I Krajowa Konferencja
Algorytmy Ewolucyjne
Murzasichle, 12-15.06.1996

ALGORYTM POSZUKIWAŃ EWOLUCYJNYCH Z MIĘKKĄ SELEKCJĄ W PROCESIE NAUCZANIA SZTUCZNEJ SIECI NEURONOWEJ

MARCIN MAKUCH, KRZYSZTOF PATAN, ANDRZEJ OBUCHOWICZ

Instytut Robotyki i Inżynierii Oprogramowania, Wyższa Szkoła Inżynierska w Zielonej Górze, ul. Podgórna 50, 65-246 Zielona Góra, e-mail: ao@irio.wsi.zgora.pl

1. Wprowadzenie

Sztuczne sieci neuronowe, których intensywny rozwój doświadczamy od drugiej połowy lat osiemdziesiątych, są interdyscyplinarną dziedziną badań mieszczącą się w zakresie zainteresowań między innymi bioników, neurofizjologów, psychologów, cybernetyków, informatyków, fizyków i automatyków. Z punktu widzenia informatyki interesującym jest porównanie własności systemów komputerowych z własnościami wzorowanej na mózgu sieci neuronowej. Z jednej strony komputery są dokładniejsze i szybsze w przetwarzaniu numerycznym niż mózg, z drugiej mózg przewyższa efektywnością przetwarzania dowolny superkomputer w zadaniach rozpoznawania mowy lub obrazu.

Sieci neuronowe związane są silnie z procesami optymalizującymi. Takimi procesami jest np. minimalizacja w procesie nauczania błędu średniokwadratowego w sieciach typu *Madaline* lub perceptronu wielowarstwowego, czy też minimalizacja w procesie odtwarzania funkcji energii w sieciach Hopfielda (*Korbicz i in.*, 1994). Klasyczne algorytmy poszukiwań punktu ekstremalnego danej funkcji były oparte na zasadzie akceptacji jedynie lepszych stanów od aktualnego. Tego rodzaju postępowanie nie daje żadnej gwarancji, że znaleziono globalnie najniższy punkt krajobrazu optymalizowanej funkcji. W rezultacie działania takich procedur zwykle osiąga się jedno z minimów lokalnych.

Najczęściej stosowanym typem sieci neuronowej jest perceptron wielowarstwowy (*Rosenblatt*, 1962), poddany procesowi nauczania nadzorowanego przy pomocy algorytmu propagacji wstecznej (*Werbos*, 1974; *Rumelhart i in.*, 1986). Topologia powierzchni funkcji błędu perceptronu jest multimodalna i nie najlepiej poznana. Proces uczenia często "utyka" w minimum lokalnym i, wobec braku zmian wartości błędu, jest przerywany. Algorytm propagacji wstecznej i inne stosowane algorytmy uczenia sieci neuronowych są oparte na klasycznych metodach optymalizacji (*Findeisen i in.* 1980). Metody te bazują na zasadzie *twardej selekcji*, zakładającej tworzenie nowych punktów bazowych do dalszych poszukiwań wyłącznie na bazie najlepszych z dotychczasowych. Nie posiadają zatem, bądź posiadają ze znikomym prawdopodobieństwem, umiejętności przekraczania siodeł pomiędzy dolinami funkcji multimodalnej, co znacznie ogranicza obszar przeszukiwań tych algorytmów. Odrzucenie zasady twardej selekcji, tzn., dopuszczenie możliwości akceptacji w kolejnych krokach algorytmu rozwiązań gorszych od dotychczasowych, prowadzi do wzrostu szans ucieczki z doliny (pułapki) minimum lokalnego. Na takiej nieprecyzyjnej regule wyboru opierają się metody tzw. *miękkiej selekcji*. Dwie z nich , *algorytmy genetyczne* (*Holland*, 1975) i *symulowane odprężanie* (*Metropolis i in.*, 1953), stosuje się z powodzeniem w sztucznych sieciach neuronowych (np. *Baker*, 1985; *Montana* i *Davis*, 1989).

