Krzysztof Kaflowski

IS

Podstawy Sztucznej Inteligenji

Sprawozdanie ze Scenariusza nr 4

„Nauczanie Regułą Hebba bez nauczyciela”

1. Celem ćwiczenia było stowrzenie sieci neuronowej, która pogrupuje duże litery według ich podobieństwa nie korzystając z nauczyciela

Sieć neuronowa ma za zadanie pogrupować sieć neuronową według podobieństwa liter, np. M jest podobne do N więc wartość odpowiedzi sieci będzie w podobnym zakresie. Sieć neuronowa zwraca wartość funkcji sigmoidalnej, której wartość oscyluje w przedziale <-1;1>. W związku z tym grupowanie polega na tym, że litery będą miały podobną odpowiedź sieci. Zakresie taki może to być przykłado <0,2;0,4> i jeżeli litery mają odpowiedź sieci która zawiera się w tym przedziale to oznacza, że najprawdopodobniej są podobne do siebie.

Reguła Hebby bez nauczyciela polega na wykorzystaniu współczynnika uczenia, współczynnika zapominania, średniej ważonej oraz funkcji aktywacji bipolarnej, a także funkcji aktywacji sigmoidalnej. Wagi są obliczane z następującego wzoru:

Gdzie:

Weight - waga

Alfa – współczynnik uczenia

Beta – współczynnik zapominania

WeightSum – średnia ważona danych wejściowych

Input – dane wejściowe

Activ – funkcja aktywacji

Wykorzystane funkcje aktywacji:

Bipolarna

Sigmoidalna

Gdzie:

X – Średnia ważona danych wejściowych

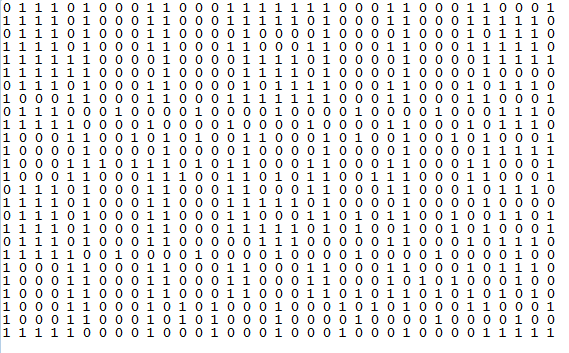
W regule Hebba ogranicznikami długości wykonywania programu jest ilość iteracji podana wcześniej, bądź wartość minimalnej zmiany wag, gdzie poniżej program przestaje działać.

Stworzona sieć neuronowa posiada jedną warstwę, która posiada jeden neuron. Uznano, że będzie to liczba wystarczająca.

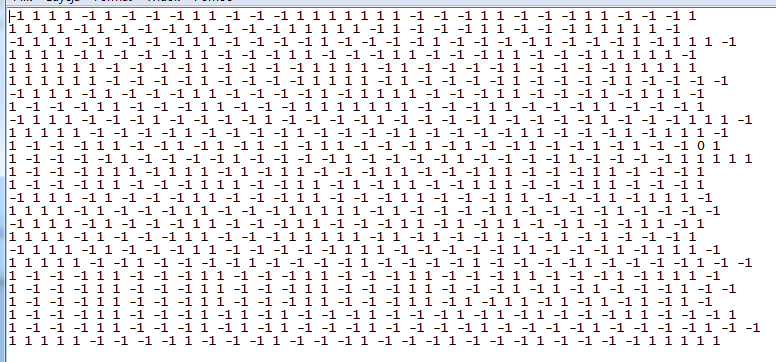
**2a)** Dane uczące oraz testujące:

Dane uczące oraz testujące są takie same i zawierają 26 liter alfabetu angielskiego. Stworzono dwa rodzaje notacji liter. Pierwsza to standardowa wersja reprezentacji zero-jedynkowej. Druga wersja to modyfikacja pierwszej, zamian zer na minus jeden.

