Uczenie Wielowarstwowych Sieci Neuronów o Ciągłej Funkcji Aktywacji

(c) Marcin Sydow

Plan

- uczenie neuronu o ciągłej funkcji aktywacji
- uczenie jednowarstwowej sieci neuronów o ciągłej funkcji aktywacji
- uczenie sieci wielowarstwowej - metoda propagacji wstecznej

Uczenie neuronu o ciągłej funkcji aktywacji

Przypomnienie: neuron z ciągłą funkcją aktywacji

- sigmoidalna funkcja unipolarna:

$$f(net) = \frac{1}{1 + e^{-net}}$$

- sigmoidalna funkcja bipolarna:

$$f(net) = \frac{2}{1 + e^{-net}} - 1$$

gdzie:

$$net = w^T x$$

Błąd neuronu ciągłego

Zdefiniujmy następującą miarę błędu pojedynczego neuronu:

$$E = \frac{1}{2}(d - y)^2 = \frac{1}{2}(d - f(w^T x))^2$$

gdzie:

- d - pożądane wyjście (ciągłe)
- y - aktualne wyjście (ciągłe) (y = f(net))

(współczynnik 1/2 wybrany dla uproszczenia późniejszych rachunków)

Cel uczenia: minimalizacja błędu

Chcemy tak modyfikować wektor wag w, żeby zminimalizować błąd.

Metoda gradientu: największy spadek funkcji (w kierunku minimum) wskazywany jest przez przeciwny wektor gradientu (czyli pochodnych cząstkowych błędu jako funkcji wektora wag)

$$\nabla E(w) = \frac{\partial E}{\partial w}$$

$$\nabla E(w) = -(d-y)f'(net)(\frac{\partial net}{\partial w_1}, \ldots, \frac{\partial net}{\partial w_p})^T =$$

$$= -(d-y)f'(net)x$$

Pochodne funkcyj sigmoidalnych

Zauważmy, że:

- dla funkcji sigmoidalnej unipolarnej:

$$f'(net) = f(net)(f(net) - 1) = y(y - 1)$$

- dla funkcji sigmoidalnej bipolarnej:

$$f'(net) = \frac{1}{2}(1 - f^2(net)) = \frac{1}{2}(1 - y^2)$$

Czyli pochodna funkcji f jest łatwo wyrażalna przez samą funkcję f.

(Teraz można zrozumieć dlaczego zaproponowano akurat takie formy ciągłych funkcyj aktywacji)

Reguła uczenia dla ciągłego neuronu

Reasumując, wagi neuronu ciągłego modyfikujemy zgodnie ze wzorem:

- unipolarny:

$$w_{new} = w_{old} + \eta(d - y)y(1 - y)x$$

- bipolarny:

$$w_{new} = w_{old} + \frac{1}{2}\eta(d - y)(1 - y^2)x$$

gdzie: η to współczynnik uczenia (learning rate)

Zauważmy wyraźną analogię do reguły delta dla dyskretnego perceptronu

Uczenie jednowarstwowej sieci neuronów o ciągłej funkcji aktywacji

Jednowarstwowa sieć neuronów ciągłych

Załóżmy, że sieć ma J wejść i K neuronów ciągłych.

Wprowadźmy następujące oznaczenia:

- wektor wejść: $y^T = (y_1, \ldots, y_J)$
- wektor wyjść: $z^T = (z_1, \ldots, z_K)$
- macierz wag: $W = [w_{kj}]$ (w_{kj}: k-ty neuron, j-ta waga)
- macierz funkcyj aktywacji: $\Gamma = diag[f(\cdot)]$ (wymiar: $K \times K$)

Obliczenie wyjścia można więc teraz zapisać jako:

$$z = \Gamma[Wy]$$

Uczenie jednowarstwowej sieci ciągłych neuronów

Wprowadźmy dodatkowe oznaczenia:

- pożądany wektor wyjściowy: $d^T = (d_1, \ldots, d_K)$
- błąd wyjścia dla pojedynczego wektora wejść:

$$E = \frac{1}{2}\sum_{k=1}^{K}(d_k - z_k)^2 = \frac{1}{2}||d - z||^2$$

Zastosujemy ponownie metodę gradientu (jak dla pojedynczego neuronu).

Zmiana pojedynczej wagi dana jest więc wzorem:

$$\Delta w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}}$$

Uczenie jednej warstwy, cd.

Mamy więc:

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial net_k} \frac{\partial net_k}{\partial w_{kj}}$$

sygnał błędu delta k-tego neuronu ostatniej warstwy:

$$\delta_{zk} = -\frac{\partial E}{\partial net_k} = (d_k - z_k) z_k (1 - z_k)$$

$$\delta_{zk} = -\frac{\partial E}{\partial net_k} = \frac{1}{2}(d_k - z_k)(1 - z_k)^2$$

Zauważmy, że $\frac{\partial net_k}{\partial w_{kj}} = y_j$

Otrzymujemy więc wzór modyfikacji wag w postaci macierzowej:

$$W_{new} = W_{old} + \eta \delta_z y^T$$

Algorytm uczenia jednej warstwy

- wybór η, E_{max}, inicjalizacja losowych wag W, $E = 0$
- dla każdego przykładu ze zbioru uczącego:
  - oblicz wyjście z
  - zmodyfikuj wagi k-tego neuronu (unipolarny/bipolarny):

$$w_k \leftarrow w_k + \eta(d_k - z_k)z_k(1 - z_k)y$$

$$w_k \leftarrow w_k + \frac{1}{2}\eta(d_k - z_k)(1 - z_k^2)y$$

  - kumuluj błąd:

$$E \leftarrow E + \frac{1}{2}\sum_{k=1}^{K}(d_k - z_k)^2$$

- jeśli pokazano wszystkie elementy zbioru uczącego i $E < E_{max}$ to zakończ uczenie. W przeciwnym wypadku wyzeruj E i ponownie wykonaj uczenie na całym zbiorze uczącym

Uczenie sieci wielowarstwowej - metoda propagacji wstecznej

Sieć wielowarstwowa

Jedna warstwa sieci neuronowej ma możliwości podzielenia przestrzeni obrazów wejściowych na obszary liniowo separowalne.

