|  |  |
| --- | --- |
| *Nikodem Kirsz 236559*  *Oskar Trela 236677* | *czwartek, 13:30*  *26.05.2022* |

**Sztuczna inteligencja i systemy ekspertowe**

Zadanie: Dopasowanie funkcji za pomocą sieci neuronowej

1. **Wstęp**

Celem zadania było zaimplementowanie oraz przetestowanie sieci neuronowej typu MLP (*Multi-layer Perceptron*). Sieć ma charakter uniwersalny gwarantujący poprawność działania i nauki perceptronu bez względu na liczbę warstw i neuronów. Neurony ukryte oraz wyjściowe wykorzystują sigmoidalną funkcję aktywacji (współczynnik nachylenia jest równy 1). Zgodnie z zaleceniami wagi sieci są inicjalizowane w sposób pseudolosowy z zakresu [-0.5; 0.5].

Program działa w dwóch trybach: tryb nauki oraz tryb testowania.

Tryb nauki korzysta z metody online, która to stara się dostosować sieć do odpowiedniego zestawu danych po każdym jednym wzorcu zamiast na całym zestawie naraz tak jak w przypadku metody offline. W ten sposób mamy możliwość przesyłania danych sekwencyjnie a model jest aktualizowany przy każdej zmianie danych wejściowych.

Nauka sieci polega na podaniu wartości na wejście sieci, a następnie sprawdzenie zgodności ze wzorcem wartości wyjściowych (*Forward propagation*). W tym momencie obliczany jest błąd i rozpoczyna się praca algorytmu propagacji w tył (*Backpropagation algorithm*). Błąd obliczony na wyjściu jest przekazywany do warstwy poprzedzającej dzięki czemu wagi oraz obciążenia są dostosowywane do poprawnego wyjścia. Algorytm jest propagowany w ten sposób aż do warstwy wejściowej.

Dodatkiem do naszej implementacji jest momentum, dzięki któremu gradient zejścia zmniejsza swój szum. Skutkiem tego jest zwiększona wydajność nauczania sieci.

Nasz wygenerowany model sieci będziemy uczyć na zbiorze danych Irysów. Drugi model będzie reprezentantem sieci typu autoenkoder – wzorce zostały przekazane w treści zadania.

Znalezienie najstabilniejszej oraz najwydajniejszej sieci polega na wykonaniu dużej ilości eksperymentów, tj. sprawdzenie wydajności sieci dla różnych ilości warstw, czy też sprawdzenie wydajności dla różnej ilości neuronów w warstwie.

1. **Badania**

Implementacja miała nam posłużyć do realizacji dwóch zadań: klasyfikacji zbioru irysów oraz autoasocjacji.

(wykresy - errors)

Porównując szybkość uczenia z momentum oraz bez gołym okiem możemy zobaczyć różnicę. Przewaga po stronie nauki z tym parametrem wynika ze zmniejszonego szumu przy wyznaczaniu gradientu.

(wykresy albo tabele – różne modele sieci, nadal errors)

Jako najstabilniejszy model sieci dla danych treningowych oraz testowych Irysów przebadaliśmy architekturę sieci 4-3-3. Błąd całej sieci zachowywał się w granicach poniżej 1% i kończył proces nauki w 100-200 epokach.

(wykresy – shuffle oraz nie shuffle)

Dzięki uporządkowanym danym sieć jest w stanie szybciej zaadaptować się do zbioru wzorców. Jest to przyczyną tego, że dla dużej liczby kolejnych danych algorytm online dostosowuje wagi dla konkretnego wzorca danych wyjściowych. W przeciwieństwie do pomieszanego zbioru danych, gdzie sieć dostosowuje swoje wagi raz do jednego wzorca wyjściowego, a raz do drugiego.

Dane Irysów zostały podzielone na zestaw treningowy oraz zestaw testowy dla rzetelniejszego przebadania powodzenia nauki naszej sieci. Najlepsza sieć jaką udało nam się uzyskać posiada 0.51 błędu całkowitego, co przekłada się na bardzo dobre wyniki przewidywania poprawnych danych wyjściowych. W przypadku danych testowych, które nie były częścią danych przekazanych w procesie trenowania sieci, uzyskaliśmy 100% poprawnych wyników dla każdej klasy kwiatu.

(wykres – przykład przeuczenia)

Przeuczenie to zjawisko podczas którego sieć traci swoją wydajność na podanym zbiorze w wyniku zbyt dużej ilości generacji.

1. **Wnioski**

* Zbyt duża ilość warstw powoduje skomplikowanie kalkulacji dotyczących dostosowania wag do wzorca, tym samym do niepoprawnej nauki sieci.