Sieci neuronowe

Marek Zalewski

27 grudnia 2018

Streszczenie

Krótki opis działania i uczenia sieci metodą wstecznej propagacji błędu sieci neuronowej ukierunkowanej nierekurencyjnej, gdzie wszystkie neurony jednej warstwy są połączone z wszystkimi neuronami warstw sąsiednich.
Opis sieci ukierunkowanej nierekurencyjnej bez pełnego połączenia pomiędzy warstwami i z połączeniami pomiędzy neuronami dowolnych warstw (poza daną jedną warstwą) Opis pomysłu uczenia sieci neuronowej metodą genetyczną wspomaganą metodą kar i nagród (bez wykorzystania wstecznej propagacji błędu).

Spis treści

1 Wykaz zmiennych i funkcji **4**
- 1.1 Wykaz zmiennych 4
- 1.2 Wykaz funkcji 4

2 Obliczanie wyjścia sieci **5**
- 2.1 Funkcja aktywacji neuronu 5

3 Uczenie sieci **6**
- 3.1 Błąd standardowy 6
- 3.2 Cel uczenia sieci 6
- 3.3 Minimalizacja błędu standardowego 6

4 Uproszczenie gradientu funckcji błędu standardowego **7**
- 4.1 Wyliczenie gradientów dla neuronów warstwy wyjściowej 7
  - 4.1.1 Dla wag 7
  - 4.1.2 Dla biasów 7
- 4.2 Wyliczenie gradientów dla neuronów warstw ukrytych 8
  - 4.2.1 Dla warstwy L-2 8
  - 4.2.2 Dla warstwy L-3 9
  - 4.2.3 Rekurencyjny wzór dla biasów neuronów dowolnych wag ukrytych 10
  - 4.2.4 Rekurencyjny wzór dla wag neuronów dowolnych wag ukrytych 10

5 Efekt jo-jo **11**

6 Over-learning i under-learning **12**

7 Współczynnik uczenia **13**
- 7.1 Zmiana współczynnika uczenia w czasie 13

8 Uczenie genetyczne **14**
- 8.1 Sposób uczenia 14
  - 8.1.1 Mutacje 14
  - 8.1.2 Cross-over 14
  - 8.1.3 Funkcja dopasowania 14
- 8.2 System kar i nagród 14

9 Sieć warstwowa nie mająca w pełni połączonych dwuch warstw **15**

10 Przykładowa implementacja sieci neuronowej uczonej metodą wstecznej propagacji błędu **16**
- 10.1 Obliczanie wyjścia sieci 16
- 10.2 Obliczanie gradientów 16
- 10.3 Modyfikacja wag 16

1 Wykaz zmiennych i funkcji

1.1 Wykaz zmiennych

L	-	ilość warstw sieci neuronowej
N_l	-	ilość neuronów na l-tej warstwie
ω	-	zbiór wszystkich wag i biasów
$\omega_{l,n,i}$	-	i-ta waga n-tego neuronu na l-tej warstwie
$\omega_{l,n}$	-	zbiór wag i biasu n-tego neuronu na l-tej warstwie
ω_l	-	zbiór wag i biasów neuronów l-tej warstwy
$\beta_{l,n}$	-	bias n-tego neuronu na l-tej warstwie
a	-	pojedyńczy zbiór wejść sieci
$S_{l,n}$	-	suma ważona wejść neuronu n na warswie l
s_i	-	i-te wejście sieci neuronowej
t_i	-	i-te wyjście docelowe sieci neuronowej
$a_{l,n}$	-	wyjście n-tego neuronu l-tej warstwy
$a_{0,n}$	$=$	s_n
$a_{L-1,n}$	-	n-te wyjście sieci
a_l	-	zbiór wyjść neuronów na l-tej warstwie
α	-	współczynnik prędkości uczenia sieci
A	-	zbiór wejść zestawów uczących
T	-	zbiór wyjść oczekiwanych zestawów uczących
$\Delta\omega_{l,n,i}$	-	zmiana wagi $\omega_{l,n,i}$ do poprawienia aktualnego wyniku
$\Delta\beta_{l,n}$	-	zmiana biasu $\beta_{l,n}$ do poprawienia aktualnego wyniku
W	-	wektor wag i biasów
ΔW	-	wektor zmian wag i biasów
$g_{l,n}$	-	gradient dla biasu n-tego neuronu l-tej warstwy
$g_{l,n,i}$	-	gradient dla wagi n-tego neuronu l-tej warstwy