Zestaw 1:



Zestaw 2:

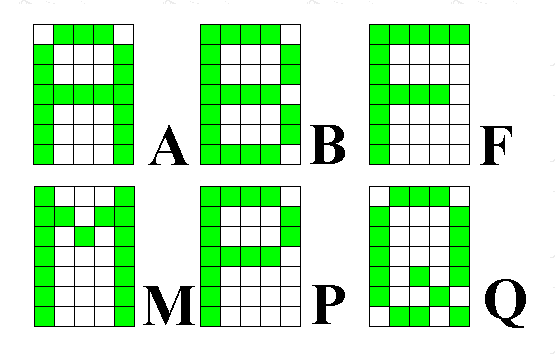


Każda linijka reprezentuje kolejne litery alfabetu. Zostały zapisane jako jedna linia dla ułatwienia implementacji.

Wszystkie schematy liter zostały pobrane z poniższego źródła:

<http://www.ai.c-labtech.net/sn/litery.html>

Litery są w postaci siatki 5x7, przykład wyglądu:



**2b)** Wyniki testów uczenia:

Jedna epoka uczenia zawiera cały alfabet, czyli 26 liter. Testowanie nauczonego neuronu badano na trzy sposoby:

* Stały współczynnik uczenia oraz liczba iteracji, zmienny współczynnik zapominania
* Stały współczynnik zapominania oraz liczba iteracji, zmienny współczynnik uczenia
* Stały współczynnik uczenia oraz zapominania, zmienna liczba iteracji

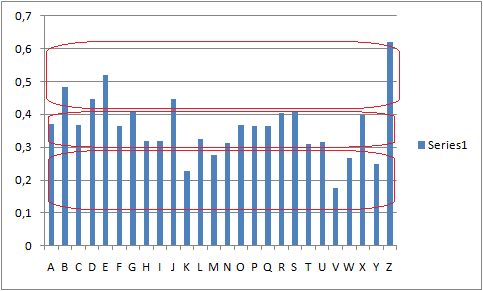
**Testowanie dla Zestawu 1 (zero-jedynkowo):**

**Stały współczynnik uczenia oraz liczba iteracji, zmienny współczynnik zapominania**

Współczynnik uczenia = 0,1

Ilość iteracji = 1000000

Współłczynnik zapominania = 0



Na wykresie zaznaczono jakie można wyróżnić grupy liter. Nie jest to dokłądny podział ale widać że sieć rozpoznała kilka podobnych liter:

Grupy:

P R S X O

M N

Współłczynnik zapominania = 0,1

Współłczynnik zapominania = 0,2

Współłczynnik zapominania = 0,4

Współłczynnik zapominania = 0,6

Współłczynnik zapominania = 0,9

**Stały współczynnik zapominania oraz liczba iteracji, zmienny współczynnik uczenia:**

Współczynnik zapominania = 0,1

Ilość iteracji = 1000000

Współczynnik uczenia = 0,01

Rozpoznane grupy:

H I LT U Y M N

B D J R S X

Z E

Współczynnik uczenia = 0,05

Współczynnik uczenia = 0,1

Współczynnik uczenia = 0,25

Współczynnik uczenia = 0,5

Współczynnik uczenia = 0,75

**Stały współczynnik uczenia oraz zapominania, zmienna liczba iteracji**

Współczynnik uczenia = 0,1

Współłczynnik zapominania = 0,1

Ilość iteracji = 100

Ilość iteracji = 1000

Ilość iteracji = 10000

Ilość iteracji = 100000

Ilość iteracji = 1000000

Ilość iteracji = 10000000

Rozponane grupy:

P R S T U H I

G D X E

K W Y

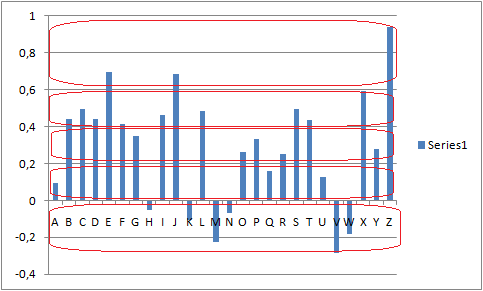
**Testowanie dla Zestawu 2 (-1 i 1):**

**Stały współczynnik uczenia oraz liczba iteracji, zmienny współczynnik zapominania:**

Współczynnik uczenia = 0,1

Ilość iteracji = 1000000

Współłczynnik zapominania = 0



Rozponane grupy:

W V M K

A U

G P O Y

B C D I L S T

J E Z

Współłczynnik zapominania = 0,1

Współłczynnik zapominania = 0,2

Współłczynnik zapominania = 0,4

Współłczynnik zapominania = 0,6

Współłczynnik zapominania = 0,9

**Stały współczynnik zapominania oraz liczba iteracji, zmienny współczynnik uczenia:**

Współczynnik zapominania = 0,1

Ilość iteracji = 1000000

Współczynnik uczenia = 0,01

Rozponane grupy:

P R S U

W V M

X Y J E

Współczynnik uczenia = 0,05

Współczynnik uczenia = 0,1

Współczynnik uczenia = 0,25

Współczynnik uczenia = 0,5

Współczynnik uczenia = 0,75

**Stały współczynnik uczenia oraz zapominania, zmienna liczba iteracji**

Współczynnik uczenia = 0,1

Współłczynnik zapominania = 0,1

Ilość iteracji = 100

Rozponane grupy:

X Y Z

W V M K H A

I J

P R

B D G

O U

=

Ilość iteracji = 1000

Ilość iteracji = 10000

Ilość iteracji = 100000

Ilość iteracji = 1000000

Ilość iteracji = 10000000

Rozponane grupy:

W V M K A

X Y Z

L I J E T S

P R G

O U

**2c)** Analiza wyników:

Zarówno dla Zestawu 1 jak i 2 współczynniki uczenia oraz zapominania pełnią bardzo ważną rolę. Współczynnik uczenia o małej wartości sprawia, że poprawa wag jest mniej szarpana i wynik końcowy jest dokładniejszy. Niesie to za sobą większą ilość iteracji potrzebnej do nauki przez co obciążenie komputera jest większe.

Współczynnik zapominania również jest bardzo ważny w nauce sieci. Ustawiony na małą wartość powoduje wzmocnienie nowej wartości przy poprawianiu wag. Ustawiony zbyt wysoko sprawia, że za każdym obrotem wagi zapominają cały progres nauczania i wynik jest spłycony.

Ustawienie na bardzo wysoką wartość współczynnika uczenia jak i współczynnika zapominania sprawia, że odpowiedź neuronu jest zbliżona dla wszystkich liter, przez co nie można wyróżnić żadnej grupy. Zwiększanie tych wartości sprawia spłycanie wyniku, a zmniejszanie sprawia, że wyniki mają większe skoki przez co można wyróżnić kilka podobnych do siebie liter.

Liczba iteracji ma najmniejszy wpływ i jest ściśle powiązana ze współczynnikami. Wraz z zwiększaniem się liczby iteracji, wyniki się wyostrzają, ale nie jest to taki przeskok jak w przypadku współczynników uczenia i zapominania.

Neuronowi udało się rozpoznać kilka prawidłowych według mnie grup. Litery M N K W przykładowo są z budowy podobne i można zakwalifikować jako jedno. Większość liter można zakwalifikować do grup uwzględniając wzrokowo ich podobny wygląd korelujący z podobną wartością odpowiedzi neuronu, lecz nie dla wszystkich.

Wykresy wyników z Zestawu 2 okazały się ku zaskoczeniu nieco bardziej czytelniejsze w odbiorze i analizie. Łatwiej na nich znaleźć odpowiednie rozpoznane grupy. Wykresy z zestawu 1 nie odbiegają wiele i też nadają się do analizy działania programu.

**2d)** Wnioski:

Neuronowi udaje się rozpoznać grupy, które można potwierdzić wzrokową analizą wyglądu. Wynik działania jest to jednak aproksymacja wyniku, nie jest ona defnitywną odpowiedzią z jaką literą mamy do czynienia. Rozwiązanie takie bardziej nadawałoby się w problemie, gdzie znajduje się większa granica błędu.

Aby zoptymalizować działanie programu należy ustawić parametry według swoich potrzeb. Dla testowanych ustawień najlepiej sprawdziło się:

Współczynnik uczenia = 0,1

Współczynnik zapominania = 0,1

Liczba iteracji = 1000000

Program wtedy wykonywał się bardzo szybka, a także dokładność rozpoznawania grup była już wysoka. Zwiększanie liczby iteracji powodowało kilkunastokrotnie zwiększenie czasu bez dużej zmiany dokładności działania programu.

**2e)** Listing kodu:

Cały kod programu razem z danymi uczącymi oraz testującymi znajdują się pod linkiem do GitHuba:

<https://github.com/kkaflows/PSI---Scenariusz-4>