Dokumentacja zadania laboratoryjnego - badanie perceptronu dwuwarstwowego

Adam Szostek, nr indeksu 331443
Krzysztof Wyrzykowski, nr indeksu 331455

22 stycznia 2025

1.1 Opis działania algorytmu

Zadaniem perceptronu jest rozwiązanie problemu regresji, polegającego na przybliżeniu wartości funkcji $f(x)$. Perceptron składa się z neuronów połączonych węzłami, które posiadają swoje wagi. Neurony zgrupowane są w warstwy, tak aby każdy neuron danej warstwy był połączony z każdym neuronem warstwy następnej. Perceptron składa się z jednej warstwy wejściowej, jednej warstwy wyjściowej oraz co najmniej 1 warstwy wewnętrznej (ukrytej). Warstwa wejściowa zawiera tyle neuronów ile argumentów przyjmuje aproksymowana funkcja. Liczba neuronow w wartswie zewnętrznej opowiada liczbie wyjść funkcji. W warstwach ukrytych może znajdować się dowolna liczba neuronów, a od ich liczby i rozmieszczenia zależy działanie perceptronu. Wynik modelu określany jest wzorem:

$$\hat{f}(x) = w^\top \phi(x) + b$$

gdzie:

- w – wektor wag (parametry modelu),
- $\phi(x)$ – wektor cech wyjściowych z warstwy ukrytej (transformacja danych wejściowych),
- b – przesunięcie (bias).

Celem algorytmu jest minimalizacja błędu między rzeczywistymi wartościami funkcji $f(x)$ a wartościami przewidywanymi $\hat{f}(x)$, poprzez odpowiednią aktualizację wag w oraz przesunięcia b. Dla zadania regresji funkcją kosztu jest błąd średniokwadratowy (MSE), który wyraża się wzorem:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{f}(x_i) - y_i \right)^2$$

gdzie:

- N – liczba próbek w zbiorze uczącym,
- y_i – rzeczywista wartość funkcji dla próbki x_i,
- $\hat{f}(x_i)$ – wartość przewidziana przez model dla próbki x_i.

Celem perceptronu jest minimalizacja tej funkcji kosztu poprzez iteracyjne dopasowywanie wag w i przesunięć b.

Kluczowy aspektem działania perceptronu jest dodanie funkcji aktywacji na warstwach ukrytych, gdyż pozwala to na wprowadzenie nieliniowości. Wyjście a_{hidden} warstwy ukrytej z uwzględnieniem funkcji aktywacji jest obliczane według wzoru:

$$z_{\text{hidden}} = W_{\text{hidden}} \cdot x + b_{\text{hidden}}$$

$$a_{\text{hidden}} = \phi(z_{\text{hidden}})$$

gdzie:

- W_{hidden} – wagi warstwy ukrytej,
- b_{hidden} – przesunięcia warstwy ukrytej,
- $\phi(\cdot)$ – funkcja aktywacji (np. *ReLU*, *tanh*, *sigmoid*).

Wyjście sieci obliczane jest jako liniowa kombinacja aktywacji warstwy ukrytej:

$$\hat{f}(x) = W_{\text{output}} \cdot a_{\text{hidden}} + b_{\text{output}}$$

Proces uczenia perceptronu polega na minimalizacji funkcji kosztu za pomocą algorytmu propagacji wstecznej i gradientowego spadku. Aktualizacja wag odbywa się według reguły:

$$W \leftarrow W - \eta \cdot \frac{\partial \text{MSE}}{\partial W}$$

$$b \leftarrow b - \eta \cdot \frac{\partial \text{MSE}}{\partial b}$$

gdzie η to współczynnik uczenia (*learning rate*).

Obliczanie gradientów odbywa się w dwóch krokach:

1. **Propagacja wprzód (forward pass):** Obliczane są wartości wyjściowe dla warstw ukrytych i wyjściowej.
2. **Propagacja wsteczna (backward pass):** Wyznaczane są gradienty funkcji kosztu względem wag i przesunięć

Do trenowania perceptronu zastosowano zmodyfikowaną metodę gradientowego spadku nazywaną stochastic gradient descent, która oblicza gradienty na losowych próbkach ze zbioru treningowego o ustalonej wielkości. Podejście to znacznie poprawia czas zajmowany przez proces uczenia modelu, oraz w pewnych przypadkach poprawia również jakość uzyskanych wyników.