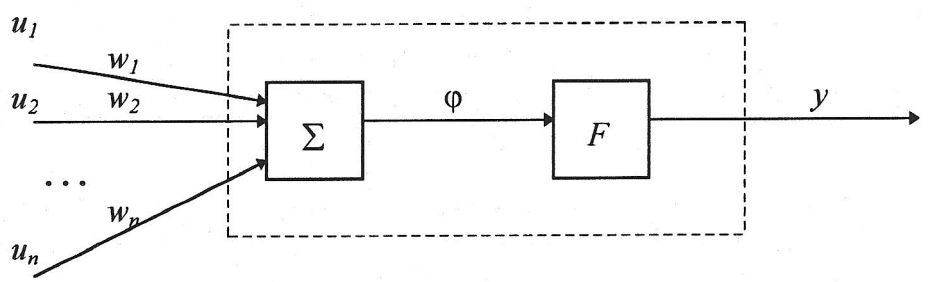
Naturalną drogą rozwoju mózgu jest jego ewolucja, oparta na swoistej metodzie prób i błędów. Kuszącą wydaje się idea wprowadzenia tych naturalnych praw do procesu uczenia sztucznych sieci neuronowych. Jednym z takich praw jest dopuszczenie w procesie ewolucji do rozwoju osobników nie najlepiej dostosowanych do środowiska naturalnego. Metodą opartą na powyższym prawie ewolucyjnym jest metoda poszukiwań ewolucyjnych z miękką selekcją Galara (*Galar*, 1990). "Miękkość" algorytmu ewolucyjnego Galara jest czynnikiem zwiększającym szansę na znalezienie ekstremum globalnego, lecz jednocześnie czyni go nie tyle metodą optymalizującą co adaptacyjną, nie gwarantującą znalezienie rozwiązania optymalnego, lecz jego lepsze bądź gorsze przybliżenie. Wynik jego pracy może być punktem wyjścia do poszukiwań końcowych przy pomocy klasycznych twardych algorytmów optymalizacji.

W niniejszej pracy rozważana jest sieć neuronowa rozwiązująca problem logiczny XOR. Jako algorytmu uczącego użyto algorytm ewolucyjny Galara. Problem XOR jest w literaturze klasycznym problemem testującym nowe metody nauczania sieci neuronowych, stąd uzyskane wyniki pozwolą na dokonanie porównań z dotychczas znanymi metodami.

Podstawowe pojęcia sieci neuronowych i struktura rozważanej sieci przedstawione są w części drugiej niniejszej pracy. W części trzeciej omówiono zasady działania algorytmu ewolucyjnego Galara. Część czwarta pracy zawiera opis eksperymentu i omówienie uzyskanych wyników. Podsumowanie pracy i plan dalszych badań zawiera część ostatnia.

2. Sieć neuronowa XOR

Podobnie jak w przypadku neuronowych sieci biologicznych, podstawowymi elementami, z których buduje się sztuczne sieci neuronowe, są sztuczne neurony. Ogólnie sztuczny neuron można rozpatrywać jako specyficzny przetwornik sygnałów. Składa się on (rys.1) z dwóch bloków: bloku sumowania i aktywacji. Na wejście bloku sumowania podawane są sygnały wejściowe mnożone przez odpowiednie współczynniki wag. Suma "ważonych" sygnałów jest przekazywana na blok aktywacji, który w zależności od jej wartości generuje sygnał wyjściowy.



Rys. 1. Model sztucznego neuronu.

W sposób formalny aktywność neuronu można wyznaczyć według wzoru:

$$y = F(\varphi) = F\left(\mathbf{w}^T \mathbf{u}\right) = F\left(\sum_{i=1}^{n} w_i u_i\right), \tag{1}$$

w którym **w** jest wektorem współczynników wag, **u** - wektorem sygnałów wejściowych, $(\bullet)^T$ - operatorem transponowania wektora lub macierzy, n - liczbą wejść neuronu, F - funkcją aktywacji neuronu.

