Politechnika Wrocławska

Podstawy sieci neuronowych

Projekt

13 stycznia 2025

Natalia Gazda 272590

Wstęp

Celem projektu była implementacja wielowarstwowej sieci neuronowej, która miała wykonywać następujące zadania:

1. Klasyfikacja obrazów (MNIST)
2. Aproksymacja funkcji Ackley'a

Należało zaimplementować sieć używając macierzy i wektorów, bez korzystania z gotowych bibliotek.
Projekt został zrealizowany w środowisku **Python**

Analiza problemu

Klasyfikacja obrazów

Klasyfikacja obrazów jest jednym z podstawowych problemów w dziedzinie uczenia maszynowego i sztucznej inteligencji. Polega na przypisaniu **etykiety** do obrazu na podstawie jego cech charakterystycznych. Dane wejściowe stanowią obrazy, które mogą być reprezentowane jako **macierze pikseli**. Obrazy te mogą mieć różne rozdzielczości, kolory (np. odcienie szarości, RGB).
Jednym z typowych przykładów klasyfikacji obrazów jest rozpoznawanie ręcznie pisanych cyfr. W takim przypadku system klasyfikacyjny ma za zadanie przypisać każdemu obrazowi cyfrę od 0 do 9. Dane mogą być zorganizowane w **zbiór treningowy i testowy**, gdzie obrazy treningowe służą do uczenia modelu, a obrazy testowe do oceny jego skuteczności. Każdy obraz jest reprezentowany jako macierz pikseli o stałej rozdzielczości i określonym formacie kolorów.

Aproksymacja funkcji

Funkcja Ackley'a jest narzędziem stosowanym do testowania algorytmów optymalizacyjnych i metod aproksymacji funkcji. Zadanie aproksymacji polega na znalezieniu funkcji, która na podstawie danych wejściowych będzie w stanie przewidywać wartości funkcji w punktach, które nie były wcześniej obserwowane.
Matematycznie, funkcja Ackley'a jest definiowana jako:

$$f(x) = -a \cdot \exp\left(-b \cdot \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(c \cdot x_i)\right) + a + \exp(1)$$

gdzie:

- $x = [x_1, x_2, \ldots, x_n]$ to wektor zmiennych wejściowych,
- a, b, c to stałe parametry (najczęściej $a = 20$, $b = 0.2$, $c = 2\pi$),
- n to liczba wymiarów.

Przygotowanie architektury oraz danych programu, analiza danych

W tej sekcji przedstawione zostały szczegóły dotyczące struktury programu, w tym wykorzystane biblioteki, format danych wejściowych oraz wyjściowych, opis procesu przetwarzania danych.

Biblioteki

Do realizacji zadania w języku Python, wykorzystano następujące biblioteki:

- NumPy - biblioteka używana do obliczeń numerycznych, wykorzystana do przechowywania i przetwarzania danych oraz do obliczeń związanych z funkcją Ackley'a
- Matplotlib - biblioteka używana do wizualizacji danych w postaci wykresów

Proces przetwarzania danych

1. Klasyfikacja (MNIST)
   (a) Wczytywanie danych z plików (np. train-images.idx3-ubyte)
   (b) Normalizacja (skalowanie wartości pikseli do przedziału [0,1])
   (c) Zmiana kształtu (z macierzy do wektora)
   (d) Kodowanie one-hot
   (e) Zastosowanie w sieci neuronowej
2. Aproksymacja funkcji Ackley'a
   (a) Generowanie danych
   (b) Obliczanie wartości funkcji Ackley'a
   (c) Normalizacja danych
   (d) Przygotowanie danych wejściowych (przekształcenie danych (x_1,x_2) do macierzy)
   (e) Zastosowanie w sieci neuronowej

Format danych

1. Klasyfikacja (MNIST)
   - ***Dane wejściowe*** to obrazy reprezentujące cyfry od 0 do 9 w rozdzielczości 28x28 pikseli, zapisane w odcieniach szarości. Każdy obraz został przekształcony w wektor o wymiarach 784x1 ($28 * 28 = 784$). Każda wartość wektora określa natężenie szarości danego piksela (z zakresu 0-255).
   - ***Dane wyjściowe*** to wyniki klasyfikacji, które zostały przedstawione za pomocą kodowania ona-hot. Oznacza to, że etykiety (cyfry od 0 do 9) zostają przekształcone na wektory o długości 10, w których tylko jeden element = 1 (reprezentuje on klasę), a pozostałe = 0