Dodatkowo do zaimplementowanego perceptronu dodano możliwość trenowania strategią ewolucyjną ES (1+1) z regułą 1/5 sukcesu. Aby poprawnie zaimplementować algorytm, wektory wag modelu są spłaszczane do jednej macierzy razem z biasami. W ten sposób strategia może prawidłowo eksplorować przestrzeń rozwiązań w poszukiwaniu optymalnych parametrów perceptronu.

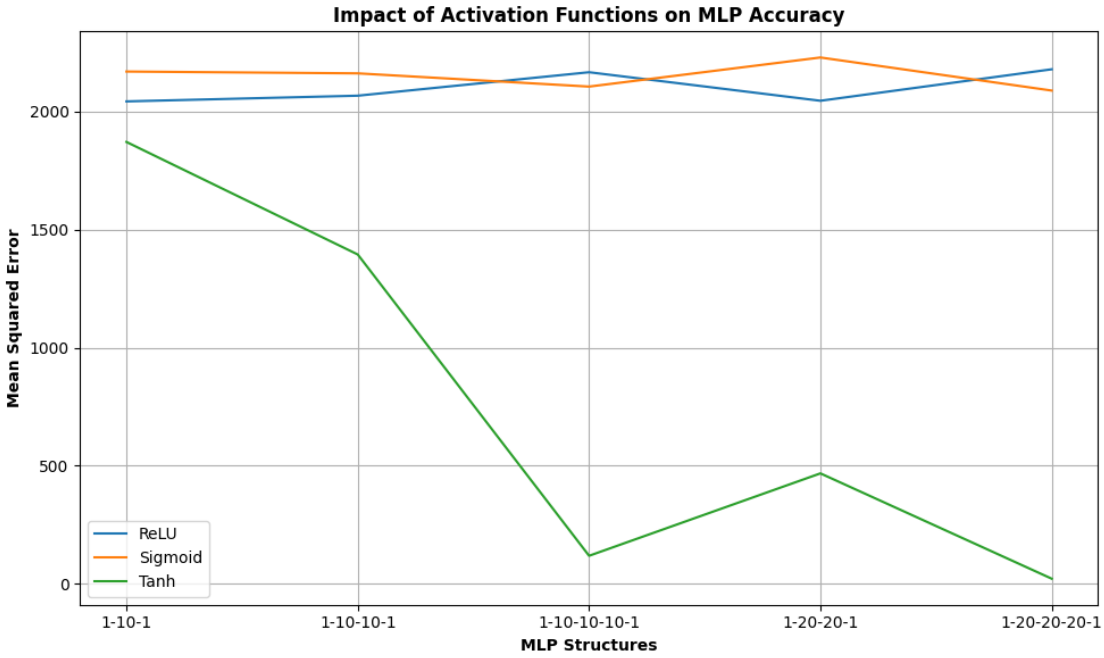
1.2 Planowane eksperymenty numeryczne

W celu przetestowania zaimplementowanego perceptronu, w problemie regresji, użyto danych pochodzących z funkcji danej wzorem:

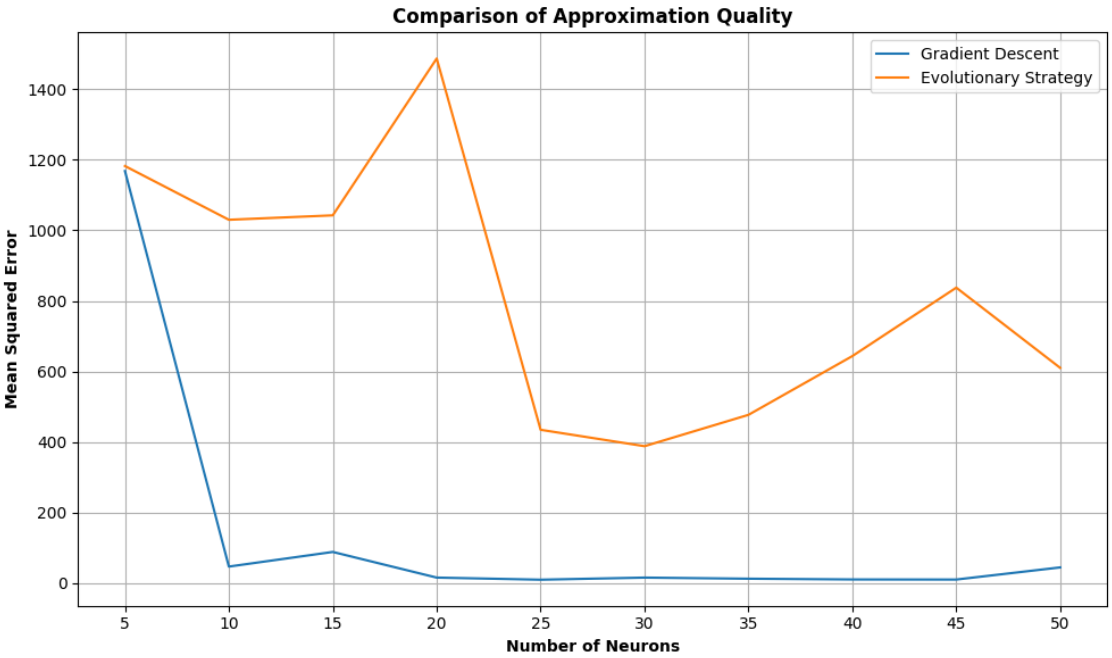
$$f\colon [-10, 10] \to R, \quad f(x) = x^2 \cdot \sin(x) + 100 \cdot \sin(x) \cdot \cos(x)$$

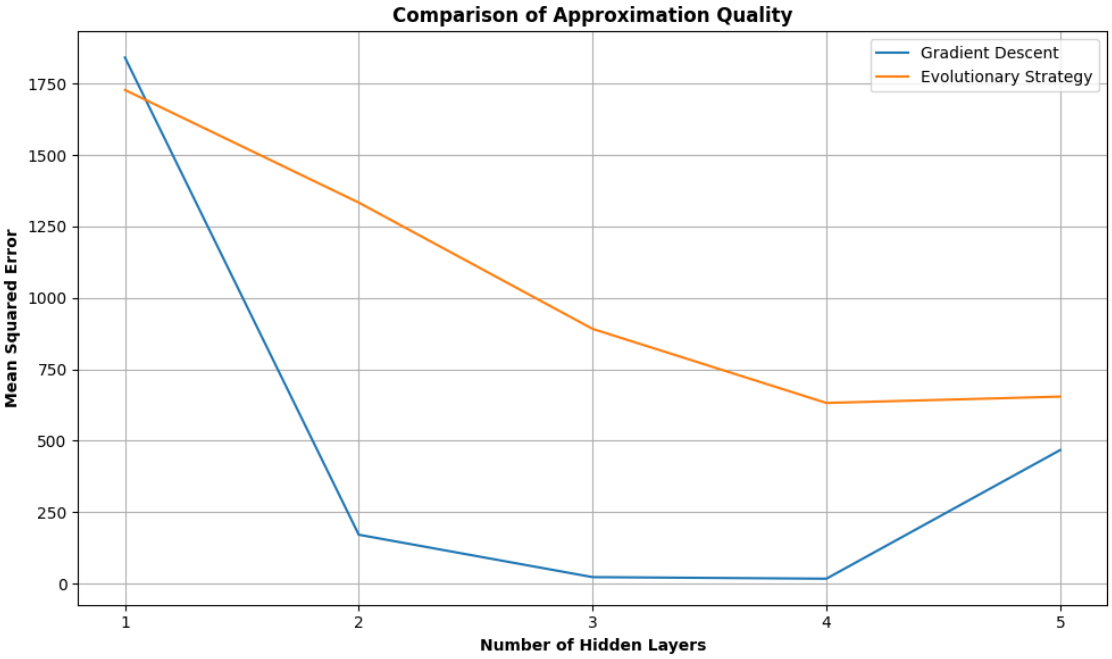
Zadaniem perceptronu jest przybliżenie funkcji na podstawie próbek danych. Podczas eksperymentów zbadano wpływ ilości warstw ukrytych oraz ilości neuronów w nich zawartych na jakość predykcji perceptronu. W celu analizy wyników sporządzono wykresy, na których zawarte są punkty testowe, predykcje perceptronu i krzywa reprezentująca aproksymowaną funkcję. Zbadano również wpływ dwóch zaimplementowanych metod treningu modelu (ES i SGD) na błąd na zbiorze testowym w zależności od różnych ilości neuronów w trzech warstwach ukrytch oraz samej liczby warstw ukrytych (20 neuronów każda), po czym sporządzono wykres z otrzymanych wyników. Dodatkowo w celu wybrania najlepszej funkcji aktywacji, sporządzono wykresy zależności średniego błędu od nieliniowości w zależności od różnych struktur perceptronu. Każdy wykres przedstawia uśrednione dane z pięciu prób trenowania.

