Sieci neuronowe i aplikacje sztucznej inteligencji — metoda uczenia RPROP

1 Metoda RPROP

Algorytm RPROP (od ang. *Resilient backPROPagation*) opracowali Riedmiller i Braun (1992). Jest to algorytm przeznaczony dla *pełnego* lub inaczej *wsadowego* trybu korekcji parametrów (wag). Oznacza to, że jedno skorygowanie parametrów (wag) następuje dopiero po przeglądnięciu przez sieć całego zbioru uczącego i obliczeniu sumarycznego a tym samym dokładnego gradientu (a nie jak było to trybie on-line po każdej próbce[1]).

Kluczowymi elementami algorytmu RPROP są: wykorzystywanie jedynie samego znaku każdej składowej gradientu (natomiast wartości są pomijane), a także modyfikowanie współczynnika (współczynników) uczenia w każdym kroku. Współczynnik uczenia jest zwiększany, gdy znaki kolejnych gradientów pozostają zgodne, natomiast zmniejszany (a dokładnie połowiony), gdy są różne. Należy jednocześnie zwrócić uwagę, że w tej metodzie mamy tak naprawdę wiele współczynników uczenia — każdy z nich skojarzony jest ze swoim jednym parametrem (wagą) — innymi słowy z każdą składową gradientu. I dodatkowo, jak powiedziano wcześniej, każdy z tych współczynników uczenia nie jest stały, a dynamiczny w procesie uczenia.

Niech $v(t)$ oznacza dowolny parametr (wagę) sieci na kroku uczenia o numerze t. Wzór na korektę każdego takiego parametru ma postać:

$$v(t+1) = v(t) - \eta(t)\,\mathrm{sgn}\Big(\frac{\partial E^2(t)}{\partial v(t)}\Big), \tag{1}$$

gdzie $E^2(t)$ oznacza sumaryczny błąd kwadratowy (suma po całym zbiorze uczącym), a tym samym $\frac{\partial E^2(t)}{\partial v(t)}$ reprezentuje dokładną wartość składowej v gradientu. Pochodna sumy to suma pochodnych, a zatem:

$$\frac{\partial E^2(t)}{\partial v(t)} = \sum_{i=1}^{p} \frac{\partial e_i^2(t)}{\partial v(t)}, \tag{2}$$

gdzie p oznacza rozmiar zbioru uczącego.

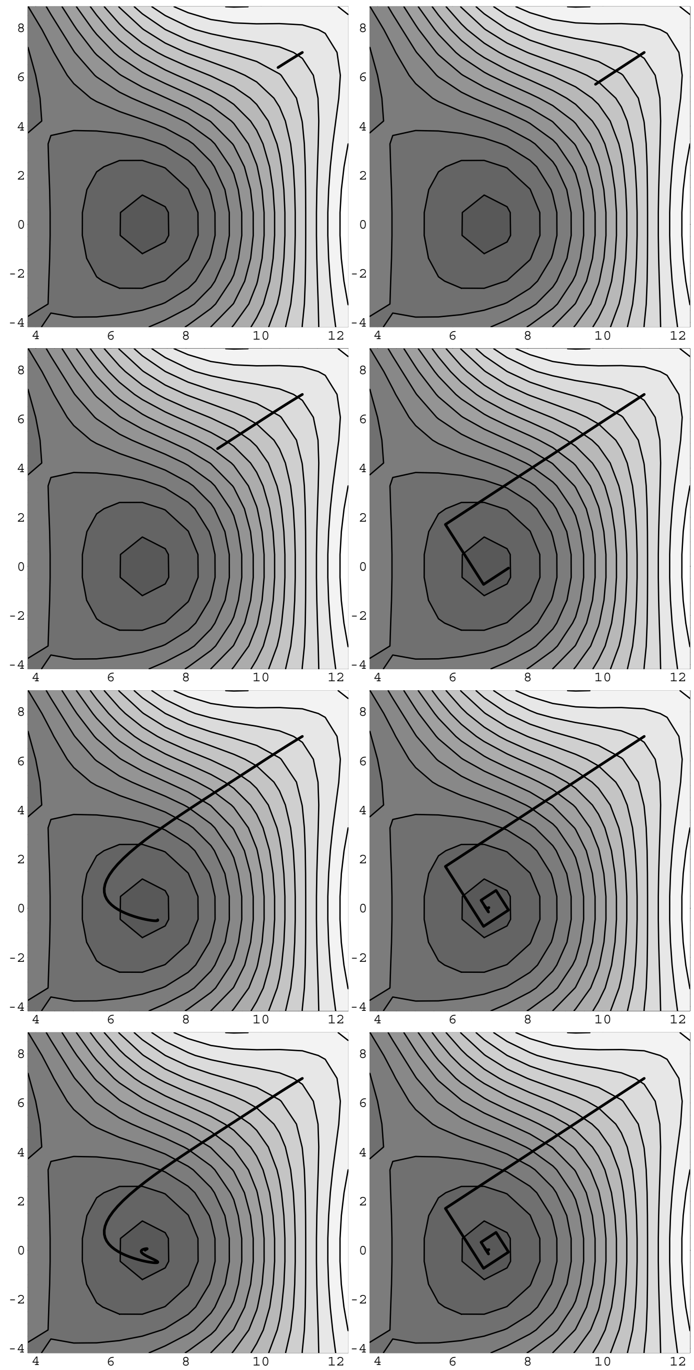
Wzór na modyfikację każdego ze współczynników uczenia ma postać:

$$\eta(t) = \begin{cases} \min\{a\eta(t-1), \eta_{\mathsf{max}}\}, & \text{dla } \frac{\partial E^2(t)}{\partial v(t)}\frac{\partial E^2(t-1)}{\partial v(t-1)} > 0; \\ \max\{b\eta(t-1), \eta_{\mathsf{min}}\}, & \text{dla } \frac{\partial E^2(t)}{\partial v(t)}\frac{\partial E^2(t-1)}{\partial v(t-1)} < 0; \\ \eta(t-1), & \text{w innym przypadku.} \end{cases} \tag{3}$$