Przykład: dla etykiety 4, wektor one-hot wygląda następująco:
$[0, 0, 0, 0, 1, 0, 0, 0, 0, 0]$

2. Aproksymacja funkcji Ackley'a

   - ***Dane wejściowe*** to losowo wybrane punkty (x_1,x_2) w przedziale $[-2, 2]$, w którym dla każdego punktu obliczana będzie wartość funkcji Ackley'a $f_A(x_1,x_2)$. Dane treningowe (1000 punktów) są tworzone jako macierz o wymiarach 1000x2, gdzie każda para (x_1,x_2) staje się jednym wierszem macierzy. Analogiczne stworzono macierz dla danych testowych. Przed wprowadzeniem danych do sieci wartości (x_1,x_2) są normalizowane - dane zostają przesunięte z zakresu $[-2, 2]$ do zakresu $[0, 1]$ dla każdego punktu.
   - ***Dane wyjściowe*** to przewidywane wartości funkcji dla nowych punktów. Na wejściu modelu podawane są współrzędne (x_1,x_2), na wyjściu zwracana zostaje wartość funkcji $f_A(x_1,x_2)$, która jest aproksymowaną wartością funkcji w tym punkcie.

     Przykład: dla punktu ($x_1 = 0$,$x_2 = 1$), na wyjściu modelu powinna pojawić się przybliżona wartość $f_A(0,1)$

Podział danych

1. Klasyfikacja (MNIST)

   - ***Zbiór treningowy:*** 60 000 obrazów (używane do trenowania modelu)
   - ***Zbiór testowy:*** 10 000 obrazów (używane do oceny wydajności modelu)

2. Aproksymacja funkcji Ackley'a

   - ***Zbiór treningowy:*** 1000 losowych punktów w przestrzeni [-2,2] (służące do nauki funkcji Ackley)
   - ***Zbiór testowy:*** Siatka punktów rozmieszczonych co 0.01 w przedziale [-2,2] (służąca do testowania jakości aproksymacji funkcji)

Przygotowanie danych

Wybór architektury sieci

Rodzaj sieci

W projekcie została wykorzystana **sieć jednokierunkowa** (feedforward neural network), która charakteryzuje się przepływem informacji w jednym kierunku - od warstwy wejściowej, przez warstwę ukrytą, do warstwy wyjściowej.
Sieć zawiera **jedną warstwę ukrytą** i została osobno dostosowana do problemu klasyfikacji i aproksymacji.

Funkcje aktywacji (decydują czy dany neuron powinien zostać aktywowany) wykorzystane w projekcie:

- Sigmoid - nieliniowa funkcja aktywacji (funkcja logistyczna), która przekształca dowolną liczbę wejściową x na wartość, która mieści się w przedziale (0,1)

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

- Liniowa - liniowa funkcja aktywacji, wynik zawsze jest proporcjonalny do wejścia, bez jakichkolwiek modyfikacji

$$f(x) = x$$

Dodatkowo w zadaniu wykorzystano **funkcję błędu MSE** (błąd średniokwadratowy), która została wykorzystana do oceny jakości modelu.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

(y_i - rzeczywista wartość; $\hat{y}_i$ - przewidywana wartość; n - liczba próbek)

Parametry sieci neuronowej

Sieć neuronowa została poddana badaniom w celu sprawdzenia wpływu zmiany kluczowych paramterów na jej działanie. Zmieniając **liczbę neuronów**, **liczbę epok** oraz **współczynnik uczenia** starano się znaleźć optymalne ustawienia, które zapewnią najlepsze wyniki w zadaniu klasyfikacji, jak i aproksymacji. Poniżej zostały przedstawione wartości tych parametrów.

- Liczba neuronów: [10, 50, 100, 200]
- Liczba epok: [10, 5000, 10000]
- Współczynnik uczenia: [0.005, 0.05, 0.1]

