Katedra Wytrzymałości Materiałów
i Metod Komputerowych Mechaniki
www.kwmimkm.polsl.pl

Wydział Mechaniczny Technologiczny
Politechnika Śląska

METODY HEURYSTYCZNE

LABORATORIUM 5: Sztuczne sieci neuronowe

opracował: dr inż. Witold Beluch
witold.beluch@polsl.pl

Gliwice 2012

Cel ćwiczenia

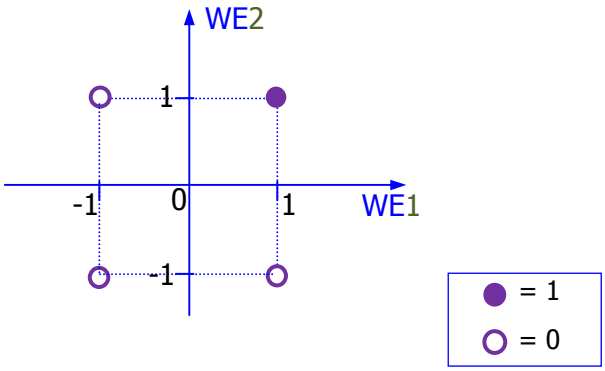
Wykonując ćwiczenie zastosujesz program *Evolutionary Algorithms* do zagadnień związanych ze sztucznymi sieciami neuronowymi. Przeprowadzisz eksperymenty numeryczne dotyczące zastosowania sztucznych sieci neuronowych do rozwiązywania problemów liniowo-separowalnych oraz do rozpoznawania znaków. Sprawdzisz też, jak sieć neuronowa reaguje na dane wejściowe podobne do tych, jakich się uczyła (zaburzone).

Program *Evolutionary Algorithms* – sztuczne sieci neuronowe

Program *Evolutionary Algorithms* oprócz realizacji obliczeń ewolucyjnych pozwala na korzystanie ze sztucznych sieci neuronowych (SSN). W ramach niniejszego ćwiczenia zastosujesz niniejszy program do zbudowania SSN, jej wytrenowania (nauczenia) i sprawdzenia działania. W pierwszej części (zadanie 1) wykorzystany będzie jeden neuron służący do rozwiązania zadania liniowo-separowalnego. W drugiej części (zadanie 2) zajmiesz się rozpoznawaniem przez SSN znaków i sprawdzisz, jak SSN reaguje na zaburzone dane wejściowe. W obydwu przypadkach wszystkie sztuczne neurony będą mieć sigmoidalną funkcję aktywacji[1].

Zadanie 1

Naucz pojedynczy neuron rozwiązywania problemu liniowo-separowalnego przedstawionego na Rys. 1. Sprawdź, jak przebiega uczenie z wyłączoną metodą momentum oraz z włączoną (dla różnych wartości współczynnika momentum). Zbadaj reakcję wytrenowanego neuronu na odpowiednie sygnały wejściowe.



Rys. 1 Problem liniowo-separowalny - opis

Założenia:

- funkcja aktywacji neuronu: sigmoidalna (*sigmoid activation function*);
- współczynnik uczenia (*learning rate*): 0.5.
- neuron jest uczony do spełnienia warunku, że błąd średniokwadratowy na wyjściu nie przekracza 0.06.

Z rysunku 1 jednoznacznie wynika, jaka jest pożądana odpowiedź neuronu na poszczególne wartości na wejściu. I tak np. zestaw sygnałów wejściowych (1,1) powinien spowodować, że na wyjściu pojawi się jedynka, z kolei podanie sygnałów wejściowych (-1,-1) ma skutkować zerem na wyjściu neuronu.