Każda następna może dokonywać kolejnych transformacji.

W efekcie, wielowarstwowa sieć neuronowa jest uniwersalnym narzędziem, które teoretycznie może dowolnie dokładnie aproksymować dowolne transformacje przestrzeni wejściowej w przestrzeń odwzorowań wyjściowych.

Uczenie Sieci Wielowarstwowej

Zilustrujemy uczenie sieci wielowarstwowej na przykładzie sieci 2-warstwowej. W tym celu dodamy jedną warstwę umieszczoną "przed" ostatnią (wyjściową) warstwą sieci i pokażemy jak ją uczyć.

Każda warstwa poza wyjściową nazywana jest warstwą ukrytą, gdyż nie jest wiadome jakie powinno być jej "prawidłowe" wyjście.

Metodę uczenia sieci wielowarstwowej odkryto dopiero w latach 70. i zaczęto stosować w latach 80. XX. wieku - nazywa się ona metodą **wstecznej propagacji błędów**, gdyż wagi modyfikuje się od warstwy ostatniej do pierwszej (wstecz).

Metodę tę można naturalnie rozszerzać z sieci 2-warstwowej na dowolną liczbę warstw ukrytych.

Sieć dwuwarstwowa

Wprowadzimy następujące oznaczenia:

- wektor wejść: $x^T = (x_1, \ldots, x_I)$
- macierz wag pierwszej warstwy: $V = [v_{ji}]$
  (v_{ji}: j-ty neuron, i-ta waga)
- wektor wyjść pierwszej (wejść drugiej) warstwy:
  $y^T = (y_1, \ldots, y_J)$
- wektor wyjść drugiej warstwy (całej sieci):
  $z^T = (z_1, \ldots, z_K)$
- macierz wag drugiej warstwy: $W = [w_{kj}]$
  (w_{kj}: k-ty neuron, j-ta waga)
- operator funkcyj aktywacji: $\Gamma = diag[f(\cdot)]$
  (wymiar: $J \times J$ lub $K \times K$)

Obliczenie wektora wyjść można więc teraz zapisać jako:

$$z = \Gamma[Wy] = \Gamma[W\Gamma[Vx]]$$

Uczenie sieci wielowarstwowej

Metoda wstecznej propagacji błędu:

Po obliczeniu wektora wyjść z, wagi modyfikowane są od ostatniej do pierwszej warstwy (wstecz).

Pokazano wcześniej, jak modyfikować wagi ostatniej warstwy.

Po zmodyfikowaniu wag ostatniej warstwy, modyfikowane są wagi warstwy drugiej od końca (itd.)

Przy modyfikowaniu wag warstwy drugiej od końca stosuje się również metodę gradientu:

$$\Delta v_{ji} = -\eta \frac{\partial E}{\partial v_{ji}}$$

Metoda wstecznej propagacji błędu, cd.

Przez analogię, wagi V modyfikowane są następująco:

$$V_{new} = V_{old} + \eta \delta_y x^T$$

gdzie, δ_y oznacza *wektor sygnału błędu* warstwy ukrytej:

$$\delta_y^T = (\delta_{y_1}, \ldots, \delta_{y_J})$$

Sygnał błędu warstwy ukrytej obliczamy następująco:

$$\delta_{yj} = -\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial net_j} = -\frac{\partial E}{\partial y_j} \cdot f'(net_j) = \sum_{k=1}^{K} \delta_{zk} w_{kj} \cdot f_j'(net_j)$$

Algorytm uczenia sieci dwuwarstwowej

- wybór η, E_{max}, inicjalizacja losowych wag W i V, $E = 0$
- dla każdego przykładu ze zbioru uczącego:
  - oblicz kolejno wektory wyjść y oraz z
  - kumuluj błąd: $E \leftarrow E + \frac{1}{2}\sum_{k=1}^{K}(d_k - z_k)^2$
  - oblicz sygnały błędów (ostatniej, pierwszej warstwy):
    unipolarny:
    $\delta_{zk} = (d_k - z_k)z_k(1 - z_k),\ \delta_{yj} = y_j(1 - y_j)\sum_{k=1}^{K}\delta_{zk}w_{kj}$
    bipolarny:
    $\delta_{zk} = \frac{1}{2}(d_k - z_k)(1 - z_k^2),\ \delta_{yj} = \frac{1}{2}(1 - y_j^2)\sum_{k=1}^{K}\delta_{zk}w_{kj}$
  - zmodyfikuj wagi ostatniej warstwy:
    $w_{kj} \leftarrow w_{kj} + \eta\delta_{zk}y_j$
  - zmodyfikuj wagi pierwszej (ukrytej) warstwy:
    $v_{ji} \leftarrow v_{ji} + \eta\delta_{yj}x_i$
- jeśli pokazano wszystkie elementy zbioru uczącego i $E < E_{max}$ to zakończ uczenie. W przeciwnym wypadku wyzeruj E i ponownie wykonaj uczenie na całym zbiorze uczącym

Zagadnienia do przyswojenia:

- uczenie neuronu o ciągłej funkcji aktywacji
- uczenie jednowarstwowej sieci neuronów o ciągłej funkcji aktywacji
- uczenie sieci wielowarstwowej - metoda propagacji wstecznej

Dziękuję za uwagę.