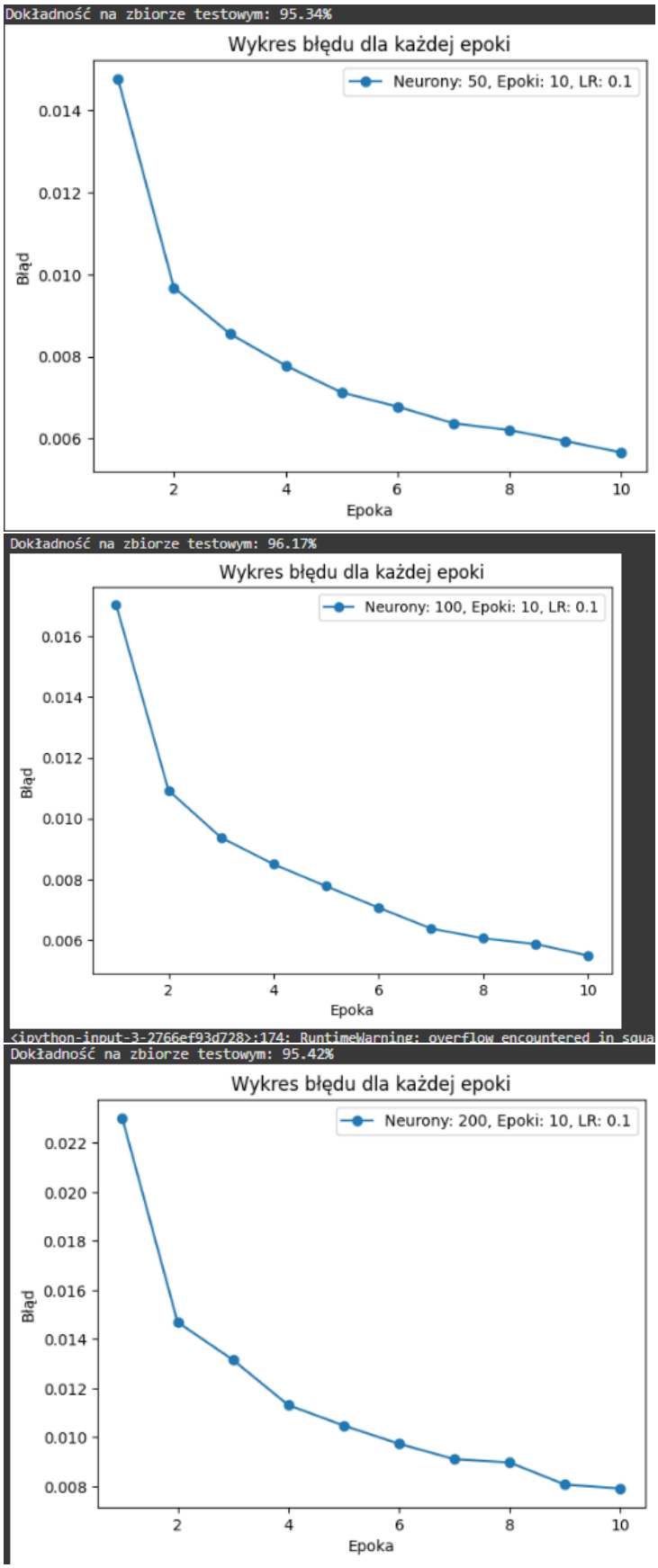
Trening i badania

Poniżej zostały przedstawione przykładowe wykresy z przeprowadzonych badań. Wszystkie wyniki zostały zebrane w tabeli.

1. Klasyfikacja

KLASYFIKACJA									
Liczba neuronów	50	50	50	100	100	100	200	200	200
Liczba Epok	10	10	10	10	10	10	10	10	10
Współczynnik uczenia	0.005	0.05	0.1	0.005	0.05	0.1	0.005	0.05	0.1
Dokładność na zbiorze testowym [%]	95.91	96.08	95.34	96.71	97.16	96.17	97.10	97.31	95.42

