Wstęp do sztucznej inteligencji
Sztuczne sieci neuronowe

Jakub Robaczewski, Michał Matak

Wstęp:
Celem ćwiczenia było stworzenie sieci neuronowej, która będzie kwalifikować obrazy ze zbioru MNIST jako cyfry.

Architektura:
Zostały stworzone dwie klasy Layer oraz Net. Klasa Layer posiada atrybut będący wartościami neuronów, funkcje aktywacji i jej gradient (przyjmujące i zwracające wektor), a także macierz wag dla połączenia z następną warstwą (wymiar *wejście x wyjście*) i bias-y dla następnej warstwy. Posiada metody, które:

- liczą wartość neuronów dla następnej warstwy (propagacja - realizowana jako __call__)
- liczą sumę x θ + b (gdzie θ to macierz wag)
- łączą warstwy ze sobą
- wyliczają pewną pomocniczą wartość przy wyliczaniu gradientu i modyfikują według niej wagi (propagacja wsteczna)

Klasa Net grupuje obiekty klasy Layer i wykonuje na nich operacje propagacji i propagacji wstecznej (według odpowiedniej kolejności).

Liczenie gradientu:
Przyjętą przez nas funkcją celu jest suma kwadratów odchyleń. W związku z tym gradient dla ostatniej warstwy jest wyliczany według wzoru:

$$\delta_1^{(L)} = 2\left(h^{(L)} - y\right) \odot g'\left(\left(\theta^{(L-1)}\right)^T h^{(L-1)} + b^{(L-1)}\right)$$

Gdzie δ to pomocnicza wartość, o której wspomniano wcześniej (dla *L-tej* warstwy).
A dla kolejnych warstw *l = L-1, L-2, …, 2, 1:*

$$\delta^{(l)} = \left(\left(\theta^{(l)}\right)^T \delta^{(l+1)}\right) \odot g'\left(\left(\theta^{(L-1)}\right)^T h^{(l-1)} + b^{(l-1)}\right)$$

Wagi i gradient są dekrementowane według tych wartości:

$$\Delta\theta^{(l)} = \delta^{(l+1)}(h^{(l)})^T$$
$$\Delta b^{(l)} = \delta^{(l+1)}$$

Odczytywanie plików:
Dane trenujące i testujące dla sieci neuronowej są odczytywane odpowiednio z plików zawierających zbiór MNIST:

- data/t10k-images.idx3-ubyte – obrazy do testowania
- data/t10k-labels.idx1-ubyte – etykiety do testowania
- data/train-images.idx3-ubyte – obrazy do trenowania
- data/train-labels.idx1-ubyte – etykiety do trenowania

Pliki są odczytane binarnie, najpierw odczytywany jest deskryptor zawierający kod walidacyjny, ilość obrazów oraz wysokość i szerokość obrazów, natomiast w przypadku etykiet tylko kod walidacyjny i ilość etykiet. Znając kody walidacyjne jesteśmy w stanie sprawdzić, czy plik został prawidłowo odczytany.

Wyniki:

Zbiór uczący: 60 000 elementów
Zbiór testujący: 10 000 elementów
Sieć neuronowa: 784, 300, 10
β=0.2
Funkcja aktywacji: arcus tangens

Próba	Próba 1	Próba 2	Próba 3	Próba 4	Próba 5	Próba 6
Poprawność	42.48%	37.85%	26.27%	37.95%	33.45%	35.35%

Zbiór uczący: 60 000 elementów
Zbiór testujący: 10 000 elementów
Sieć neuronowa: 784, 300, 10
β=0.2
Funkcja aktywacji: sigmoida

Próba	Próba 1	Próba 2	Próba 3	Próba 4	Próba 5	Próba 6
Poprawność	96.07%	95.91%	95.43%	96.02%	96.01%	95.35%

Zbiór uczący: 60 000 elementów
Zbiór testujący: 10 000 elementów
Sieć neuronowa: 784, 30, 10
β=0.2
Funkcja aktywacji: sigmoida

Próba	Próba 1	Próba 2	Próba 3	Próba 4	Próba 5	Próba 6
Poprawność	93.64%	94.08%	94.42%	94.24%	94.11%	93.95%

Zbiór uczący: 60 000 elementów
Zbiór testujący: 10 000 elementów
Sieć neuronowa: 784, 10
β=0.2
Funkcja aktywacji: sigmoida

Próba	Próba 1	Próba 2	Próba 3	Próba 4	Próba 5	Próba 6
Poprawność	88.49%	89.94%	89.60%	90.35%	90.51%	90.64%

Dane były dobierane metodą prób i błędów. Zauważyliśmy, że arcus tangens jako funkcja aktywacji daje dużo słabsze wyniki niż sigmoida. Natomiast strukturą sieci dającą najlepsze rezulaty okazała się sieć 784, 300, 10.