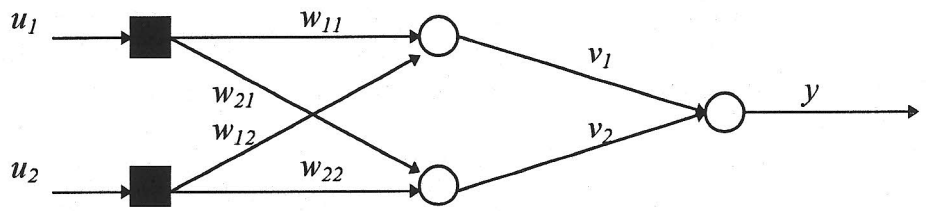
W literaturze istnieje wiele modeli funkcji aktywacji, liniowych i nieliniowych, najczęściej stosowaną jest funkcja sigmoidalna opisana wzorem :

$$y = F(\varphi) = \frac{1}{1+\exp(-\beta\varphi)}, \tag{2}$$

gdzie β jest zadanym parametrem.

Elementarne operacje wykonywane przez pojedyncze sztuczne neurony nie są szczególnie interesujące, gdyż faktyczna moc obliczeń neuronowych wynika dopiero z połączenia wielu neuronów w sieci tworzące przy tym różne struktury. Ogólnie można wyróżnić trzy główne typy architektur sztucznych sieci neuronowych, spośród wielu znanych w literaturze (*Osowski*, 1993) : sieci jednokierunkowe - tworzą je neurony ułożone w warstwach o jednym kierunku przepływu sygnałów; sieci rekurencyjne - sieci o połączeniach dwukierunkowych między neuronami lub ze sprzężeniami zwrotnymi; sieci komórkowe - neurony ułożone są w sposób wysoce uporządkowany (niczym w strukturze kryształu dwu lub więcej wymiarowego), sprzężenia wzajemne pomiędzy neuronami dotyczą jedynie najbliższego sąsiedztwa w strukturze.

Sieć XOR rozważana w niniejszej pracy należy do klasy sieci jednokierunkowych, ściślej, jest perceptronem wielowarstwowym (*Rosenblatt*, 1962). Jej architekturę przedstawia rysunek 2.



Rys. 2. Architektura sieci neuronowej XOR

Sygnał wyjściowy sieci XOR (rys.2) wyznacza wyrażenie :

$$y = F\left(\sum_{i=1}^{2} v_i F\left(\sum_{j=1}^{2} w_{ij} u_j\right)\right) \tag{3}$$

gdzie funkcja aktywności $F(\bullet)$ zdefiniowana jest wzorem (2); u_1 i u_2 - sygnały wejściowe przyjmujące wartości binarne 0 i 1; w_{ij} - waga połączenia pomiędzy j-tym elementem wejściowym a i-tym elementem przetwarzającym warstwy ukrytej; v_i - waga połączenia pomiędzy i-tym elementem przetwarzającym warstwy ukrytej a wyjściowym elementem przetwarzającym.

Na wejście sieci można podać cztery różne pary sygnałów binarnych. Proces uczenia sieci polega na takim doborze wag w_{ij} i v_i, aby zminimalizować błąd sieci, rozumiany jako suma po wszystkich możliwych parach (u_1, u_2) kwadratów błędów $(y - u_1 \text{ XOR } u_2)^2$.

3. Algorytm poszukiwań ewolucyjnych z miękką selekcją

Prosty model ewolucji fenotypowej zaproponował Galar w 1985 roku (*Galar*,1985). Model ten można sformalizować i wyrazić w języku algorytmów numerycznej optymalizacji (*Galar*,1990). Dana jest *n*-wymiarowa przestrzeń rzeczywista poszukiwań, na której określono nieujemną funkcję jakości *q*. Próba (osobnik) ma w przestrzeni ściśle określone położenie (typ) *x* i wskaźnik jakości (wskaźnik przystosowania) *q*. Poszukiwania to nic innego jak sekwencja generacji prób (generacje osobników), po *m* prób w generacji. Nowe próby powstają w wyniku wyselekcjonowania próby bazowej z generacji poprzedniej (osobnik rodzicielski) i zmodyfikowania jej współrzędnych.

Z tego wynika, że proces tworzenia nowej generacji jest tworzony dwuetapowo.

1. *Selekcja.* Z aktualnej generacji losuje się ze zwracaniem próby bazowe, prawdopodobieństwo wylosowania próby jest proporcjonalne do jej wskaźnika jakości. W odróżnieniu od twardej selekcji, która wybiera na bazę najlepszą z prób dotychczasowych - opisana metoda realizuje miękką selekcję, która nie gwarantuje, że najlepsza w generacji próba zostanie wylosowana jako bazowa.
2. *Modyfikacja.* Nową próbę otrzymuje się modyfikując losowo (rozkład normalny ze średnią 0 i odchyleniem standardowym v) współrzędne próby bazowej.

