WSI - ćwiczenie 5

Agnieszka Wójtowicz, Jakub Kowalczuk

12 maja 2022

Spis treści

1 Polecenie oraz elementy języka

1.1 Polecenie Prowadzącego

W ramach piątych ćwiczeń będą Państwo musieli zaproponować architekturę, zaimplementować, wytrenować i przeprowadzić walidację sieci neuronowej do klasyfikacji ręcznie pisanych cyfr.

1.2 Elementy języka

Wykorzystano język programowania Python w wersji 3.8. Skorzystano z pomocniczych bibliotek: *numpy 1.20.1*, *pandas 1.3.4*, *matplotlib*, *sklearn*.

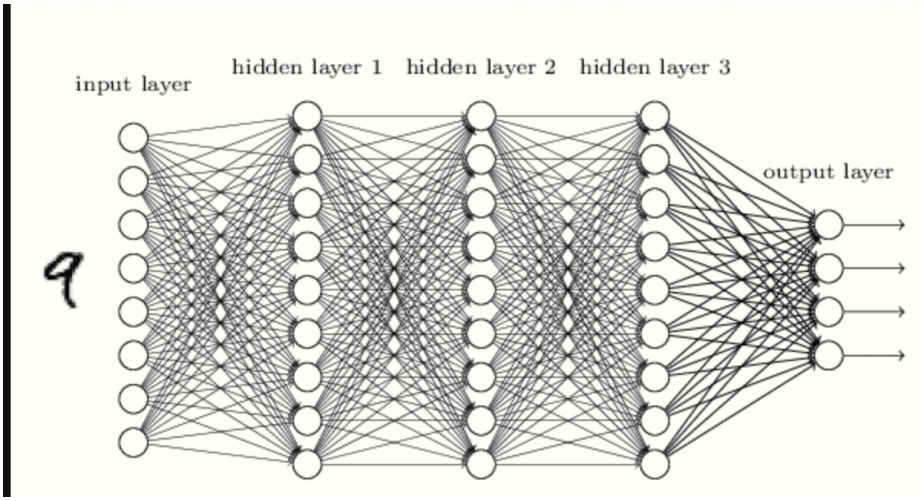
2 Cel zadania

Celem zadania jest implementacja sieci neuronowej oraz wytrenowanie jej na zbiorze danych MNIST.

3 Wstęp

3.1 Opis architektury

Zaimplementowano sieć neuronową i wykorzystano ją do klasyfikacji danych. W przypadku warstw ukrytych zaimplementowano warstwy w pełni połącznone (dense) z funkcją aktywacji sigmoidalną. Warstwę wyjściową zaimplementowano z funkcją aktywacji softmax, by uzyskać na wyjściu uzyskać prawdopodobieństwa przynależności do klas. Jako końcową predykcje wybierano tą, która miała maksymalną wartość prawdopodobieństwa. Poniższy rysunek przedstawia przykładową architekture sieci (źródło: https://chsasank.com/deep-learning-crash-course-2.html).



Rysunek 1: Przykładowa architektura sieci.

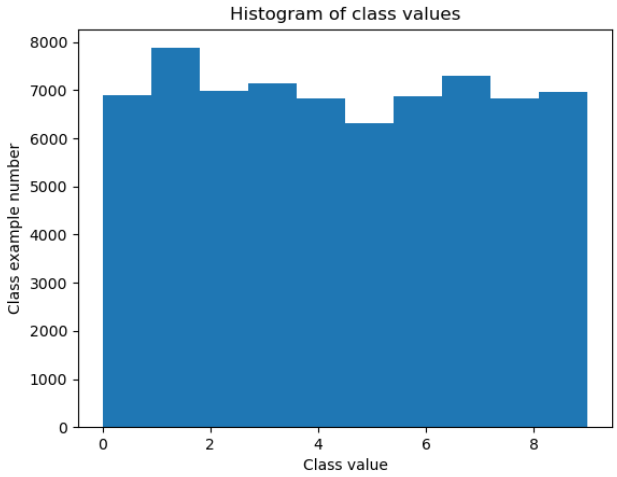
W wypadku sieci wykorzystywanej w poniższym zadniu najczęściej, składa się ona z:

- warstwa wejściowa danych (784x1),
- warstwa ukryta (784x20) + sigmoidalna f.aktywacji,
- warstwa wyjściowa (20x10) + f. aktywacji softmax.