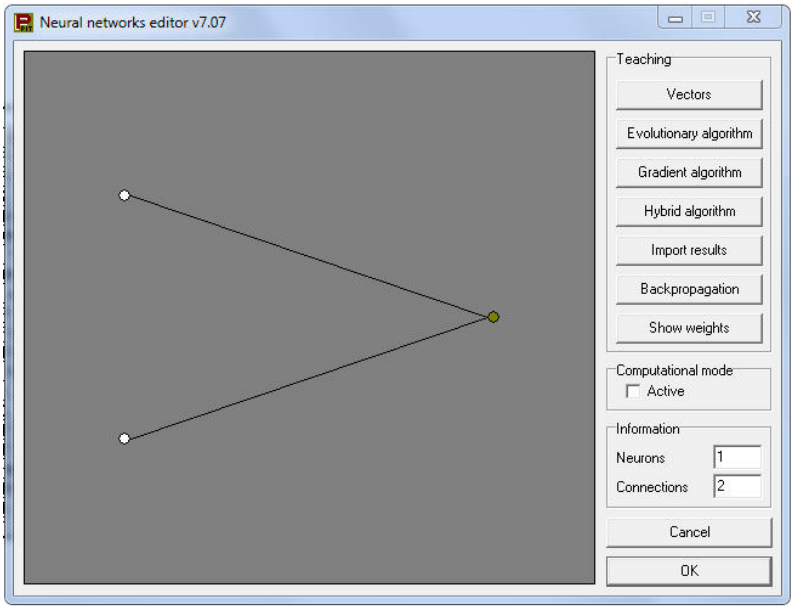
[1] Jak zapewne wiesz, sigmoidalna funkcja aktywacji neuronu jest bardzo często stosowana. Wynika to zarówno z faktu, iż jest ona funkcją o łatwej do policzenia pochodnej (co jest istotne przy stosowanych do uczenia sieci metodach gradientowych), jak również z powodu możliwości dostosowania kształtu (nachylenia) funkcji do potrzeb poprzez zmianę wartości pewnego współczynnika.

Do wykonania – zadanie 1

Uruchom program *Evolutionary Algorithms*. Jeśli pojawi się okienko z informacją o programie naciśnij *OK*.

1. Tworzenie SSN. Przejdź do menu *Tools->Neural Networks Editor*:

W zależności od tego, co w edytorze było uprzednio robione, okno edytora sieci może wyglądać różnie. Twoim zadaniem jest doprowadzenie go do poniższego stanu:



Rys. 2 Pojedynczy neuron z sigmoidalną funkcją aktywacji

Kliknięcie w oknie tworzenia sieci prawym klawiszem skutkuje pojawieniem się menu kontekstowego. Pozycje menu różnią się nieco w zależności od tego, czy kliknąłeś w któreś z wejść, w inny neuron czy w okolice neuronu. Sprawdź, jakie są możliwości związane z menu kontekstowym w poszczególnych przypadkach.

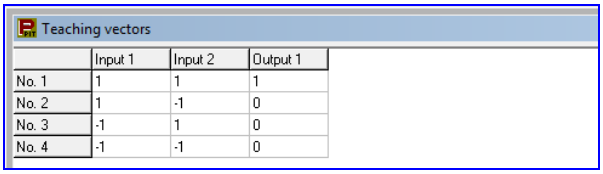
Gdy doprowadzisz do sytuacji, gdy masz 1 neuron i 2 wejścia, możesz je połączyć (jeśli nie są połączone) poprzez wybranie w menu kontekstowym *Create connection with* lub (bardziej praktyczne, gdy mamy sieć neuronową) *Create all connections*.

Zwróć uwagę na to, czy neuron ma sigmoidalną funkcję aktywacji. W tym celu kliknij na neuronie prawym klawiszem i wybierz *Activation function* a następnie *Sigmoid*[1]. Zaznaczenie opcji *Use for the whole network* skutkuje zastosowaniem danej funkcji aktywacji do wszystkich neuronów w sieci, co znów jest szczególnie użyteczne w przypadku, gdy rozpatrywany jest więcej niż jeden neuron.

2. Tworzenie wektorów uczących. Wciśnij przycisk *Vectors* w ramce *Teaching*. Pojawi się okno edytora wektorów uczących.

- Ustaw w ramce *Number of vectors* liczbę wektorów uczących na 4.
- Wypełnij tabelę zestawami danych wejściowych wraz z pożądanym sygnałem wyjściowym (kolejność wpisywania wektorów nie jest istotna). Powinieneś uzyskać tabelę podobną jak ta na rys. 3.

[1] Zauważ, że wraz z wybraniem funkcji aktywacji neuronu zmienia się jego kolor. Kolor „khaki” świadczy właśnie o sigmoidalnej funkcji aktywacji.


Teaching vectors

	Input 1	Input 2	Output 1
No. 1	1	1	1
No. 2	1	-1	0
No. 3	-1	1	0
No. 4	-1	-1	0

Rys. 3 Wektory uczące

- Wciśnij OK, by powrócić do okna edytora sieci neuronowych.