1.3 Uzyskane wyniki

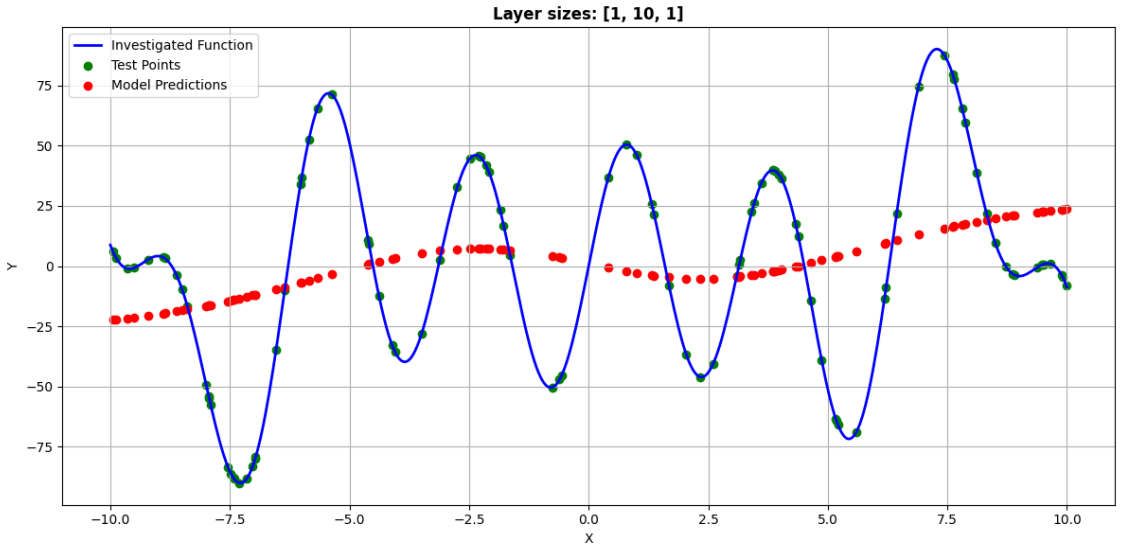


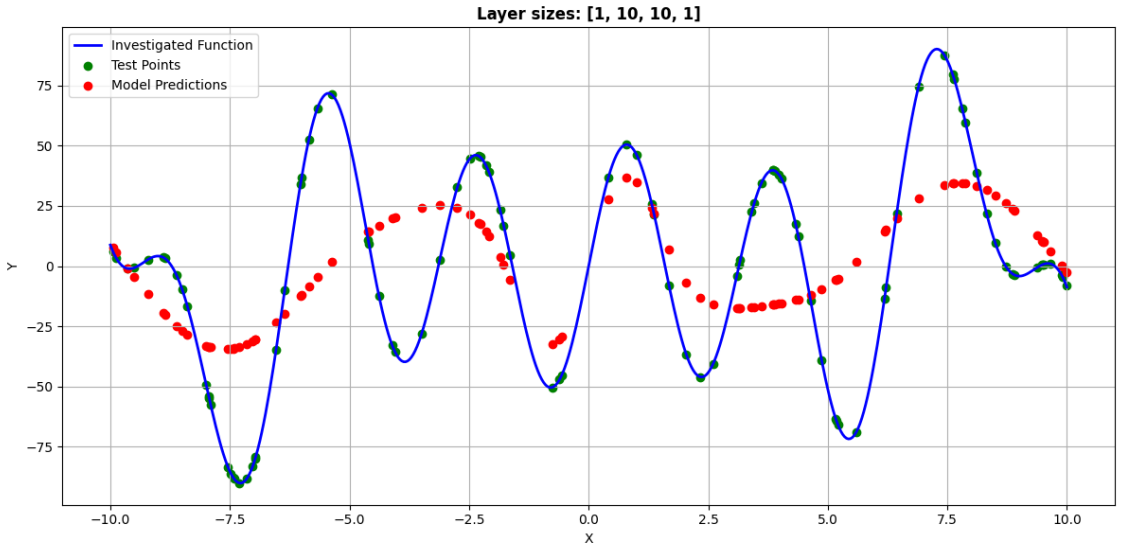
Na wykresie jasno widać, że zdecydowanie najlepsze wyniki osiąga tangens hiperboliczny, stąd w dalszych badaniach stosowano go jako funkcję aktywacji.