3.2 Dane

Dane treningowe składają się z opisu rysunków przedstawiających cyfry. Każdy przykład składa się z 784 pikseli. Dane zawierając opis natężenia koloru dla każdego z pikseli.

Sprawdzono rozkład klas w danych. Poniższy rysunek przestawia histogram klas. Częstości występowania klas sa zbliżone - zbiór danych jest zbalansowany.



Rysunek 2: Histogram rozkładu klas.

3.3 Badania

Dla sieci dwuwarstwowej zbadano, jak zmiana ilości neuronów w warstwie ukrytej zmienia wyniki modelu. Sprawdzono również, jak zmiana kroku uczenia wpływa na proces nauki.

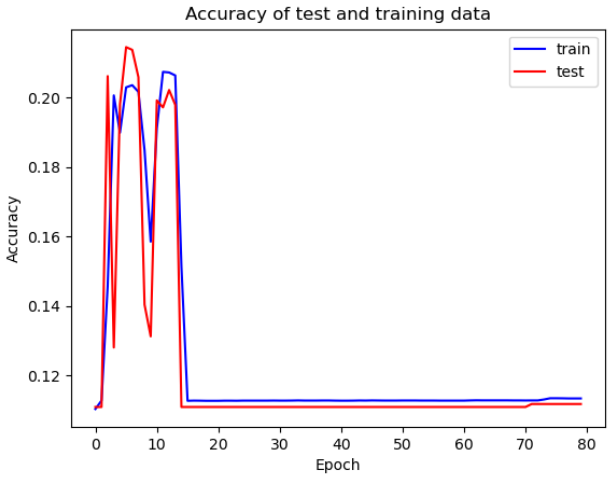
Dla ilości neuronów w warstwie ukrytej oraz kroku uczenia, które dawały zadowalające wyniki, sprawdzono jak zmiana ilości warstw ukrytych wpływa na wyniki modelu.

Wszystkie badania przeprowadzano na zbiorze treningowym będącym 0.8 całego zbioru. Podczas nauki zbierano również dokładność sieci na pozostałym fragmencie zbioru. Parametry opisujące architekturę i proces nauki sieci, o ile nie były przedmiotem badania, to: krok uczący = 0.01, liczba warstw ukrytych = 1, ilość neuronów warstwy ukrytej = 20.

4 Wyniki

4.1 Liczba neuronów w warstwie ukrytej

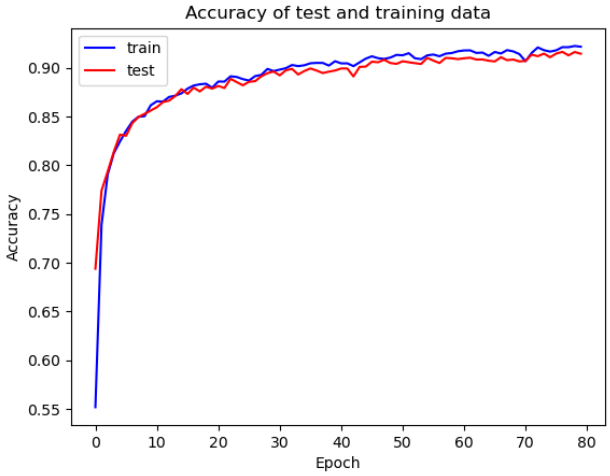
Rysunki przedstawiają dokładność sieci na zbiorze treningowym i testowym, dla różnej ilości neuronów w warstwie ukrytej.



Rysunek 3: Wyniki sieci dla 1 neuronu ukrytego.

Rysunek 4: Wyniki sieci dla 5 neuronów ukrytych.

Rysunek 5: Wyniki sieci dla 10 neuronów ukrytych.

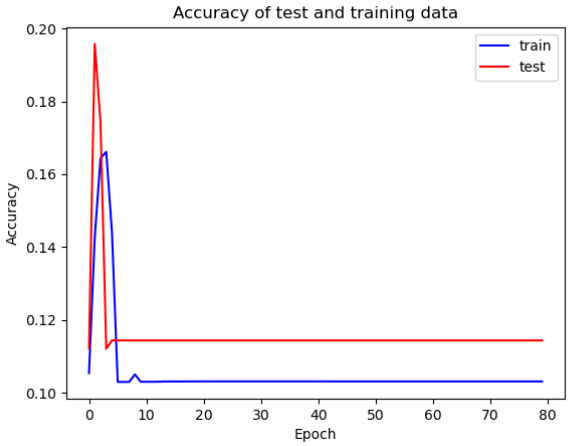


Rysunek 6: Wyniki sieci dla 30 neuronów ukrytych.