! Ważne – z powodu nie zawsze właściwego działania programu (odświeżanie plików z danymi) **zawsze** po wpisaniu wektorów uczących a przed rozpoczęciem uczenia należy w oknie „*Neural networks editor*" nacisnąć przycisk *OK* a następnie ponownie otworzyć okno „*Neural networks editor*".

3. Uczenie neuronu cz. 1. Wciśnij przycisk *Backpropagation* w ramce *Teaching*. Pojawi się okno uczenia sieci metodą propagacji wstecznej[1].

Zastosuj następujące ustawienia:

- Ramka *Weights*: zaznacz *Random*, *Radius*=1;
- Ramka *Learning rate*, *Rate*=0.5;
- Ramka *Momentum method* – początkowo *Active* ma być odznaczone.
- Ramka *Stop condition* – wybierz RMS[2], ustaw jego wartość zgodnie z treścią zadania.

Wciśnij przycisk *Compute*. Zaobserwuj w ramce *Results* między innymi:

- w polu *Iteration* – liczbę epok uczenia. Zapisz ją w tabeli 1 w protokole.
- w polu *RMS* – wartość błędu RMS;
- w polu *Corrects* – liczbę poprawnie rozpoznanych sygnałów. Jeśli widnieje tam wartość inna niż 4 oznacza to, że neuron nie nauczył się rozpoznawać któregoś z 4 prezentowanych znaków (jeśli tak jest – sprawdź poprawność wektorów trenujących).

Powtórz proces uczenia jeszcze dwukrotnie przy niezmienionych parametrach zapisując liczbę epok uczenia w tabeli 1. Zauważ, że wyniki niekoniecznie są takie same.

Włącz metodę momentum i ustaw stosowną wartość współczynnika momentum (*Rate*). Powtarzaj uczenie neuronu aż do wypełnienia tabeli 1.

4. Uczenie neuronu cz. 2.

Sprawdź skuteczność działania neuronu po 1000 epokach uczenia (ustaw odpowiednią opcję) z momentum. Przyjmij współczynnik momentum równy 0.8. Zanotuj wartość RMS a następnie:

- Wciśnij przycisk *Close*. Od tego momentu zakładamy, że neuron jest wytrenowany. W oknie edytora sieci neuronowych w polu *Computational mode* zaznacz *Active* – w ten sposób przejdziesz do trybu obliczeniowego sieci - pojawią się odpowiednie pola w pobliżu wejść i wyjścia.

[1] Jak wiesz metoda ta umożliwia uczenie sieci wielowarstwowych (z tzw. warstwami ukrytymi). Polega ona na dokładnym liczeniu błędów neuronów w warstwie wyjściowej, a następnie szacowaniu na tej podstawie błędów neuronów w warstwie/warstwach poprzednich (w odróżnieniu od warstwy wyjściowej, dla neuronów warstw ukrytych nie są znane pożądane wyjścia).

[2] RMS oznacza błąd średniokwadratowy liczony dla wszystkich wyjść w sieci (oczywiście w przypadku 1-go neuronu jest tylko 1 wyjście...). Liczy się go jako:

$$Q = \frac{1}{2}\sum_{j=1}^{N}\left(z^j - y^j\right)^2$$

gdzie: y – otrzymana odpowiedź neuronu, z – pożądana odpowiedź neuronu, j – liczba neuronów warstwy wyjściowej.

