpropagation Lern-Algorithmus realisieren. Das Framework muss folgende Eigenschaften aufweisen:

   - Modifikation der Parameter: Lernrate, Momentum und Anfangsbelegung der Gewichte
   - Unterstützung für reine Feedforward-Netze (Vermaschungen, Schleifen oder weiterführenden Topologien wollen wir in diesem Projekt nicht betrachten.)
   - Unterstützung von Backpropagation
   - Konstruktionen mehrlagiger Netze mit mindestens 1 und 2 versteckten Schichten (*hidden Layers*) mit jeweils wählbarer Neuronenanzahlen
   - Unterstützung für musterweises und epochenweises Lernen
   - Identischer innerer Aufbau der Neuronen etwa mit sigmoider Aktivierungsfunktion (Netze mit verschiedenen Aktivierungsfunktionen für unterschiedliche Neuronen betrachten wir nicht.)

   Mögliche Frameworks für Python, Java oder C finden Sie weiter unten in Abschnitt 3. Bitte machen Sie sich mit dem Framework Ihrer Wahl vertraut.

3. Schreiben Sie unter Verwendung des von Ihnen gewählten Frameworks ein Programm, dass ein passendes neuronales Netz mit Backpropagation Lern-Algorihmus implementiert und parametrisieren Sie es bitte in verschiedener Weise, um eine geeignete Konfiguration zu ermitteln.

4. Organisieren Sie bitte das Anlernen ihres neuronalen Netzes anhand der Prototypen für die 26 Großbuchstaben mit vorgegebenen Zeichensätzen (vgl. Abschnitt 2):

- Training mit den Windows-Schriften und Test (Recall-Phase) mit den *ungesehenen* Windows-Schriften
- Training wiederum mit den Windows-Schriften und Test mit den ungesehenen Windows-Schriften incl. FH-Handschriften

5. Bitte diskutieren Sie die Ergebnisse hinsichtlich der Aspekte *Erkennungs-*, *Falschzuweisungs-* und *Rückweisungsraten*.
6. Stellen Sie bitte alle Versuchsergebnisse in Form aussagekräftiger Diagramme dar, die von Ihrem Programm erzeugt werden.
7. Insbesondere untersuchen Sie bitte die Parameter:
   - Netzwerktopologie: Anzahl und Anordnung der Neuronen bezüglich der verdeckten Schichten (mindestens 1-2 verdeckte Schichten)
   - Lernen von Einzelmustern bzw. Lernen in Epochen
   - Lernrate und Momentum
8. Bitte führen Sie zum Abschluss der Veranstaltung am Ende des Semester Ihr Programm vor und präsentieren Sie bitte in einem Vortrag Ihre Ergebnisse (ca. 15 min pro Gruppe).
9. Erstellen Sie bitte ein Abschlussprotokoll, dass die wesentlichen Ergebnisse Ihrer Untersuchungen zusammenfasst.

Beim Entwickeln Ihres Programms ist es sinnvoll, einfach zu beginnen und erst nach dem Sie erste Erfahrungen gesammelt haben, nach und nach kompliziertere Sachverhalte zu betrachten.

Beginnen Sie also nach der grundsätzlichen Einarbeitung und ersten Tests bitte mit folgenden Einstellungen:

- Netztopologie mit nur einer versteckten Schicht
- Musterweises Lernen
- einfache Initialisierung der Gewichte mit Zufallswerten zwischen -1 und 1.

Weitere Versuche und Messungen sollten Sie erst starten, wenn Sie mit diesen Randbedingungen plausible Ergebnisse erreichen können.

2 Datenmaterial

Zum Trainieren und zum Testen des Lernvorgangs steht Ihnen Datenmaterial zur Verfügung, das Eingabemuster und zugehörige erwartete Ausgabemuster enthält.

Datenformat

Das Datenmaterial für die neuronalen Netze liegt in einem *Patternfiles* genannten Format vor (Datei-Endung `.pat`). Die einzelnen Buchstabenformen (Eingabemuster) sind in einer 14x14-Binärbild-Matrix mit weiss/schwarz-Pixeln (196 Dimensionen) erfasst, die Klassifikation (Ausgabemuster) in einem 26-dimensionalen Vektor. Um das Lernverhalten zu verbessern, sind die Eingabemuster auf den Wertebereich $[-0.5, 0.5]$, die Ausgabemuster auf $[0.2, 0.8]$ normiert.

Patternfiles haben folgende Struktur :

```
14   ; 1. Zeile: Anzahl der Zeilen des Eingabemusters
14   ; 2. Zeile: Anzahl der Spalten des Eingabemusters
1    ; 3. Zeile: Anzahl der Zeilen des Ausgabemusters
26   ; 4. Zeile: Anzahl der Spalten des Ausgabemusters

; [ein] Es folgt das erste Eingabemuster (hier ein A), entsprechend 14x14 binär codierter und
; in das Intervall [-0.5, 0.5] transformierter Pixeldaten (Zeilen 5-18)
-0.50 -0.50 -0.50 -0.50 -0.50  0.50  0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50 -0.50
-0.50 -0.50 -0.50 -0.50 -0.50  0.50  0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50 -0.50
-0.50 -0.50 -0.50 -0.50 -0.50  0.50  0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50 -0.50
-0.50 -0.50 -0.50 -0.50  0.50  0.50  0.50  0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50
-0.50 -0.50 -0.50 -0.50  0.50  0.50 -0.50 -0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50
-0.50 -0.50 -0.50 -0.50  0.50  0.50 -0.50 -0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50
-0.50 -0.50 -0.50  0.50  0.50  0.50 -0.50 -0.50  0.50  0.50  0.50 -0.50 -0.50 -0.50
-0.50 -0.50 -0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50  0.50  0.50 -0.50 -0.50 -0.50
-0.50 -0.50 -0.50  0.50  0.50  0.50  0.50  0.50  0.50  0.50  0.50 -0.50 -0.50 -0.50
-0.50 -0.50  0.50  0.50  0.50  0.50  0.50  0.50  0.50  0.50  0.50  0.50 -0.50 -0.50
-0.50 -0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50 -0.50  0.50  0.50  0.50 -0.50 -0.50
-0.50 -0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50 -0.50 -0.50  0.50  0.50 -0.50 -0.50
-0.50  0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50 -0.50 -0.50  0.50  0.50  0.50 -0.50
-0.50  0.50  0.50 -0.50 -0.50 -0.50 -0.50 -0.50 -0.50 -0.50  0.50  0.50  0.50 -0.50

; [aus] Danach folgt in der Zeile 19 das dazu passende erste Ausgabemuster, entsprechend
; 26 binär codierter und in das Intervall [0.2, 0.8] transformierter gewünschter Ausgabewerte.
(Hier auf zwei Zeilen gedruckt.)

0.80  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20
0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20  0.20

; Danach folgt eine beliebige Anzahl von Ein- und Ausgabemustern ([ein] gefolgt von [aus]).
```

In [Kruse+91] findet sich zur Skalierung der Ein- und Ausgabemuster folgende Überlegung:

> (...) Das Bestimmen der Ein- und Ausgabewerte von zu lernenden Mustern hat einen großen Einfluss auf das Lernverhalten des Netzwerkes. Durch eine ungünstige Auswahl der Musterwerte kann es zu einer Konvergenz des Fehlerwertes gegen ein lokales Minimum der Fehlerfunktion (s.u.) oder sogar zu einem Abbruch des Backpropagation-Lernverfahrens kommen.

Werden an einem Backpropagation-Netzwerk binäre Eingabewerte 0 und 1 angelegt, so werden die Verbindungen zu Eingabeneuronen, an denen eine 0 anliegt, nicht trainiert. Dieses liegt darin begründet, dass die Veränderung der Verbindungsgewichte zu Eingabeneuronen proportional zum Ausgabewert der Eingabeneuronen erfolgt (...). Um für binäre Eingabewerte die gleiche proportionale Veränderung der Verbindungsgewichte zu erhalten, empfiehlt es sich, den Wert der Eingabevektoren von Betrag her gleich zu setzen. In den folgenden Beispielen werden binäre Eingabewerte durch die Werte -0.5 und 0.5 ersetzt. Kontinuierlich verlaufende Eingabewerte sollten ebenfalls in das Intervall [-0.5, 0.5] transformiert werden. Bei einer Anhäufung der transformierten Eingabewerte um den Wert 0 empfiehlt es sich aus oben genannten Gründen, das Intervall in einen anderen Wertebereich zu verschieben. Als alternative Lösung hat sich das Intervall [0.2, 0.8] erwiesen.

Durch die Wahl der Ausgabefunktion wird der Bereich der möglichen Ausgabewerte festgelegt. Bei einer sigmoiden Aktivierungsfunktion und der Identitätsfunktion als Ausgabefunktion wird der Ausgabewert von Backpropagation-Netzwerken auf das Intervall (0,1) beschränkt. Die Ausgabewerte 0 und 1 können durch oben beschriebene Backpropagation-Netzwerke nicht erreicht werden, da sich die sigmoide Aktivierungsfunktion nur asymptotisch diesen Werten nähert. Werden diese Werte als Zielwerte vorgegeben, versucht der Backpropagation-Lernalgorithmus, eine möglichst große Annäherung der tatsächlichen Ausgabewerte an die Zielwerte zu erreichen. Für Ausgabewerte, deren Wert nahe bei 0 bzw. 1 liegt, ist ein sehr großer negativer bzw. positiver Wert des Nettoinputs erforderlich. Da der Ausgabewert von inneren Neuronen durch die sigmoide Aktivierungsfunktion ebenfalls auf das Intervall (0,1) beschränkt ist, müssen die Verbindungsgewichte sehr hohe positive bzw. negative Werte annehmen, um den erforderlichen großen Nettoinput zu erzeugen. Abgesehen von anderen erheblichen Beeinflussungen des Lernverfahrens kann es bei zu großen positiven bzw. negativen Verbindungsgewichten zu einem Laufzeitfehler des Programms und damit zum Abbruch des Lernverfahrens kommen. Es empfiehlt sich daher, die Zielwerte für die zu lernenden Muster auf ein Intervall zu beschränken, dessen Grenzwerte ohne extrem hohe positive bzw. negative Verbindungsgewichte erreicht werden können. In den folgenden Beispielen werden die vorgegebenen Ausgabewerte auf das Intervall [0.2, 0.8] beschränkt. (...)

Trainings- und Test-Muster

Das Datenmaterial ist in *Trainings-Muster* und in *Test-Muster* (ungesehen) eingeteilt. Letztere gibt es als gedruckte und handgeschriebene Muster.

1. Die Datei `Trainingsdaten.pat` zur Verwendung während der Lernphase enthält die Großbuchstaben der folgenden Schriften im Patternfile-Format:

Schriftart	Tahoma	*Tahoma*	Times New Roman	*Times New Roman*	Arial	*Arial*	Courier New	*Courier New*	AR CENA	*AR CENA*
Schnitt	Standard	*Kursiv*	Standard	*Kursiv*	Standard	*Kursiv*	Fett	*Fett-Kursiv*	Standard	*Standard-Kursiv*
Grad	150pt	*150pt*	150pt	*150pt*	150pt	*150pt*	150pt	*150pt*	150pt	*150pt*
	A	*A*	A	*A*	A	*A*	**A**	***A***	A	*A*
	B	*B*	B	*B*	B	*B*	**B**	***B***	B	*B*
	C	*C*	C	*C*	C	*C*	**C**	***C***	C	*C*
	D	*D*	D	*D*	D	*D*	**D**	***D***	D	*D*
	E	*E*	E	*E*	E	*E*	**E**	***E***	E	*E*
	F	*F*	F	*F*	F	*F*	**F**	***F***	F	*F*
	G	*G*	G	*G*	G	*G*	**G**	***G***	G	*G*
	H	*H*	H	*H*	H	*H*	**H**	***H***	H	*H*
	I	*I*	I	*I*	I	*I*	**I**	***I***	I	*I*
	J	*J*	J	*J*	J	*J*	**J**	***J***	J	*J*
	K	*K*	K	*K*	K	*K*	**K**	***K***	K	*K*
	L	*L*	L	*L*	L	*L*	**L**	***L***	L	*L*
	M	*M*	M	*M*	M	*M*	**M**	***M***	M	*M*
	N	*N*	N	*N*	N	*N*	**N**	***N***	N	*N*
	O	*O*	O	*O*	O	*O*	**O**	***O***	O	*O*
	P	*P*	P	*P*	P	*P*	**P**	***P***	P	*P*
	Q	*Q*	Q	*Q*	Q	*Q*	**Q**	***Q***	Q	*Q*
	R	*R*	R	*R*	R	*R*	**R**	***R***	R	*R*
	S	*S*	S	*S*	S	*S*	**S**	***S***	S	*S*
	T	*T*	T	*T*	T	*T*	**T**	***T***	T	*T*
	U	*U*	U	*U*	U	*U*	**U**	***U***	U	*U*
	V	*V*	V	*V*	V	*V*	**V**	***V***	V	*V*
	W	*W*	W	*W*	W	*W*	**W**	***W***	W	*W*
	X	*X*	X	*X*	X	*X*	**X**	***X***	X	*X*
	Y	*Y*	Y	*Y*	Y	*Y*	**Y**	***Y***	Y	*Y*
	Z	*Z*	Z	*Z*	Z	*Z*	**Z**	***Z***	Z	*Z*

2. Die Datei `TestWindowsSchrift.pat` zur Verwendung während der Recall-Phase enthält die Großbuchstaben der folgenden Schriften im Patternfile-Format:

Schriftart	Verdana	*Verdana*	AR CHRIST	*AR CHRIST*
Schnitt	Standard	*Kursiv*	Standard	*Kursiv*
Grad	150pt	*150pt*	150pt	*150pt*
	A	*A*	A	*A*
	B	*B*	B	*B*
	C	*C*	C	*C*
	D	*D*	D	*D*
	E	*E*	E	*E*
	F	*F*	F	*F*
	G	*G*	G	*G*
	H	*H*	H	*H*
	I	*I*	I	*I*
	J	*J*	J	*J*
	K	*K*	K	*K*
	L	*L*	L	*L*
	M	*M*	M	*M*
	N	*N*	N	*N*
	O	*O*	O	*O*
	P	*P*	P	*P*
	Q	*Q*	Q	*Q*
	R	*R*	R	*R*
	S	*S*	S	*S*
	T	*T*	T	*T*
	U	*U*	U	*U*
	V	*V*	V	*V*
	W	*W*	W	*W*
	X	*X*	X	*X*
	Y	*Y*	Y	*Y*
	Z	*Z*	Z	*Z*

3. Die Datei `TestHandschriften.pat` soll ebenfalls während der Recall-Phase verwendet werden. Sie enthält handgeschriebene Großbuchstaben, die Teilnehmer der vergangenen Semester abgegeben haben. Auch wir wollen Handschriftenproben abgeben. Dazu werden passende Bögen verteilt und Ihre Schriftproben geeignet digitalisiert.

(Rohmaterialien mit den Ausgangs-Schriftmustern im Portable-Network-Grafics-(PNG)-Format finden sich in den Dateien `Trainingsdaten.zip`, `TestWindowsSchrift.zip` bzw. `TestHandschriften.zip`. Diese Dateien dienen Ihnen zur Illustration und sollen **nicht** für das Trainieren Ihres neuronalen Netzes verwendet werden.)

3 Frameworks

Hier eine Auswahl typischer Frameworks zur Realisierung neuronaler Netze.

- Java
  Java Neuro Network Framework *Neuroph*
  `http://neuroph.sourceforge.net`
- Python
  Python-Based Reinforcement Learning, Artificial Intelligence and Neural Network Library
  `http://pybrain.org/`
- C/C++
  Fast Artificial Neural Network Library
  `http://leenissen.dk/fann/wp/`

Gerne können Sie auch andere Frameworks für die Programmiersprache und die Plattform Ihrer Wahl untersuchen und ggf. einsetzen.

4 Literatur

Die folgenden Bücher finden sich auch in unserer Bibliothek:

[Braun+96] Braun, Heinrich ; Feulner, Johannes ; Malaka, Rainer: *Praktikum Neuronale Netze*, Springer, Berlin 1996

[Brause91] Brause, Rüdiger: *Neuronale Netze: Eine Einführung in die Neuroinformatik*, Teubner Verlag, Stuttgart 1991

[Brunack+93] Brunak, Sören ; Lautrup, Benny: *Neuronale Netze: Die nächste Computer-Revolution*, Hanser Verlag, München 1993

[Hamilton93] Hamilton, Patrick: *Künstliche neuronale Netze: Grundprinzipien, Hintergründe, Anwendungen*, VDE-Verlag, Berlin 1993

[Kratzer90] Kratzer, Klaus Peter: *Neuronale Netze: Grundlagen und Anwendungen*, Hanser Verlag, München 1990

[Kruse+91] Kruse, Hilger ; Mangold, Roland ; Mechler, Bernhard ; Penger, Oliver *Programmierung Neuronaler Netze: eine Turbo Pascal Toolbox*, Addison Wesley, Reading Mass. 1991

[Ritter+91] Ritter, Helge ; Martinez, Thomas ; Schulten, Klaus: *Neuronale Netze: Eine Einführung im die Neuroinformatik selbstorganisierender Netzwerke*, Addison Wesley, Reading Mass. 1991

[Stanley91] Stanley, Jeanette ; Bak, Evan: *Neuronale Netze: Computersimulation biologischer Intelligenz*, Systema Verlag, München 1991

Außerdem bieten die Einzelnachweis-, Literatur- und Weblinks-Teile der deutschen Wikipedia-Artikel zu den Stichworten *Künstliches neuronales Netz*, *Backpropagation*, *Perzeptron* bzw. die References- und Bibliography-Teile im englischen Wikipedia zu *Artificial neural network*, *Backpropagation*, *Multilayer perceptron* eine Fülle von Artikeln zum Thema.

Verwenden Sie bitte auch das CiteSeer-System (`citeseerx.ist.psu.edu`), um Artikel zu finden und Zitierzusammenhänge zu verfolgen.