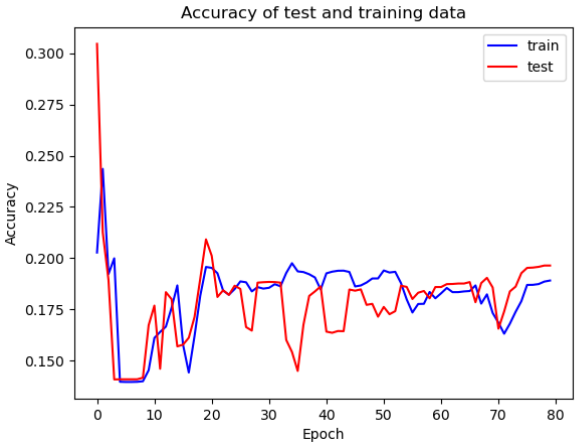
Rysunek 7: Wyniki sieci dla 100 neuronów ukrytych.

4.2 Wielkość kroku uczącego

Rysunki przedstawiają dokładność sieci na zbiorze treningowym i testowym, dla różnej wielkości kroku uczącego.



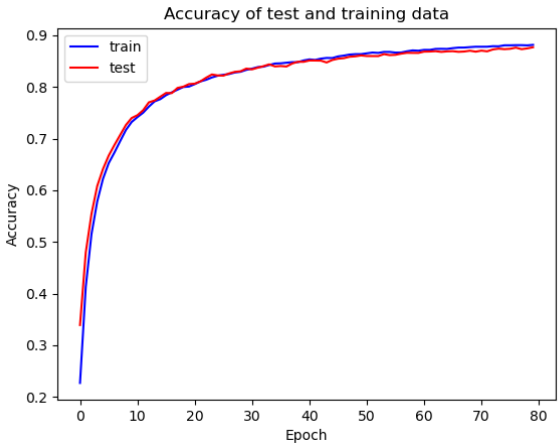
Rysunek 8: Wyniki modelu dla kroku uczącego = 1.



Rysunek 9: Wyniki modelu dla kroku uczącego = 0.5.

Rysunek 10: Wyniki modelu dla kroku uczącego = 0.1.

Rysunek 11: Wyniki modelu dla kroku uczącego = 0.01.



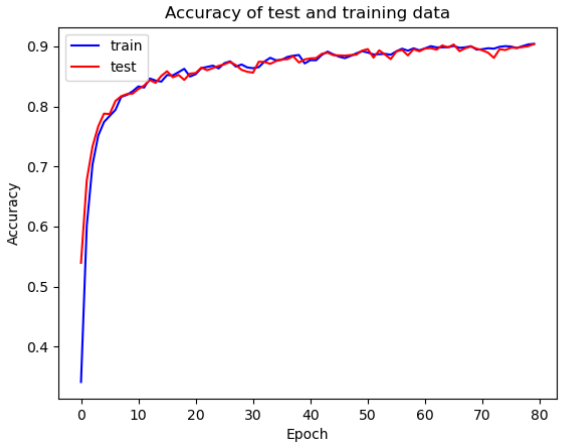
Rysunek 12: Wyniki modelu dla kroku uczącego = 0.001.

4.3 Liczba warstw ukrytych

Rysunki przedstawiają dokładność sieci na zbiorze treningowym i testowym, dla różnej ilości warstw ukrytych.