Poniżej został przedstawiony algorytm poszukiwań ewolucyjnych z miękką selekcją. Algorytm ten składa się z 5 kroków :

KROK 1: Określenie danych wejściowych procesu:

n - liczba parametrów,

m - liczebność generacji prób,

v - odchylenie standardowe modyfikacji,

q - funkcja celu,

$\left\{x_{k,j}^0\right\}$ - początkowe wartości parametrów : $k = 1,...,m;\quad j = 1,...,n$,

ξ_r - wartości zmiennej losowej rozłożonej równomiernie na przedziale [0,1),

$N(v)$ - funkcja generująca liczby losowe o rozkładzie normalnym o średniej 0 i wariancji v^2.

KROK 2 : Ocena :

$$\left\{x_1^i, x_2^i, \ldots, x_m^i\right\} \rightarrow \left\{q_1^i, q_2^i, \ldots, q_m^i\right\},$$

$$q_k^i = q\left[x_{k,1}^i, x_{k,2}^i, \ldots, x_{k,n}^i\right];\quad k = 1,\ldots m.$$

KROK 3 : Selekcja :

$$\left\{q_1^i, q_2^i, \ldots, q_m^i\right\} \rightarrow \{h(i,1), h(i,2), \ldots, h(i,m)\},$$

$$h(i,k) = \min\left\{h:\ \frac{\sum_{l=1}^{h} q_l^i}{\sum_{l=1}^{m} q_l^i} > \xi_r\right\};\quad r := r + 1.$$

KROK 4 : Modyfikacja :

$$\left\{x_{h(i,1)}^i, x_{h(i,2)}^i, \ldots, x_{h(i,m)}^i\right\} \rightarrow \left\{x_1^{i+1}, x_2^{i+1}, \ldots, x_m^{i+1}\right\},$$

$$x_k^{i+1} = x_{h(i,k)}^i + N(v)\ ;\ .$$

KROK 5 : Sprawdzenie warunku stopu. Jeżeli jest nie spełniony to $i = i + 1$ i powrót do kroku drugiego.

Powyższy algorytm przeprowadza poszukiwanie maksimum globalnego dla nieujemnie określonej funkcji.

W porównaniu z bardziej znanymi, zbliżonymi biorąc pod uwagę zasady działania, algorytmami genetycznymi, konstrukcja wyżej przedstawionego algorytmu jest znacznie prostsza, zależna od znacznie mniejszej ilości parametrów. Dla ściśle określonej funkcji celu algorytm sterowany jest zaledwie dwoma parametrami : liczebnością generacji i odchyleniem standardowym modyfikacji wyselekcjonowanych próbek. Drugim atutem algorytmu miękkiej selekcji jest dużo niższa złożoność obliczeniowa na poziomie jednej generacji w stosunku do algorytmów genetycznych.