Rysunek 1: Wyniki badań dla klasyfikacji

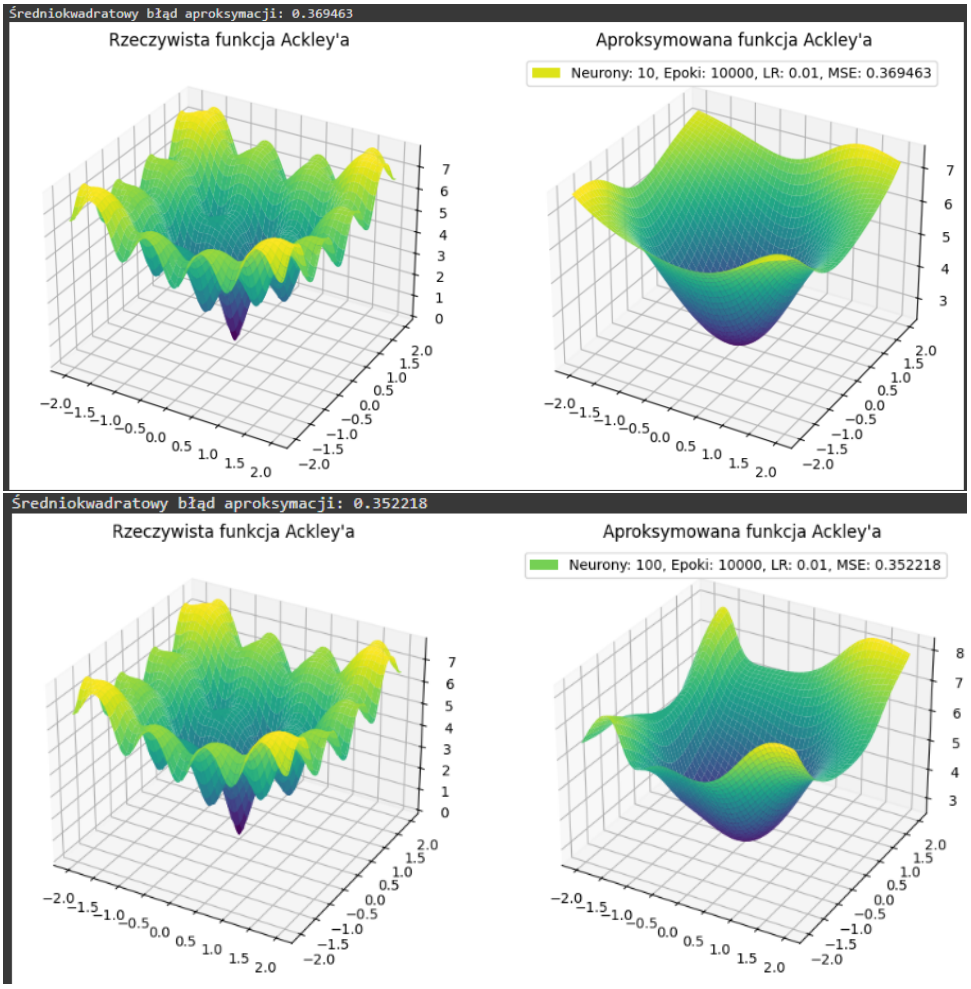


Rysunek 2: Badania dla zmian ilości neuronów [50,100,200] przy stałym współczynniku uczenia = 0.1

2. Aproksymacja

APROKSYMACJA						
Liczba neuronów	10	100	10	100	10	100
Liczba Epok	10000	10000	5000	5000	5000	5000
Współczynnik uczenia	0.01	0.01	0.01	0.01	0.005	0.005
MSE	0.369	0.352	0.331	0.276	0.358	0.351

Rysunek 3: Wyniki badań dla aproksymacji



Rysunek 4: Badania dla zmian ilości neuronów [10,100] przy stałym współczynniku uczenia = 0.01 oraz dla 10000 epok

Analiza wyników

1. Klasyfikacja
   Analizując wyniki klasyfikacji obrazów z użyciem sieci neuronowej, zauważono znaczącą poprawę dokładności po zwiększeniu liczby neuronów w warstwie ukrytej. Najlepsze wyniki osiągnięto dla konfiguracji z 200 neuronami i współczynnikiem uczenia 0.05, gdzie dokładność klasyfikacji wynosiła **97.31%**. Natomiast przy mniejszej liczbie neuronów, np. 50, dokładność była niższa, wynosząca około **95%**.

2. Aproksymacja
   W przypadku aproksymacji funkcji Ackley'a, wyniki były bardziej złożone ze względu na nieliniowość funkcji oraz wymagającą konfigurację sieci. Zmiany w liczbie neuronów i epokach miały znaczący wpływ na dokładność aproksymacji. Najlepsze wyniki uzyskano dla 100 neuronów i wysokiej liczby epok 5000, co doprowadziło do dokładniejszych przybliżeń funkcji Ackley'a w porównaniu do innych konfiguracji.

Podsumowanie

Podsumowując eksperymenty z siecią neuronową, można stwierdzić, że wybór odpowiedniej konfiguracji sieci ma kluczowe znaczenie dla uzyskania satysfakcjonujących wyników. Klasyfikacja obrazów wymagała większej liczby neuronów, aby osiągnąć wysoką dokładność, podczas gdy aproksymacja funkcji Ackley'a potrzebowała głębszej architektury z większą liczbą epok.
Sukcesy w obu przypadkach były wynikiem skrupulatnych badań i dostosowania parametrów, chociaż napotkano wyzwania związane z długim czasem uczenia się w przypadku skomplikowanych funkcji.