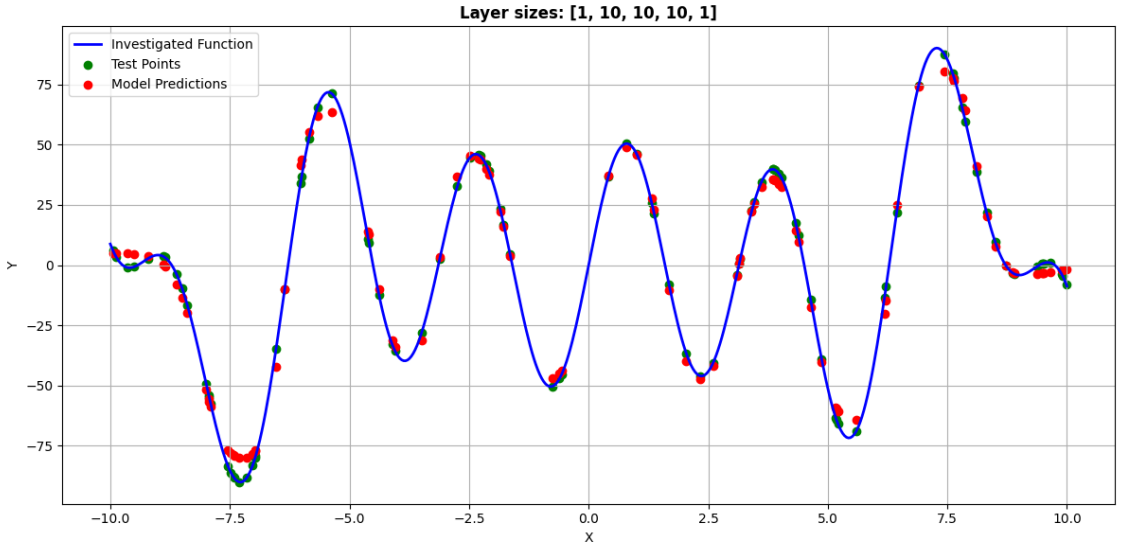

Comparison of Approximation Quality
Gradient Descent
Evolutionary Strategy
Mean Squared Error
Number of Neurons
0
200
400
600
800
1000
1200
1400
5
10
15
20
25
30
35
40
45
50

