Perceptron wielowarstwowy

Multilayer perceptron (MLP)

- sieć jednokierunkowa
- MLP składa się z co najmniej 3 warstw neuronów - warstwy wejściowej, ukrytej i wyjściowej
- wykorzystuje się jako klasyfikatory w problemach liniowo nieseparowalnych
- sieć uczymy za pomocą zbióru treningowego par wejścia-wyjścia $\{\xi_k^\mu, \zeta_i^\mu\}$
- podstawowa metoda uczenia opiera się na spadku gradientu (wsteczna propagacja błędu)

Budowa MLP

Oznaczenia

- indeks i zawsze dotyczy warstwy wyjściowej, j ukrytej, a k wejściowej
- indeks μ oznacza numer wzorca ze zbioru uczącego
- O_i - neurony wyjściowe
- V_j - neurony ukryte
- ξ_k - neurony wejściowe
- W_{ij} - wagi połączeń neuronów wyjściowych z ukrytymi
- w_{jk} - wagi połączeń neuronów ukrytych z wejściowymi

Uczenie sieci wielowarstwowej

Propagacja sygnału

Dla danego wzorca μ na ukryty neuron j działa pole lokalne:

$h_j^\mu = \sum_k w_{jk}\xi_k^\mu$

Neuron ten wytwarza sygnał:

$V_j^\mu = g(h_j^\mu) = g\left(\sum_k w_{jk}\xi_k^\mu\right)$

$g(h)$ jest funkcją aktywacji.

Na neuron wyjściowy i działa więc pole lokalne:

$h_i^\mu = \sum_j W_{ij}V_j^\mu = \sum_j W_{ij}g\left(\sum_k w_{jk}\xi_k^\mu\right)$

Neuron i z warstwy wyjściowej wytwarza sygnał:

$O_i^\mu = g\left(h_i^\mu\right) = g\left(\sum_j W_{ij}g\left(\sum_k w_{jk}\xi_k^\mu\right)\right)$

Wsteczna propagacja błędu

Uczenie sieci polega na dostosowaniu wag połączeń między neuronami w celu zminimalizowania błędu, czyli funkcji różnicy sygnałów wyjściowych i wyjściowych wektorów uczących.

Definiujemy średni błąd kwadratowy (*mean squared error - MSE*):

$E(w) = \frac{1}{2}\sum_{i\mu}\left(\zeta_i^\mu - O_i^\mu\right)^2$

$E(w) = \frac{1}{2}\sum_{i\mu}\left(\zeta_i^\mu - g\left(\sum_j W_{ij}g\left(\sum_k w_{jk}\xi_k^\mu\right)\right)\right)^2$

$E(w) = \frac{1}{2}\sum_{i\mu}\left(\zeta_i^\mu - g\left(\sum_j W_{ij}V_j^\mu\right)\right)^2$

Stosujemy algorytm spadku gradientu, zaczynając od modyfikacji wag między warstwą wyjściową a ukrytą:

$\Delta W_{ij} = -\eta\frac{\partial E}{\partial W_{ij}} =$

$= \eta\sum_\mu\left(\zeta_i^\mu - O_i^\mu\right) g\prime(h_i^\mu)V_j^\mu = \eta\sum_\mu \delta_i^\mu V_j^\mu$

gdzie $\delta_i^\mu = g\prime(h_i^\mu)\left(\zeta_i^\mu - O_i^\mu\right)$.

Parametr η określa szybkość spadku.

Następnie stosujemy regułę łańcuchową, by zmodyfikować wagi między warstwami ukrytą a wejściową.

$\Delta w_{jk} = -\eta\frac{\partial E}{\partial w_{jk}} = -\eta\sum_\mu\frac{\partial E}{\partial V_j^\mu}\frac{\partial V_j^\mu}{\partial w_{jk}} =$

$= \eta\sum_{\mu i}\left(\zeta_i^\mu - O_i^\mu\right) g\prime(h_i^\mu)W_{ij}g\prime(h_j^\mu)\xi_k^\mu =$

$= \eta\sum_{\mu i}\delta_i^\mu W_{ij}g\prime(h_j^\mu)\xi_k^\mu =$

$= \eta\sum_\mu \delta_j^\mu\xi_k^\mu$

gdzie $\delta_j^\mu = g\prime(h_j^\mu)\sum_i W_{ij}\delta_i^\mu$.

Uwagi końcowe

Funkcja aktywacji ma najczęściej postać jednej z funkcji sigmoidalnych, np.:

- $g(h) = tgh(\beta h)$
- $g(h) = \frac{1}{1+e^{-2\beta h}}$

Stosuje się również funkcję liniową z odcięciem.

- wejściowe wektory uczące ξ_k^μ mogą być binarne lub ciągłe.
- w zagadnieniach klasyfikacji najczęściej warstwa wyjściowa zawiera liczbę neuronów równą liczbie klas. Wtedy wzorcowi μ należącemu do klasy j odpowiada wektor wyjściowy o elementach $\zeta_i^\mu = \delta_{ij}$.
- by umożliwić klasyfikację zerowego stanu wejściowego, w warstwie wejściowej dodaje się dodatkowy neuron, który jest zawsze aktywny (*bias*). Alternatywne można dodać losowe progi w funkcji aktywacyjnej.
- istnieje wiele modyfikacji omówionej metody. Najczęstsze modyfikacje uwzględniają bezwładność, parametry stochastyczne oraz zmienną szybkość uczenia.