1.2 Wykaz funkcji

$S_{l,n}\left(a_{l-1},\omega_l\right)$	-	funkcja sumy ważonej wejść n-tego neuronu l-tej warswy
$a_{l,n}\left(a_{l-1},\omega_l\right)$	-	funkcja wyjscia neuronu
$\sigma\left(x\right)$	-	funkcja aktywacji neuronu
$\varepsilon\left(a_{l-1,n},t_i\right)$	-	funkcja błędu standardowego sieci

2 Obliczanie wyjścia sieci

2.1 Funkcja aktywacji neuronu

W całym opracowaniu będę przyjmował przykładową funckję aktywacji σ:

$$\sigma\left(x\right) = \frac{1}{1+e^{-x}} \tag{1}$$

$$\sigma'\left(x\right) = \frac{-\left(1+e^{-x}\right)'}{\left(1+e^{-x}\right)^2} = \frac{e^{-x}}{\left(1+e^{-x}\right)^2} = \sigma\left(x\right)\cdot\left(1-\sigma\left(x\right)\right) \tag{2}$$

Funkcji aktywaci jest dużo, nie potrafię stwierdzić która jest lepsza od której, natomiast mogę powiedzić że powyższa mi się podoba najbardziej.
Aby obliczyć wyjście danego neuronu, potrzeba znać wszystkie wyjścia neuronów z poprzedniej warstwy. Ponieważ $a_{0,n} = s_n$ to zaczynając od pierwszej warstwy do warstwy $L-1$ możemy korzystać z poniższych wzorów:

$$S_{l,n} = S_{l,n}\left(a_{l-1}, \omega_l\right) = \beta_{l,n} + \sum_{i=0}^{N_{l-1}-1} a_{l-1,i}\cdot\omega_{l,n,i} \tag{3}$$

$$a_{l,n} = a_{l,n}\left(a_{l-1}, \omega_l\right) = \sigma\left(S_{l,n}\right) = \sigma\Big(S_{l,n}\left(a_{l-1}, \omega_l\right)\Big) \tag{4}$$

3 Uczenie sieci

3.1 Błąd standardowy

Poprawność wyjścia sieci (czyli jej błąd standardowy) dla danego zestawu oczekiwanych wyjśś i wyjść wyliczonych przez sieć można określić wzorem:

$$\varepsilon\left(a_{L-1}, t_i\right) = \sum_{i=0}^{N_{L-1}-1} \left(a_{L-1,i} - t_i\right)^2 \tag{5}$$

3.2 Cel uczenia sieci

Uczeniem sieci neuronowej chcemy osiągnąć sytuację w której wyjście sieci dla danego wejścia będzie jak najmniej odbiegało od oczekiwanego wyjścia, czyli krócej, chcemy znaleźć minimum funkcji błędu standardowego. Ponieważ dla każdego zestawu wyjść oczekiwanych mamy inną funckje błędu standardowego. Oznacza to że musielibyśmy znaleźć minimum wszystkichh tych funkcji. Ponieważ te minima mogą być zupełnie różne dla różnych wyjść oczekiwanych, chcemy dążyć do minimów każdej z tych funkcji błędu standardowego.

3.3 Minimalizacja błędu standardowego

W jaki sposób możemy zminimalizować błąd dla jednego zestawu wejść i wyjść docelowych? Możemy modyfikować tylko wagi neuronów, czyli funkcję błędu standardowego będziemy minimalizować względem wag neuronów, a nie wejść sieci. Zmienimy argumenty funkcji błędu standardowego na jej parametry, a wagi neuronów zamienimy na argumenty funkcji błędu:

$$\varepsilon\left(\omega\right) = \sum_{i=0}^{N_{L-1}-1} \left(a_{L-1,i}\left(a_{L-2}, \omega_{L-1}\right) - t_i\right)^2 \tag{6}$$