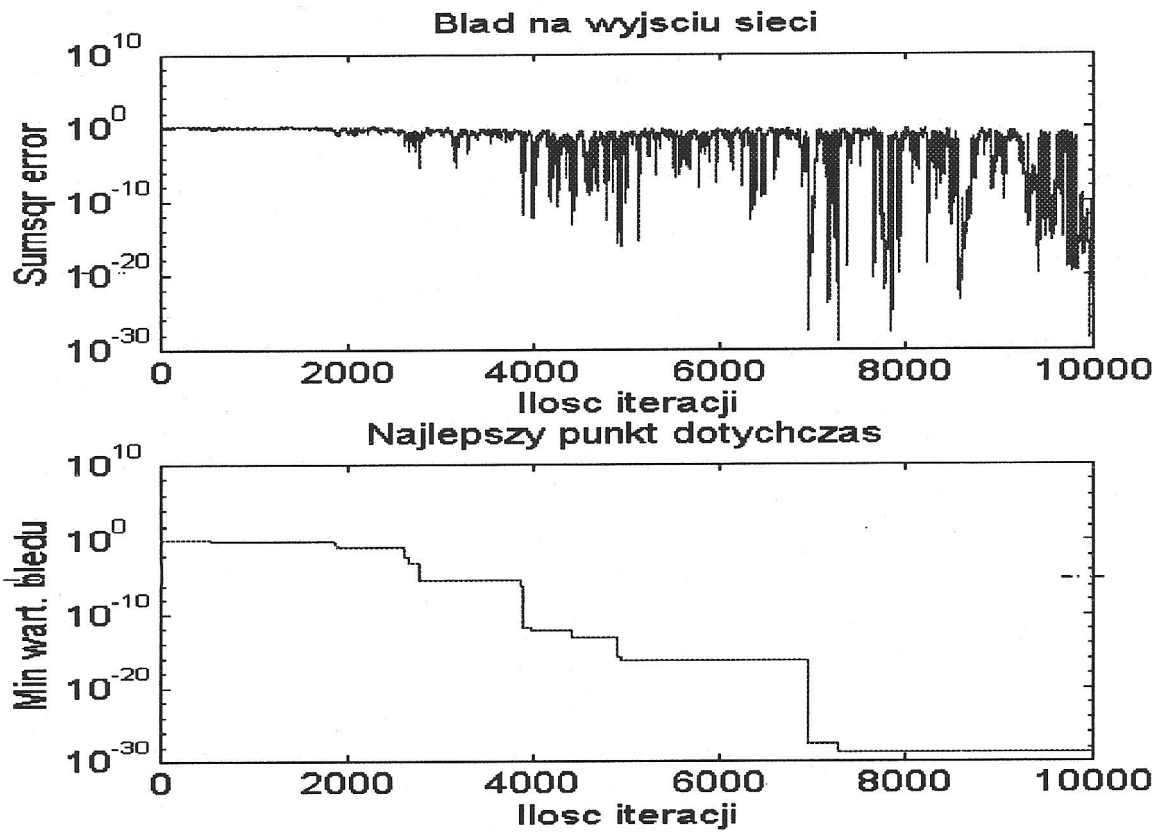
4. Nauczanie sieci neuronowej XOR

W procesie nauczania sieci neuronowej XOR minimalizowana jest funkcja celu postaci: $q = \sum_{i=1}^{4}\left(y_i - u_{1i}XORu_{2i}\right)^2$, która jest nieujemna. Ponieważ minimalna wartość błędu sieci jaką może przyjąć funkcja celu jest 0, a maksymalną 4, to funkcję celu można zdefiniować w postaci:

$$q = 4 - \sum_{i=1}^{4}\left(y_i - u_{1i}XORu_{2i}\right)^2. \tag{4}$$

gdzie y_i określa wyrażenie (3).

Funkcja celu q (4) jest nieujemna i poddana maksymalizacji przy pomocy algorytmu miękkiej selekcji. Wymiar próby określony jest liczbą wag połączeń sieci XOR (rys. 2).



Rys. 3. Wyniki nauczania sieci neuronowej XOR algorytmem poszukiwań ewolucyjnych z miękką selekcją. Liczebność populacji m=40, odchylenie standardowe modyfikacji v=1.

Proces badań nad nauczaniem sieci neuronowej XOR przy pomocy algorytmu poszukiwań ewolucyjnych z miękką selekcją, przebiegał dwutorowo. Badano skuteczność nauczania sieci w zależności od parametrów wejściowych - liczebności generacji m i odchyleniem standardowym modyfikacji wyselekcjonowanych próbek v. Dokonywano porównań z algorytmem propagacji wstecznej. Na rysunku 3 przedstawione są typowe wyniki otrzymywane w trakcie nauczania sieci. Pierwszy z wykresów przedstawia przebieg zmian błędu nauczania sieci dla próby średniej generacji w danej iteracji. Każdy z pików wykresu informuje o skupieniu się generacji wokół pewnego minimum lokalnego błędu nauczania. Wykres drugi przedstawia przebieg wartości błędu dotychczas najlepiej dopasowanej próbki.

