Tomasz Tomala  
Podstawy Sztucznej Inteligencji  
Sprawozdanie z projektu nr 5

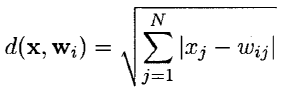
Celem ćwiczenia było poznanie budowy i działania sieci Kohonena przy wykorzystaniu reguły WTA do odwzorowania istotnych cech kwiatów.

**1) Syntetyczny opis budowy użytej sieci i algorytmów uczenia**

Reguła Kohonena opiera się na mechanizmie współzawodnictwa między neuronami.

Wagi każdego neuronu tworzą wektor wi = [wi1, wi2, …, wiN ]T. Przy założeniu normalizacji wektorów wejściowych, we współzawodnictwie wygrywa neuron, którego wagi najmniej różnią się od odpowiednich składowych tego wektora. Zwycięski neuron spełnia relację:

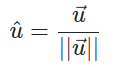
Gdzie d(x, w) oznacza odległość w sensie wybranej metryki między wektorem ***x*** i wektorem ***w***. Podczas ćwiczenia do obliczania odległości między wektorami użyłem miary według normy L1 (Mahnattan):



W strategii WTA zmiana wag dotyczy tylko neuronu zwycięzcy wg zależności:

Neurony przegrywające konkurencję nie zmieniają swoich wag.

Ważną rolę odgrywa nadmiarowość danych uczących. Wielokrotne powtórzenia podobnych wzorców stanowią „bazę wiedzy”, z której odpowiednią drogą wyciągane są wnioski decyzyjne. Do uczenia sieci użyłem zestawu danych istotnych cech kwiatów zaczerpniętych z Wikipedii. Były to wektory złożone z czterech składowych, które poddałem procesowi normalizacji. Normalizacja polega na podzieleniu każdej ze składowej wektora przez długość tego wektora:

  
Dane uczące zostały poddane normalizacji, ponieważ istnieje taka potrzeba przy małych wymiarach przestrzeni, np. n = 2, n = 3. Jeśli jednak chodzi o wektory wag neuronów, to nie musiały one być już później normalizowane w procesie uczenia, ponieważ przy znormalizowanych wektorach uczących ***x***, wektory wag – nadążając za nimi – stają się automatycznie znormalizowane.

Dodatkowo podczas ćwiczenia użyłem sporej nadmiarowości jeśli chodzi o ilość neuronów. Było to konieczne, ponieważ inicjalizacja wag sieci jest losowa, tak więc część neuronów możne znaleźć się w strefie, w której nie ma danych lub ich liczba jest znikoma. Neurony takie mają niewielkie szanse na zwycięstwo i zwane są neuronami martwymi.

Zaimplementowany przeze mnie klasa KohonenWTA składa się z następujących metod:

**2) Zestawienie otrzymanych wyników**

**3) Analiza i dyskusja błędów uczenia i testowania oraz wyłonionych cech dla wyników opracowanej sieci w zależności od wartości współczynnika uczenia**

Jak widać na powyższych wynikach, ilość epok jaka była potrzebna do nauczenia sieci znacząco różniła się w poszczególnych przypadkach. Niekiedy wynik wynosił tylko 1 epokę, a niekiedy było to ponad 200 epok. Tak więc jeśli chodzi o szybkość uczenia, widać, że nie jest to dobra miara ocenienia jakości uczenia sieci, gdyż jest to spowodowane wyłącznie początkowymi wartościami wag neuronów, a te są losowe. Jeśli jednak spojrzeć na wykresy, to widać, że zarówno jeśli chodzi o wartości maksymalne, minimalne czy też średnie, wraz ze wzrostem współczynnika uczenia, wartość ilości epok potrzebnych do nauczenia sieci stale spada.

Jeśli jednak chodzi o dobór współczynników uczenia oraz współczynników zapominania, to widać dosyć wyraźne zależności. Dla wersji ze współczynnikiem zapominania najlepiej spojrzeć na wykresy średniej procentowej poprawności uczenia się w zależności od współczynnika uczenia oraz współczynnika zapominania. Wykresy te przypominają trochę rozkład normalny Gaussa i widać, że zdecydowanym zwycięzcą, zarówno w wersji uczenia ze współczynnikiem zapominania jak i bez niego, jest współczynnik uczenia równy 0,1. Dodatkowo zwycięzcą spośród wszystkich trzech wersji jest kombinacja: współczynnik uczenia równy 0,1 oraz współczynnik zapominania równy 1/6 wartości współczynnika uczenia.

**4) Sformułowanie wniosków**

Na podstawie powyższych wyników można wnioskować, iż najlpesze wyniki można uzyskać stosując metodę modyfikacji wag ze współczynnikiem zapominania. Najlepszy wynik osiągnęła sieć o współczynniku uczenia równym 0,1 oraz o współczynniku zapominania równym 1/6 wartości współczynnika uczenia. Należy jednak rozważnie dobierać współczynnik zapominania, ponieważ jeśli będzie on zbyt duży, to sieć w trakcie nauki zbyt szybko zacznie zapominać tego czego dopiero się nauczyła. Jeśli jednak będzie on zbyt mały, również może negatywnie wpłynąć na otrzymane wyniki. Sieć ucząca się bez współczynnika zapominania również osiągała dobre wyniki, jednak nie były one aż tak obiecujące jak w powyższym przypadku.

Należy również pamiętać o tym aby normalizować wagi neuronów. Podczas wykonywania ćwiczenia wykonywałem również testy dla sieci bez normalizacji wag, jednak wyniki były daleko od poprawnych. Dzieje się tak, ponieważ bez normalizacji, wagi mogą rosnąć w nieskończoność.

Jeśli chodzi o sam proces testowania zaszumionych emotikonów, widać, iż podczas testów prawie zawsze pojawiał się jakiś błąd. Spowodowane jest to tym, że emotikony są do siebie bardzo podobne – różnią się tylko praktycznie kilkoma pikselami. Dodatkowo sam rozmiar emotikonów pozostawia wiele do życzenia. Jest to bowiem rozmiar jedynie 8x8 pikseli, tak więc nie pozwala to na zbyt dużą różnorodność emotikonów. Gdyby emotikony miały większą rozdzielczość to podczas uczenia można by zastosować więcej wzorców jednej emotikony, które byłyby do siebie zbliżone. Wtedy sieć byłaby w sanie lepiej sklasyfikować emotikony, a i podczas testowania wyniki byłyby dokładniejsze.

**5) Listing z komentarzami całego kodu programu**

**Bibliografia:**

S. Osowski „Sieci neuronowe do przetwarzania informacji”

<https://en.wikipedia.org/wiki/Iris_flower_data_set>