Aby dążyć do minimum funkcji błędu standardowego musimy podążać w kierunku w którym ta funkcja maleje, czyli w kierunku ujemnego gradientu funkcji błędu. Można więc zapisać wektor modyfikacji wag w sposób:

$$\Delta W = -\nabla\varepsilon\left(\omega\right) \tag{7}$$

$$\begin{aligned} \Delta\omega_{l,n,i} &= -g_{l,n,i} \\ \Delta\beta_{l,n} &= -g_{l,n} \end{aligned} \tag{8}$$

Aby uczenie przebiegało nie za wolno i nie za szybko (co opiszę w rozdziale 6) można skorzystać z współczynnika uczenia α. Końcowo otrzymujemy wzór na zmianę wagi:

$$W = W + \alpha \cdot \Delta W \tag{9}$$

$$\begin{aligned} \omega_{l,n,i} &= \omega_{l,n,i} - \alpha \cdot \Delta\omega_{l,n,i} \\ \beta_{l,n} &= \beta_{l,n} - \alpha \cdot \Delta\beta_{l,n} \end{aligned} \tag{10}$$

4 Uproszczenie gradientu funckcji błędu standardowego

Kolejność wag w poniższym wzorze nie ma większego znaczenia.

$$\nabla\varepsilon\left(\omega\right)=\begin{bmatrix}\frac{\partial\varepsilon(\omega)}{\partial\omega_0}\\ \\ \frac{\partial\varepsilon(\omega)}{\partial\omega_1}\\ \\ \dots\\ \\ \frac{\partial\varepsilon(\omega)}{\partial\omega_{m-1}}\end{bmatrix} \tag{11}$$

W powyższym wzorze ω_i oznacza i-tą wagę względem całego zbioru (włącznie z biasami), a m oznacza ilość elementów owego zbioru.

4.1 Wyliczenie gradientów dla neuronów warstwy wyjściowej

4.1.1 Dla wag

Najpierw zajmiemy się wyliczeniem gradientu dla wag neuronów warstwy wyjściowej $\omega_{L-1,n,i}$. Ponieważ dla każdej wagi każdego z neuronów wyjściowych liczy się to dokładnie tak samo:

$$\begin{aligned}\frac{\partial\varepsilon\left(\omega\right)}{\partial\omega_{L-1,n,i}}&=\frac{\partial\left(\sum\limits_{j=0}^{N_{L-1}-1}\left(a_{L-1,j}-t_j\right)^2\right)}{\partial\omega_{L-1,n,i}}=\frac{\partial\left(a_{L-1,n}-t_n\right)^2}{\partial\omega_{L-1,n,i}}=\frac{\partial\left(a_{L-1,n}-t_n\right)^2}{\partial a_{L-1,n}}\cdot\frac{\partial a_{L-1,n}}{\partial\omega_{L-1,n,i}}=\\&=\frac{\partial\left(a_{L-1,n}-t_n\right)^2}{\partial a_{L-1,n}}\cdot\frac{\partial\sigma\left(S_{L-1,n}\right)}{\partial\omega_{L-1,n,i}}=\frac{\partial\left(a_{L-1,n}-t_n\right)^2}{\partial a_{L-1,n}}\cdot\frac{\partial\sigma\left(S_{L-1,n}\right)}{\partial S_{L-1,n}}\cdot\frac{\partial S_{L-1,n}}{\partial\omega_{L-1,n,i}}=\\&=\frac{\partial\left(a_{L-1,n}-t_n\right)^2}{\partial a_{L-1,n}}\cdot\frac{\partial\sigma\left(S_{L-1,n}\right)}{\partial S_{L-1,n}}\cdot\frac{\partial\left(\beta_{L,n}+\sum\limits_{j=0}^{N_{L-2}-1}a_{L-2,j}\cdot\omega_{L-1,n,j}\right)}{\partial\omega_{L-1,n,i}}=\\&=2\left(a_{L-1,n}-t_n\right)\cdot\sigma'\left(S_{L-1,n}\right)\cdot a_{L-2,i}=\\&=\frac{\partial\varepsilon\left(\omega\right)}{\partial\beta_{L-1,n}}\cdot a_{L-2,i}\end{aligned} \tag{12}$$

$$g_{L-1,n,i}=\frac{\partial\varepsilon\left(\omega\right)}{\partial\omega_{L-1,n,i}}=\frac{\partial\varepsilon\left(\omega\right)}{\partial\beta_{L-1,n}}\cdot a_{L-2,i}=g_{L-1,n}\cdot a_{L-2,i} \tag{13}$$