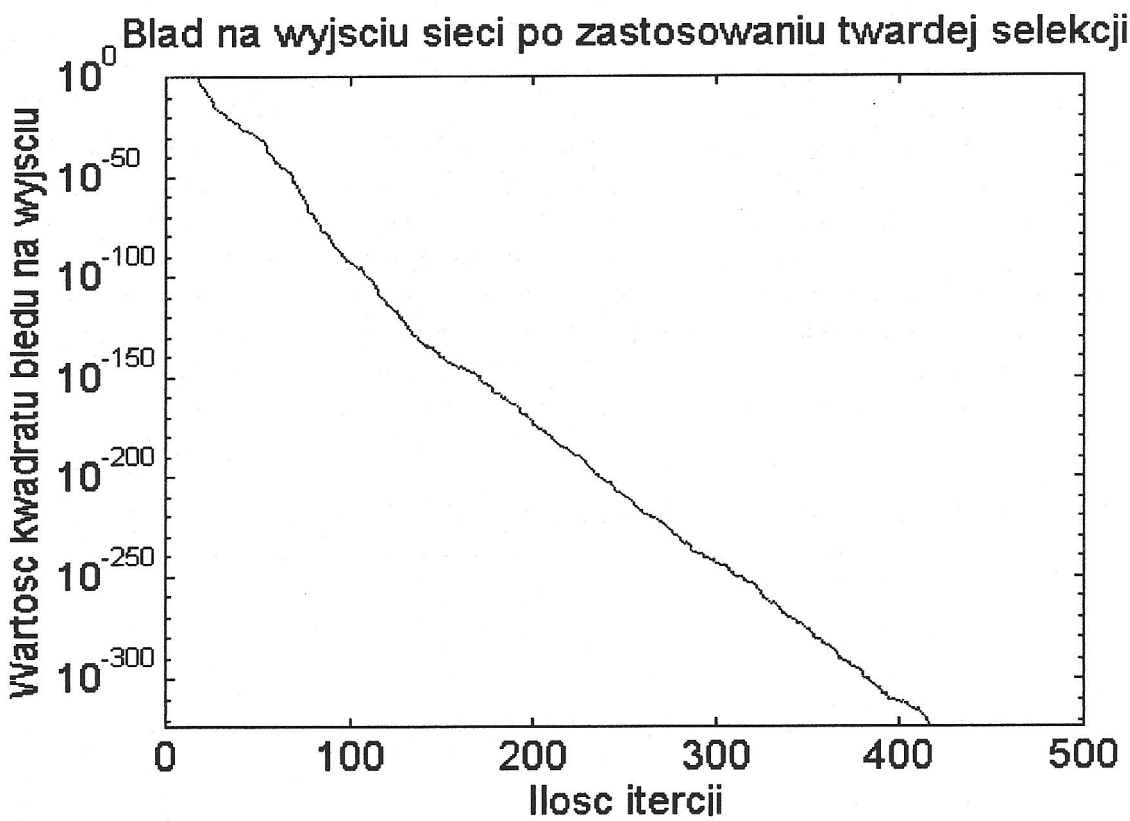
Łatwo zauważyć, że pierwsze zadawalające wartości błędu można uzyskać poniżej 3000 iteracji. Dla mniejszych generacji ilość iteracji potrzebna dla uzyskania pierwszych zadawalających wyników wzrasta, dla m=20 osiąga średnio 5000 iteracji. Jednakże należy mieć

na względzie, że złożoność obliczeniowa każdej iteracji spada, wymaga mniejszej liczby odwołań do wyznaczania wartości funkcji celu.

Dobór drugiego z parametrów - odchylenia standardowego modyfikacji ν - jest bardziej subtelny. Dla zbyt małych wartości (z doświadczeń wynika poniżej 0.3) sieć nie uczy się. Dla za dużych (powyżej 10) zmiana generacji jest zbyt chaotyczna, nieczuła na topologię optymalizowanej funkcji. Najkorzystniejsze wyniki otrzymywano dla ν=1.

Dokładność nauczenia sieci poniżej 10^{-30} osiągana po kilku tysiącach iteracji jest zupełnie nieosiągalna dla algorytmu propagacji wstecznej w tak małej ilości iteracji (w najlepszym przypadku dla propagacji wstecznej po 10000 iteracji uzyskiwano wartość błędu 10^{-3}). Ponadto dla poprawnie dobranych parametrów algorytmu Galara nie było przypadków nienauczenia sieci. W przypadku algorytmu propagacji wstecznej proces nauczania wielokrotnie utykał w minimach lokalnych i był przerywany.