Rysunek 13: Wyniki modelu liczby warstw ukrytych = 1

Rysunek 14: Wyniki modelu liczby warstw ukrytych = 2

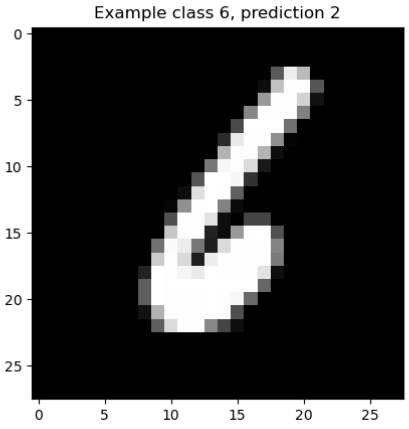


Rysunek 15: Wyniki modelu liczby warstw ukrytych = 4

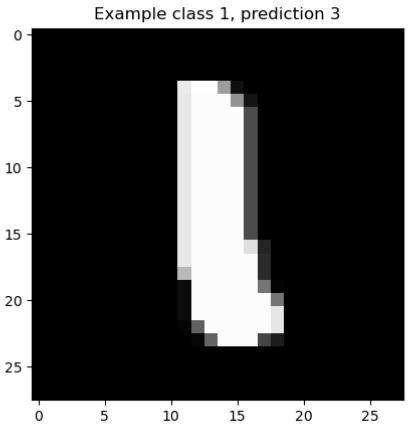
Rysunek 16: Wyniki modelu liczby warstw ukrytych = 6

4.4 Przykładowe predykcje

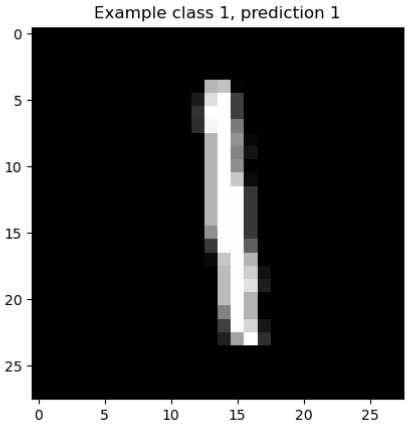
Dla wybranej, osiągającej zadowalające efekty architektury (dokładność na zbiorze treningowym 0.91), znaleziono przykłady dla których dokonuje ona predykcji prawidłowej i nieprawidłowej. Rysunki znajdują się poniżej.



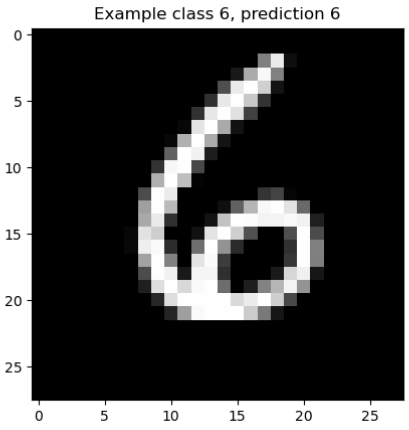
Rysunek 17: Przykład nieprawidłowej predykcji



Rysunek 18: Przykład nieprawidłowej predykcji



Rysunek 19: Przykład prawidłowej predykcji



Rysunek 20: Przykład prawidłowej predykcji

5 Wnioski

5.1 Liczba neuronów w warstwie ukrytej

Dla jednego neuronu, sieć praktycznie się nie uczy. Po kilkunastu epokach jej wyniki wskazują na losowe działanie sieci. Dla większej liczby neuronów, sieć osiąga coraz lepsze wyniki, które stają się zadowalające już dla 10 neuronów.

5.2 Wielkość kroku uczącego

Dla zbyt dużego (większego lub równego 0.1) kroku, sieć osiąga słabe wyniki. Krzywe dokładności nie są gładkie i wskazują na niemal losowe działanie sieci. Zadowalające wyniki można zaobserwować dla kroku uczącego mniejszego lub równego 0.01.

5.3 Liczba warstw ukrytych

Wraz z wzrostem liczby warstw ukrytych, poprawiają się wyniki sieci. Krzywa dokładności ma dla nich gładszy kształt. Poprawa nie jest jednak tak duża, jak by się można tego spodziewać. Wynika to prawdopodobnie z tego, że warstwy ukryte nie są różnorodne.

5.4 Przykładowe predykcje

Na podstawie jedynie kilku przykładowych predykcji trudno wysnuć daleko idące wnioski. Widzimy dosyć oczywisty fakt, tj. bardziej charakterystyczne obrazki - zbliżone do "prawidłowego" kształtu danej liczy - uzyskują prawidłowe predykcję, natomiast w przypadku gdy analizowany obrazek przedstawia niedbale zapisaną liczbę model sieci nie jest w stanie poprawnie odnaleźć cech dających poprawną predykcję.

5.5 Podział obowiązków

Kod oraz raport pisany był wspólnie, na zasadzie peer-programming, ciężko jest więc podać dokładny podział obowiązków. Agnieszka skupiła się na na fazie testowania modelu, Jakub więcej uwagi poświęcił implementacji modelu.