4.1.2 Dla biasów

Teraz należało by zająć się biasami neuronów warstwy wyjściowej. Analogicznie do wyprowadzenia 12 mamy że:

$$\begin{aligned}\frac{\partial\varepsilon\left(\omega\right)}{\partial\beta_{L-1,n}}&=\frac{\partial\left(a_{L-1,n}-t_n\right)^2}{\partial a_{L-1,n}}\cdot\frac{\partial\sigma\left(S_{L-1,n}\right)}{\partial S_{L-1,n}}\cdot\frac{\partial\left(\beta_{L,n}+\sum\limits_{j=0}^{N_{L-2}-1}a_{L-1,j}\cdot\omega_{L-1,n,j}\right)}{\partial\beta_{L-1,n}}=\\&=2\left(a_{L-1,n}-t_n\right)\cdot\sigma'\left(S_{L-1,n}\right)\end{aligned} \tag{14}$$

$$g_{L-1,n}=\frac{\partial\varepsilon\left(\omega\right)}{\partial\beta_{L-1,n}}=2\left(a_{L-1,n}-t_n\right)\cdot\sigma'\left(S_{L-1,n}\right)=2\left(a_{L-1,n}-t_n\right)\cdot a_{L-1,n}\cdot\left(1-a_{L-1,n}\right) \tag{15}$$

4.2 Wyliczenie gradientów dla neuronów warstw ukrytych

4.2.1 Dla warstwy L-2

Gradient wag neuronów warstw ukrytych jest bardziej złożony niż dla neuronów warstw wyjściowych. Dlatego na samym początku zajmiemy się wyprowadzeniem gradientu wag dla warstwy $L-2$:

$$
\begin{aligned}
\frac{\partial \varepsilon(\omega)}{\partial \omega_{L-2,n,i}} &= \frac{\partial\left(\sum\limits_{j=0}^{N_{L-1}-1}\left(a_{L-1,j}-t_j\right)^2\right)}{\partial \omega_{L-2,n,i}} = \sum_{j=0}^{N_{L-1}-1} \frac{\partial\left(a_{L-1,j}-t_j\right)^2}{\partial \omega_{L-2,n,i}} = \\
&= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\left(a_{L-1,j}-t_j\right)^2}{\partial a_{L-1,j}} \cdot \frac{\partial a_{L-1,j}}{\partial \omega_{L-2,n,i}}\right) = \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\left(a_{L-1,j}-t_j\right)^2}{\partial a_{L-1,j}} \cdot \frac{\partial \sigma\left(S_{L-1,j}\right)}{\partial \omega_{L-2,n,i}}\right) = \\
&= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\left(a_{L-1,j}-t_j\right)^2}{\partial a_{L-1,j}} \cdot \frac{\partial \sigma\left(S_{L-1,j}\right)}{\partial S_{L-1,j}} \cdot \frac{\partial S_{L-1,j}}{\partial \omega_{L-2,n,i}}\right) = \\
&= \sum_{j=0}^{N_{L-1}-1}\left(2\left(a_{L-1,j}-t_j\right) \cdot \sigma'\left(S_{L-1,j}\right) \cdot \frac{\partial\left(\beta_{L-1,j}+\sum\limits_{k=0}^{N_{L-2}-1} a_{L-2,k} \cdot \omega_{L-1,j,k}\right)}{\partial \omega_{L-2,n,i}}\right) = \\
&= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial \varepsilon(\omega)}{\partial \beta_{L-1,j}} \cdot \sum_{k=0}^{N_{L-2}-1} \frac{\partial\left(a_{L-2,k} \cdot \omega_{L-1,j,k}\right)}{\partial \omega_{L-2,n,i}}\right) = \\
&= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial \varepsilon(\omega)}{\partial \beta_{L-1,j}} \cdot \sum_{k=0}^{N_{L-2}-1} \frac{\partial a_{L-2,k}}{\partial \omega_{L-2,n,i}} \cdot \omega_{L-1,j,k}\right) = \\
&= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial \varepsilon(\omega)}{\partial \beta_{L-1,j}} \cdot \frac{\partial a_{L-2,n}}{\partial \omega_{L-2,n,i}} \cdot \omega_{L-1,j,n}\right) = \\
&= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial \varepsilon(\omega)}{\partial \beta_{L-1,j}} \cdot \frac{\partial \sigma\left(S_{L-2,n}\right)}{\partial S_{L-2,n}} \cdot \frac{\partial S_{L-2,n}}{\partial \omega_{L-2,n,i}} \cdot \omega_{L-1,j,n}\right) = \\
&= \sigma'\left(S_{L-2,n}\right) \cdot \frac{\partial S_{L-2,n}}{\partial \omega_{L-2,n,i}} \cdot \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial \varepsilon(\omega)}{\partial \beta_{L-1,j}} \cdot \omega_{L-1,j,n}\right) = \\
&= \sigma'\left(S_{L-2,n}\right) \cdot a_{L-3,i} \cdot \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial \varepsilon(\omega)}{\partial \beta_{L-1,j}} \cdot \omega_{L-1,j,n}\right) = \\
&= \frac{\partial \varepsilon(\omega)}{\partial \beta_{L-2,n}} \cdot a_{L-3,i}
\end{aligned}
\tag{16}
$$