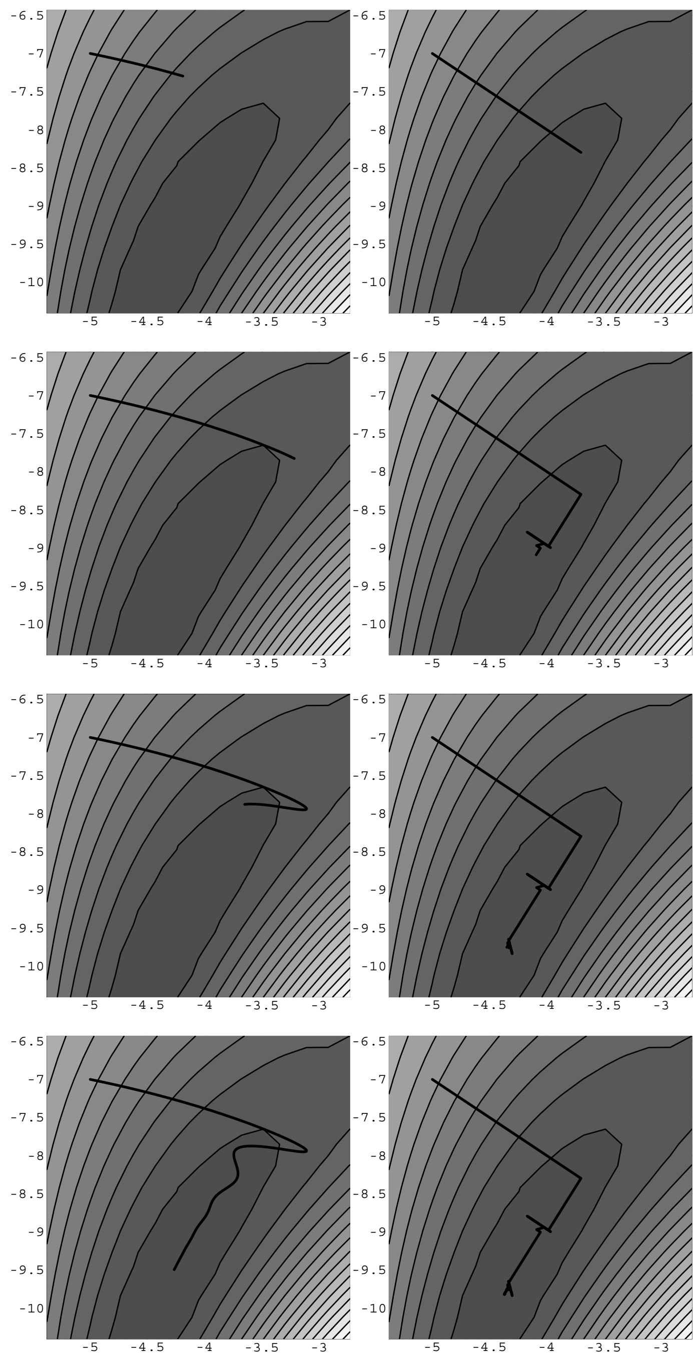
[1] W trybie on-line nie mamy do czynienia z dokładnym gradientem, a jedynie z jego składnikami ze względu na pojedyncze próbki

Przyjmuje się następujące stałe $a = 1.2, b = 0.5$, oraz $\eta_{\mathsf{min}} = 10^{-6}, \eta_{\mathsf{max}} = 50$. Przy starcie algorytmu $\eta(0)$ można ustawić na stosunkowo małą wartość wybraną według uznania, np. $\eta(0) = 0.05$.

Algorytm RPROP powoduje znaczne przyspieszenie procesu uczenia zwłaszcza w obszarach o niewielkim nachyleniu powierzchni błędu. Rysunki 1 i 2 ilustrują przykładowe porównania metody RPROP w zestawieniu z pewnym wariantem metody back-propagation z tzw. rozpędem (momentum back-propagation).



Rysunek 1: Porównanie działania metody momentum back-propagation (po lewej stronie) z metodą RPROP (po prawej stronie) w pewnej przestrzeni dwóch parametrów (wag). Kolejne wiersze od góry do dołu pokazują stan uczenia odpowiednio po: $10, 20, 60, 120$ krokach. *Źródło: opracowanie własne.*



Rysunek 2: Porównanie działania metody momentum back-propagation (po lewej stronie) z metodą RPROP (po prawej stronie) w pewnej przestrzeni dwóch parametrów (wag). Kolejne wiersze od góry do dołu pokazują stan uczenia odpowiednio po: $10, 20, 40, 200$ krokach. *Źródło: opracowanie własne.*