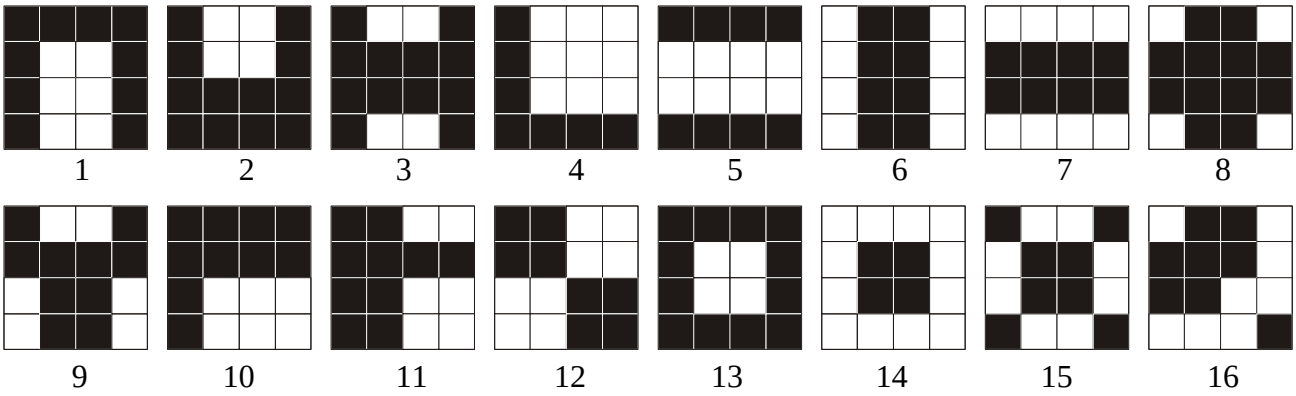
- Wpisuj jako wejścia odpowiednie zestawy wartości, tj. takie, jakich uczył się neuron. Zanotuj w tabeli 2 wartości wyjściowe. Zauważ, że uzyskujesz zwykle wartości tylko zbliżone do 0 bądź do 1. Zastanów się, dlaczego?

Ważne – jeśli na końcu wartości wyświetlanej wystąpi litera „E", to potraktuj całą wartość jako zero[1].

- Sprawdź, jak neuron reaguje na inne wartości niż te, których się uczył (np. lekko zmodyfikowane). Zanotuj wyniki (dane wejściowe i wyjściowe) w tabeli 2.

Zadanie 2

Naucz sieć neuronową 16-4-4 rozpoznawania 4 znaków z zestawu 16 znaków zaprezentowanego na rys. 4. Sprawdź reakcję wytrenowanej sieci neuronowej na odpowiednie sygnały wejściowe.



Rys. 4 Zestaw znaków do rozpoznania

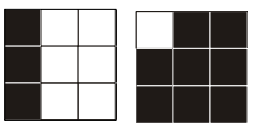
Założenia:

- funkcja aktywacji wszystkich neuronów: sigmoidalna (*sigmoid activation function*);
- współczynnik uczenia (*learning rate*): 0.6;
- metoda momentum: włączona, współczynnik (*rate*) =0.7.

Do wykonania – zadanie 2

Poproś prowadzącego ćwiczenia o wskazanie znaków, które ma rozpoznawać Twoja sieć neuronowa. Następnie:

- Wygeneruj sieć neuronową, która ma 16 wejść (reprezentujących 16 pikseli składających się na każdy znak), 4 neurony w warstwie ukrytej i 4 neurony (rozpoznające poszczególne znaki) na wyjściu.
- Wprowadź wektory uczące dla poszczególnych znaków. Jako wejścia wpisz poszczególne piksele, przyjmując, że czarny piksel to 1 a biały to 0. Wpisuj piksele wierszami. Jako wyjścia podaj 1 dla neuronu, który ma rozpoznawać dany znak a 0 na wyjściach pozostałych neuronów (neuron pierwszy ma rozpoznawać znak I, neuron drugi – znak II itp.). Przykład dla siatki 3x3 piksele (a zatem sieci 9-4-2) i rozpoznawania dwóch znaków jest przedstawiony na rys. 5.



Teaching vectors

Vectors

	Input 1	Input 2	Input 3	Input 4	Input 5	Input 6	Input 7	Input 8	Input 9	Output 1	Output 2
No. 1	1	0	0	1	0	0	1	0	0	1	0
No. 2	0	1	1	1	1	1	1	1	1	0	1

Rys. 5 Rozpoznawane znaki i odpowiadające im wektory uczące

[1] Powinien się również wyświetlać wykładnik potęgi, np. E-10 – niestety, odpowiednie pola są zbyt wąskie. Przyjmujemy zatem, że wykładnik potęgi jest na tyle duży, że cała wartość niewiele różni się od zera.