Teraz wyliczymy biasy dla tych samych neuronów korzystając z powyższego 16 wzoru:

$$\begin{aligned}\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-2,n}} &= \sigma'(S_{L-2,n})\cdot\frac{\partial S_{L-2,n}}{\partial\beta_{L-2,n}}\cdot\sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-1,j}}\cdot\omega_{L-1,j,n}\right)=\\ &= \sigma'(S_{L-2,n})\cdot\frac{\partial\left(\beta_{L-1,j}+\sum_{k=0}^{N_{L-2}-1}a_{L-2,k}\cdot\omega_{L-1,j,k}\right)}{\partial\beta_{L-2,n}}\cdot\sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-1,j}}\cdot\omega_{L-1,j,n}\right)=\\ &= \sigma'(S_{L-2,n})\cdot\sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-1,j}}\cdot\omega_{L-1,j,n}\right)\end{aligned}\tag{17}$$

4.2.2 Dla warstwy L-3

Teraz wyprowadzimy gradient wag neuronów z warstwy $L-3$. Z powyższych wzorów (16)

$$\begin{aligned}\frac{\partial\varepsilon(\omega)}{\partial\omega_{L-3,n,i}} &= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-1,j}}\cdot\sum_{k=0}^{N_{L-2}-1}\left(\frac{\partial a_{L-2,k}}{\partial\omega_{L-3,n,i}}\cdot\omega_{L-1,j,k}\right)\right)\\ \frac{\partial a_{L-2,k}}{\partial\omega_{L-3,n,i}} &= \frac{\partial\sigma(S_{L-2,k})}{\partial\omega_{L-3,n,i}}=\sigma'(S_{L-2,k})\cdot\frac{\partial\left(\beta_{L-2,k}+\sum_{c=0}^{N_{L-3}-1}(a_{L-3,c}\cdot\omega_{L-2,k,c})\right)}{\partial\omega_{L-3,n,i}}=\\ &= \sigma'(S_{L-2,k})\cdot\frac{\partial(a_{L-3,n}\cdot\omega_{L-2,k,n})}{\partial\omega_{L-3,n,i}}=\sigma'(S_{L-2,k})\cdot\omega_{L-2,k,n}\cdot\sigma'(S_{L-3,n})\cdot\frac{\partial S_{L-3,n}}{\partial\omega_{L-3,n,i}}\\ \frac{\partial S_{L-3,n}}{\partial\omega_{L-3,n,i}} &= \ldots = a_{L-4,i}\\ \frac{\partial\varepsilon(\omega)}{\partial\omega_{L-3,n,i}} &= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-1,j}}\cdot\sum_{k=0}^{N_{L-2}-1}\left(\sigma'(S_{L-2,k})\cdot\omega_{L-2,k,n}\cdot\sigma'(S_{L-3,n})\cdot\frac{\partial S_{L-3,n}}{\partial\omega_{L-3,n,i}}\cdot\omega_{L-1,j,k}\right)\right)=\\ &= \sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-1,j}}\cdot\sum_{k=0}^{N_{L-2}-1}(\sigma'(S_{L-2,k})\cdot\omega_{L-2,k,n}\cdot\sigma'(S_{L-3,n})\cdot a_{L-4,i}\cdot\omega_{L-1,j,k})\right)=\\ &= \sigma'(S_{L-3,n})\cdot a_{L-4,i}\cdot\sum_{k=0}^{N_{L-2}-1}\left(\sigma'(S_{L-2,k})\cdot\omega_{L-2,k,n}\cdot\sum_{j=0}^{N_{L-1}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-1,j}}\omega_{L-1,j,k}\right)\right)=\\ &= \sigma'(S_{L-3,n})\cdot a_{L-4,i}\cdot\sum_{k=0}^{N_{L-2}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-2,k}}\cdot\omega_{L-2,k,n}\right)=\\ &= \sigma'(S_{L-3,n})\cdot a_{L-4,i}\cdot\sum_{j=0}^{N_{L-2}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-2,j}}\cdot\omega_{L-2,j,n}\right)\end{aligned}\tag{18}$$