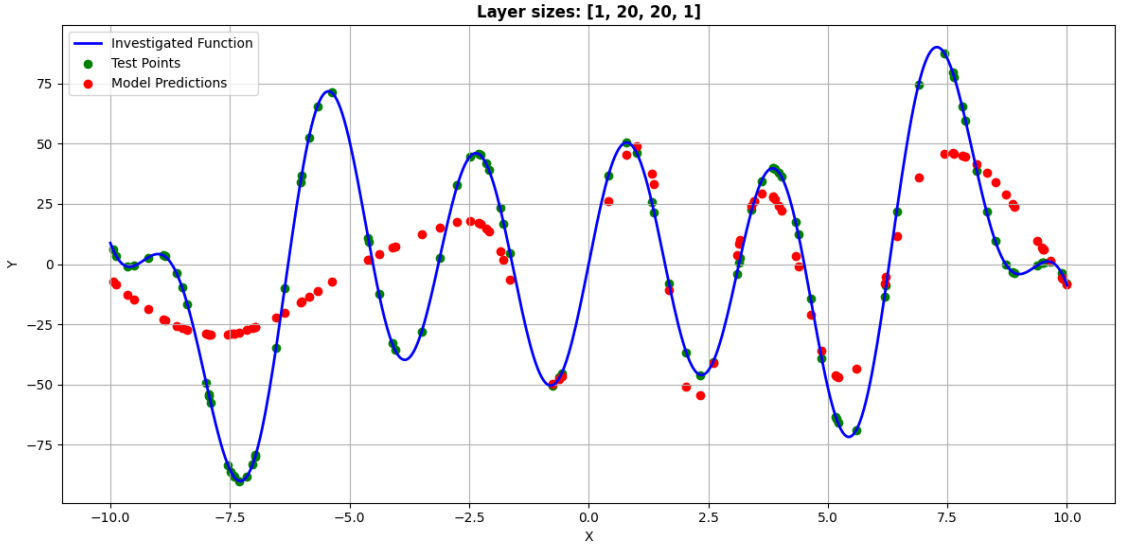

Comparison of Approximation Quality
Gradient Descent
Evolutionary Strategy
Mean Squared Error
Number of Hidden Layers
0
250
500
750
1000
1250
1500
1750
1
2
3
4
5

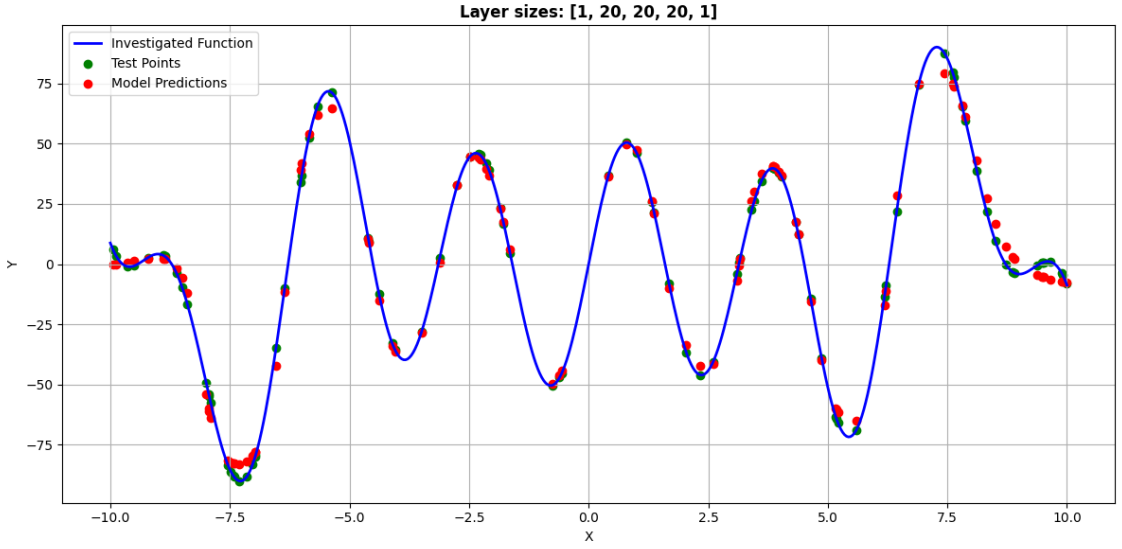
Na wykresie zależności błędu średnio-kwadratowego od liczby neuronów w warstwach ukrytych oraz od samych warstw widać, że metoda gradientowa zapewnia dużo stabilniejsze wyniki od strategii ewolucyjnej ES. Jednocześnie otrzymywany błąd również jest mniejszy dla SGD. Na podstawie wyników badań zdecydowano, że w tym konkretnym przypadku, bezpodstawne jest testowanie większej ilości neuronów w warstwach ukrytych niż 20 oraz więcej warstw niż 3, bo nie zmieni to w sposób znaczący na lepsze jakości uzyskanej aproksymacji. W dalszych badaniach model trenowano metodą gradientową (SGD).