Z pewnością najistotniejszym zarzutem przeciwko algorytmowi miękkiej selekcji jest jego złożoność obliczeniowa, niewspółmiernie duża w stosunku do szybkiego algorytmu propagacji wstecznej, lecz mniejsza od zbliżonych do niej algorytmów genetycznych. W celu przyspieszenia poszukiwań punktu optymalnego zastosować można dwuetapową pracę algorytmu. Pierwszy etap przeprowadzony jest według zasad wskazanych powyżej dopóty, dopóki nie uzyskamy pierwszych wyników z błędem na poziomie 10^{-1}-10^{-2}. W drugim przechodzimy do selekcji twardej - do modyfikacji wybierana jest grupa najlepiej dopasowanych punktów - bądź stosujemy inny algorytm np. propagacji wstecznej. Przypadek zastosowania selekcji twardej dla najlepszego punktu z 2000 iteracji dla generacji o liczebności 40 (rys.3) przedstawia rysunek 4.



Rys. 4. Przebieg nauczania sieci neuronowej XOR dla selekcji twardej dla próby określonej wstępnie przez selekcję miękką.

5. Podsumowanie

Celem niniejszej pracy była próba oceny przydatności algorytmu poszukiwań ewolucyjnych z miękką selekcję do nauczania sztucznych sieci neuronowych. Testy przeprowadzono dla struktury sieci neuronowej rozwiązującej problem logiczny XOR. Z badań wynika, że algorytm miękkiej selekcji jest bardzo przydatny dla wstępnego oszacowania rozwiązania optymalnego, bardzo skutecznie przechodząc przez napotkane doliny minimów lokalnych funkcji błędu nauczania sieci. To oszacowane rozwiązanie może być punktem początkowym dla mniej złożonych obliczeniowo metod nauczania sieci.

W trakcie badań zwrócono uwagę na możliwość zastosowania algorytmu poszukiwań ewolucyjnych z miękką selekcją do badań nad topologią multimodalnych funkcji błędu sieci jednokierunkowych. Powyższy problem określa kierunek dalszych badań.

Literatura

Baker J.E.,(1985) : *Adaptive selection methods for genetics algorithms.* - Proc. Int. Conf. Genetics Algorithms and Their Appplication, Hillsdale, New Jersey, Lawrence Erlbaum Associates, pp.101-111.

Findeisen W., Szymanowski J., Wierzbicki A.,(1980) : *Teoria i metody obliczeniowe optymalizacji.* - Warszawa : PWN.

Galar R., (1985) : *Hendicapped individua in evolutionary processes.*- Biol. Cybern.,**51**,pp.1-9.

Galar R.,(1990) : *Miękka selekcja w losowej adaptacji globalnej w R^n. Próba biocybernetycznego ujęcia rozwoju.* - Wrocław : Wydawnictwo Politechniki Wrocławskiej, seria: Monografie 17.

Holland J.H., (1975) : *Adaptation in Natural and Artificial Systems.* - Ann Arbor: University of Michigan Press.

Korbicz J., Obuchowicz A., Uciński D. (1994) : *Sztuczne sieci neuronowe, podstawy i zastosowania.* - Warszawa : Akademicja Oficyna Wydawnicza PLJ.

Metropolis N., Rosenbluth A.W., Rosenbluth M.N., Teller A.H., Teller E.,(1953) : *Equation of state calculation by fast computing machines.* - J.Chem.Phys., **21** , pp.1087-1092.

Montana D.J., Davis L., (1989) : *Training Feedforward Neural Networks Using Genetic Algorithms.* - BNN Systems and Technologies, Cambridge, Mass, USA.

Osowski S., (1993) : *Sztuczne sieci neuronowe - przegląd podstawowych struktur i metod uczenia.* - Prace 16-go seminarium SPETO, Ustroń, maj, s.31-52.

Rosenblatt F.,(1962) : *Principles of Neurodynamics.* - New York: Spartan.

Rumelhart D.E., Hinton G.E., Williams R.J., (1986) : *Learning Representations by Back-Propagating Errors.* Nature **323**, pp.533-536.

Werbos P., (1974) : *Beyong Regression: New Tools for Prediction and Analysis in Behavioral Sciences.* - Ph.D. Thesis, Harvard University.