Analogicznie do poprzednich wyprowadzeń biasów, z wzoru 18 mamy że:

$$\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-3,n}}=\sigma'(S_{L-3,n})\cdot\sum_{j=0}^{N_{L-2}-1}\left(\frac{\partial\varepsilon(\omega)}{\partial\beta_{L-2,j}}\cdot\omega_{L-2,j,n}\right)\tag{19}$$

4.2.3 Rekurencyjny wzór dla biasów neuronów dowolnych wag ukrytych

Z poprzednich wzorów można wywnioskować:

$$
\begin{aligned}
g_{l,n} = \frac{\partial \varepsilon\left(\omega\right)}{\partial \beta_{l,n}} = \sigma'\left(S_{l,n}\right) \cdot \sum_{j=0}^{N_{l+1}-1} \left(\frac{\partial \varepsilon\left(\omega\right)}{\partial \beta_{l+1,j}} \cdot \omega_{l+1,j,n} \right) = \\
= a_{l,n} \cdot \left(1 - a_{l,n}\right) \cdot \sum_{j=0}^{N_{l+1}-1} \left(\frac{\partial \varepsilon\left(\omega\right)}{\partial \beta_{l+1,j}} \cdot \omega_{l+1,j,n} \right)
\end{aligned} \tag{20}
$$

4.2.4 Rekurencyjny wzór dla wag neuronów dowolnych wag ukrytych

$$
g_{l,n,i} = \frac{\partial \varepsilon\left(\omega\right)}{\partial \omega_{l,n,i}} = \frac{\partial \varepsilon\left(\omega\right)}{\partial \beta_{l,n}} \cdot a_{l-1,i} = g_{l,n} \cdot a_{l-1,i} \tag{21}
$$

5 Efekt jo-jo

W trakcie uczenia sieci jest możliwość aby błąd standardowy rosną, może być to spowodowane przejściem przez 'górki' w funkcji błędu które otaczają minimum lokalne, za którą może się kryć niższe minimum lokalne od aktualnie znalezionego lub co gorsza wyżej położone minimum lokalne.

6 Over-learning i under-learning

7 Współczynnik uczenia

7.1 Zmiana współczynnika uczenia w czasie

8 Uczenie genetyczne

8.1 Sposób uczenia

8.1.1 Mutacje

8.1.2 Cross-over

8.1.3 Funkcja dopasowania

8.2 System kar i nagród

9 Sieć warstwowa nie mająca w pełni połączonych dwuch warstw

10 Przykładowa implementacja sieci neuronowej uczonej metodą wstecznej propagacji błędu

Link do repozytorium git z kodem

10.1 Obliczanie wyjścia sieci

10.2 Obliczanie gradientów

10.3 Modyfikacja wag