Layer sizes: [1, 10, 10, 10, 1]
Investigated Function
Test Points
Model Predictions
Y
X


Layer sizes: [1, 20, 20, 1]
Investigated Function
Test Points
Model Predictions
Y
X


Layer sizes: [1, 20, 20, 20, 1]
Investigated Function
Test Points
Model Predictions
Y
X

Na wykresach prezentujących jakość predykcji w zależności od zastosowanej struktury warstw ukrytych można zaobserwować, że najdokładniejsze przybliżenie osiągamy dla trzech warstw ukrytych.

1.4 Wnioski z uzyskanych wyników

Na podstawie wykresów porównujących metody treningu modelu od liczby neuronów w warstwach ukrytych oraz od samych warstw można jasno stwierdzić, że metoda treningu gradientowego osiąga lepsze wyniki (mniejszy błąd) od strategii ES. Ponadto strategia ewolucyjna zachowuje się dużo mniej stabilnie od SGD. Dzieje się tak, ponieważ algorytm opiera się w dużej mierze na losowości, w przeciwieństwie do metody gradientowej, która w większości przypadków zapewni odnalezienie poszukiwanego minimum przy optymalizacji. Jednocześnie obserwując krzywą reprezentującą błąd uzyskany z modelu trenowanego przez SGD przy wykresie porównującym ilość neuronów można stwierdzić, że dla przyjętego problemu, liczba neuronów ma wpływ na jakość modelu do pewnego momentu, po czym nie da się stwierdzić obserowalnej zmiany uzyskiwanego błędu aproksymacji. Ten moment ma miejsce dokładnie przy 20 neuronach w warstwie ukrytej, dla każdej kolejnej wartości, jak wcześniej wspomniano, nie da się zaobserwować różnicy. Podobnie dzieje się z ilością warstw, dla której wybranie modelu z więcej niż trzema również nie zmniejsza błędu.

Analizując wykresy prezentujące przybliżenia modelu dla różnych kombinacji neuronów i warstw, można stwierdzić, że zarówno liczba warstw ukrytych, jak i liczba neuronów wpływają na jakość modelu. Jednak na podstawie wcześniejszych badań (wykres błędu względem liczby neuronów i warstw) wiadomo, że parametry te mają znaczenie tylko do pewnego momentu – po jego osiągnięciu dalsze ich zwiększanie nie poprawia wyników. Dla jednej warstwy ukrytej model nie jest w stanie dokładnie przybliżyć badanej funkcji, co widać po kształcie krzywej reprezentującej przewidywania perceptronu – jest ona tylko w niewielkim stopniu zbliżona do aproksymowanej funkcji. Po zwiększeniu liczby warstw do dwóch, obserwuje się znaczną poprawę jakości aproksymacji, choć wciąż pozostaje ona niedoskonała. Porównując wyniki modelu z trzema warstwami ukrytymi (po 10 neuronów każda) z modelem z dwiema warstwami (po 20 neuronów każda), wyraźnie widać, że model z trzema warstwami osiąga dokładniejsze przybliżenie. W tym przypadku sugeruje to, że liczba warstw wpływa bardziej na jakość modelu niż liczba neuronów. Najdokładniejszym modelem jest perceptron z trzema warstwami ukrytymi po 20 neuronów każda, który jest nieznacznie precyzyniejszy w przewidywaniach od opisywanego wcześniej.

Podsumowując, przeprowadzone badania pozwoliły ustalić, że metoda gradientowa (SGD) jest zdecydowanie bardziej stabilna i skuteczna niż algorytm ewolucyjny. Zbadano również wpływ parametrów określających strukturę modelu na jakość aproksymacji. Dla badanego problemu regresji ustalono, że najlepsze wyniki osiąga model z trzema warstwami po dwadzieścia neuronów każda. Powyższe wyniki jednoznacznie wskazują, że dobór parametrów perceptronu wielowarstwowego ma ogromny wpływ na jego wyniki, jednakże różne problemy wymagają innych podejść i innych wartości parametrów.