- Trenuj sieć przez 1500 epok. Zanotuj błąd RMS w tabeli 3.
- Odznacz opcję *Random* w polu *Weights.* Douczaj[1] sieć przez kolejne 2000 epok. Ponownie zanotuj RMS. Przyjmij, że sieć jest wystarczająco wytrenowana.
- Sprawdź odpowiedź sieci (w trybie obliczeniowym) po wpisaniu wejść odpowiadających Twoim znakom. Wartości wyjściowe zanotuj w tabeli 3.
- Sprawdź, jak nauczona na Twoich znakach sieć reaguje na nieznacznie zmodyfikowane znaki. W tym celu zmień dwa piksele w każdym znaku (w trybie obliczeniowym) starając się np. upodobnić znaki do siebie. Zauważ, że zmiana piksela polega na zgaszeniu zapalonego lub zapaleniu zgaszonego piksela, nie zaś na „przesunięciu" np. czarnego piksela (co jest równoważne zmianie 2 pikseli).
- Narysuj zmodyfikowane znaki obok tabeli 4 i sprawdź, jak nauczona wcześniej sieć reaguje w przypadku podania na wejścia zaburzonych wartości (czyli zmodyfikowanych znaków). Zanotuj odpowiedzi sieci w tabeli 4.

Sprawozdanie

- Sprawozdanie ma być dostarczone wyłącznie w formie elektronicznej.
- Nazwa pliku wg wzorca: MH_lab5_Jan_Kowalski.doc/pdf.
- Strona pierwsza to strona tytułowa.
- W sprawozdaniu należy zamieścić:
  1. Cel ćwiczenia.
  2. Opisy zadań.
  3. Strukturę sieci i ustawienia parametrów sieci (dla obydwu zadań)
  4. Skan/fotografię protokołu.
  5. Wnioski z ćwiczenia z podziałem na wnioski dotyczące zadania 1 i zadania 2.

Literatura i źródła

[1] R. Tadeusiewicz: Sieci neuronowe. Akademicka Oficyna Wydawnicza RM, Warszawa, 1993.

[2] S. Osowski: Sieci neuronowe w ujęciu algorytmicznym. WNT, Warszawa 1996.

[3] L. Rutkowski: Metody i techniki sztucznej inteligencji. PWN, Warszawa, 2006.

[1] Jeśli opcja *Random* jest aktywna, to po każdym naciśnięciu *Compute* wartości wag są ponownie losowane. Jeśli ją odznaczysz, to po naciśnięciu *Compute* uczenie będzie kontynuowane. Można w ten sposób douczać sieć, jeśli po założonej liczbie epok uczenia błąd generowany przez sieć będzie zbyt duży (często w praktycznych zagadnieniach przyjmuje się, że RMS na poziomie 0.02 jest już akceptowalny).

Protokół do ćwiczenia laboratoryjnego 5: Sztuczne sieci neuronowe

Imię i nazwisko	Rok ak.	Gr.	Sem.	Komp.	Data	Podpis prowadzącego
_____	20__/__	ME3	I	_____	_____	

Zadanie 1

Tabela 1. Pojedynczy neuron – problem liniowo-separowalny

Nr próby	Liczba epok uczenia			
	Bez momentum	Z momentum		
		współczynnik=0.3	współczynnik=0.6	współczynnik=0.9
1				
2				
3				

Tabela 2. Pojedynczy neuron – test skuteczności działania

RMS po 1000 epokach			
Dane WE	WE1	WE2	WY
dokładne	1	1	
	1	-1	
	-1	1	
	-1	-1	
zmodyfikowane			

Zadanie 2

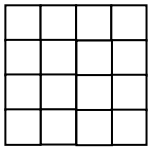
Twoje znaki to:

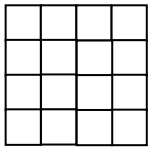
I	II	III	IV

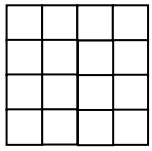
Tabela 3. Rozpoznawanie znaków – dane jak w ciągu uczącym

	RMS po 1500 epokach		RMS po 3500 epokach	
	Znak I	Znak II	Znak III	Znak IV
WY 1				
WY 2				
WY 3				
WY 4				

Zmodyfikowane znaki to:

I′ 

II′ 

III′ 

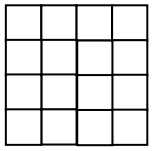
IV′ 

Tabela 4. Rozpoznawanie znaków – dane zmodyfikowane

	Znak I′	Znak II′	Znak III′	Znak IV′
WY 1				
WY 2				
WY 3				
WY 4				

Notatki: