Zadanie 4 - Perceptron wielowarstwowy (aproksymacja)

Marcin Pietrzykowski

mpietrzykowski@wi.zut.edu.pl

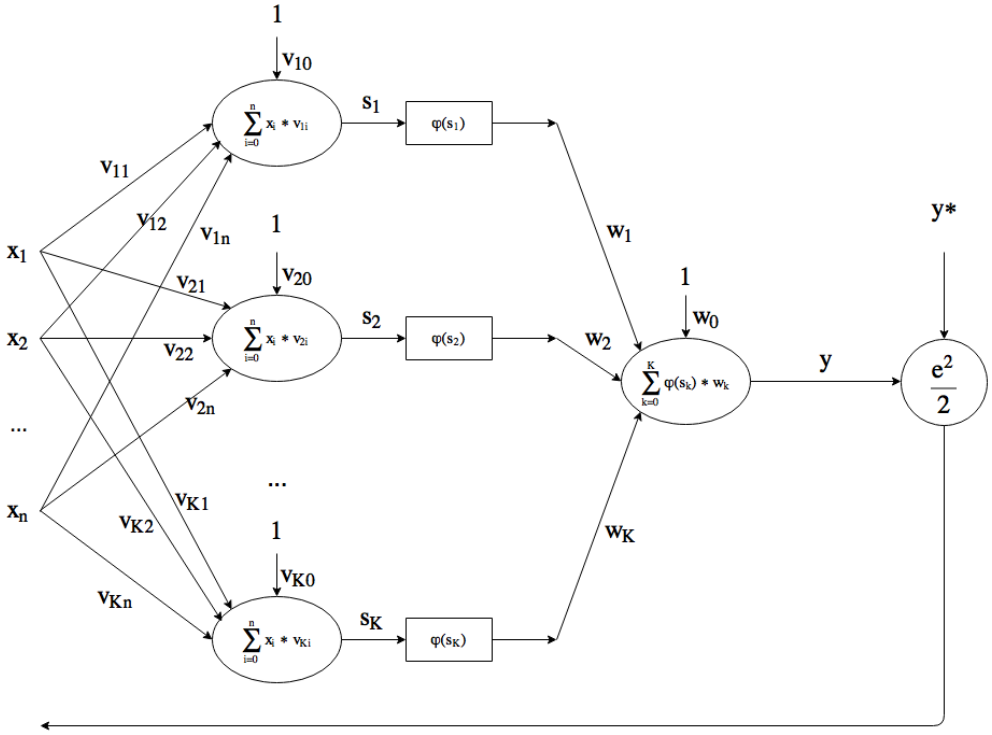
wersja 1.0

1 Cel

Celem zadania jest zapoznanie się zadaniem aproksymacji danych i algorytmem Perceptronu Wielowarstwowego. Program jest do napisania w środowisku Matlab. Osoby chętne chcące napisać ww. program w innym języku odsyłam do punktu 5. Wszystkie informacje zostały podane na zajęciach. Przypomnienie najważniejszych informacji znajduje się poniżej.

2 Architektura Sieci

Architektura sieci neuronowej w przypadku ogólnym ma następującą postać:



Rysunek 1: Schemat perceptronu wielowarstwowego.

Wzory opisujące działanie sieci neuronowej są następujące:

$$s_k = v_{k0} + \sum_{k=1}^{n} v_{ki} * x_i \tag{1}$$

$$\phi(s_k) = \frac{1}{1 + e^{-s_k}} \tag{2}$$

$$y = w_0 + \sum_{K}^{k=1} w_k * \phi(s_k) \tag{3}$$

gdzie przez k rozumiemy dowolny neuron o indeksie z zakresu od $1 \dots K$.

3 Uczenie sieci

Wzór ogólny na uczenie sieci jest następujący:

$$g_{\text{nowe}} = g_{\text{stare}} + \eta \left(-\frac{\partial \frac{1}{2} e^2}{\partial g_{\text{stare}}} \right) \tag{4}$$

gdzie przez g rozumiemy dowolny parametr sieci a przez błąd e:

$$\frac{1}{2} e^2 = \frac{1}{2} (y - y*)^2 \tag{5}$$

gdzie y to odpowiedź sieci neuronowej, a y^* to wartość oczekiwana dla danych wejściowych.

Wyprowadzenie wzorów na poprawkę wag v i w zostały podane na zajęciach. Wzory na korekcję wag sieci algorytmem *wstecznej propagacji błędów* w finalnej postaci gotowej do użycia są następujące:

$$v_{ki} = v_{ki} - \eta * (y - y^*) * w_k * \phi(s_k) \left(1 - \phi(s_k)\right) * x_i \tag{6}$$

$$w_k = w_k - \eta * (y - y^*) * \phi(s_k) \tag{7}$$

4 Polecenie

4.1 Na zajęciach

- Napisać skrypt generujący zbiór danych (zbiór próbek) pochodzących z funkcji dwóch zmiennych $y(x_1, x_2) = \cos(x_1 * x_2) * \cos(2 * x_1)$ zdefiniowanej na dziedzinie: x_1, x_2 należą do przedziału $[0, \pi]$. Przyjąć rozmiar zbioru danych I = 1000. Zbiór danych przechować w macierzy o wymiarach I x 3, gdzie kolejne kolumny będą odpowiadały zmiennym x_1, x_2, y.
- Za pomocą funkcji MATLABa `scatter3` i `surf` sporządzić wykresy odpowiednio: zbioru próbek, funkcji aproksymowanej.
- Napisać skrypt realizujący uczenie sieci neuronowej typu perceptron wielowarstwowy (Multi-Layer Perceptron) zgodnie z wiadomościami z wykładu. Skrypt powinien przyjmować na wejście następujące argumenty: zbiór danych, zadaną liczbę neuronów, zadaną liczbę kroków uczenia, współczynnik uczenia. Skrypt powinien zwracać na wyjściu macierz z nauczonymi wartościami wag V i wektor nauczonych wag W.
- Przeprowadzić uczenie i odebrać wyniki. Sugerowane ustawienia (rzędy wielkości): liczba kroków uczenia $T \sim \{10^5, ..., 10^6\}$, liczba neuronów $K \sim \{10, 20, ..., 100\}$, współczynnik uczenia $\eta \sim \{10^-3, ..., 10^-1\}$. Początkowe wartości wylosowanych macierzy V, W powinny być bardzo małe $\sim \{-10^{-3}, 10^{-3}\}$ (lub jeszcze mniejszy rząd wielkości).
- Napisać skrypt rysujący (`surf`) wykres powierzchni sieci neuronowej reprezentowanej przez V, W jako funkcji x_1, x_2. Ustalić zakres osi odpowiadający zakresom funkcji aproksymowanej.
- Wyświetlić oba wykresy powierzchni: funkcji aproksymowanej i sieci neuronowej (funkcji aproksymującej) i porównać podobieństwo wizualnie - "na oko", np nakładając oba wykresy na siebie.

4.2 Do domu

W domu należy przeprowadzić następujący eksperyment:

- Zaczerpnąć z aproksymowanej funkcji nowy zbiór uczący o rozmiarze I = 200, przy czym wartości y należy obciążyć pewnym losowym błędem, tj. $y = y(x_1, x_2) + \epsilon$, gdzie $\epsilon \sim N(0, 0.2)$ - błąd losowy o rozkładzie normalnym o średniej zero i odchyleniu standardowym 0.2. W MATLABie jest funkcja `randn()` losująca liczbę (lub macierz liczb) losową z rozkładu normalnego.

- Powyższy zbiór należy podzielić losowo na część uczącą i część testową w proporcji 70 : 30.
- W pętli (wielokrotnie) przeprowadzić proces uczenia sieci zadając coraz większą liczbę neuronów: $K = 10, 20, ..., 100$ (10 iteracji). Sieć ma być uczona tylko na zbiorze uczącym. Za każdym razem początkowe wartości wag V i W mają być wylosowane na nowo. W każdej z 10 iteracji, po nauczeniu sieci należy obliczyć błąd popełniany przez nią na zbiorze uczącym i na zbiorze testowym (nie widzianym podczas uczenia) jako średnią różnicę bezwzględną pomiędzy oczekiwanymi wartościami y a odpowiedziami sieci neuronowej. Obie wartości zapamiętać dla każdej iteracji pętli.
- Po zakończeniu pętli narysować wykres obu wielkości - błędów uczących i testowych dla kolejnych wartości K. Wskazać jaka liczba neuronów jest optymalna dla danego zbioru danych, tj. przy jakiej liczbie neuronów błąd na części testowej jest najmniejszy.
- Nauczyć ostatecznie sieć nauronową już na całym zbiorze danych (a nie tylko na części uczącej) podając optymalną liczbę neuronów K.
- Aby wyznaczyć w sposób dokładny błąd popełniany przez finalną nauczoną sieć względem aproksymowanej funkcji na całej dziedzinie należałoby np. wyliczyć całkę z bezwzględnej różnicy obu funkcji (lub kwadratu różnicy) i podzielić wynik przez miarę dziedziny (tu: π^2). Nie będzie trzeba tego robić, natomiast trzeba oszacować ten błąd poprzez przybliżenie w/w całki sumą o odpowiednio dużej liczbie składników. Należy pobrać z funkcji dodatkowy duży zbiór próbek (np. o rozmiarze rzędu 10^4) i na jego podstawie należy policzyć średni błąd bezwzględny pomiędzy oczekiwanymi wartościami y a odpowiedziami dostarczanymi przez sieć neuronową dla testowych punktów (x_1, x_2).

5 Wersją do osób nielubiących Matlba

Osoby chętne mogą napisać ww. program w języku innym niż Matlab jednakże niezbędne będzie wyświetlenie wykresów 2D i 3D. Problem można rozwiązać na kilka sposobów.

- Skorzystanie z biblioteki rysującej wykresy natywnej dla danego języka.
- Napisanie programu w dowolnym języku, a następnie wklejenie wyników do skryptu Matlaba w celu wyrysowania wykresów.
- Skorzystania z obiektu COM Matlaba do rysowanie wykresów bezpośrednio w kodzie programu. Przykład jak można to wykonać w aplikacji konsolowej napisanej w języka C# jest do pobrania tutaj. Zaprezentowane rozwiązanie wymaga instancji Matlaba zainstalowanej na maszynie, na której uruchamiany jest program.