Andrzej Pultyn

325213

WSI – zadanie 4

- Wstęp

Zadaniem było napisanie wielowarstwowej sieci neuronowej, która rozwiąże problem klasyfikacji cyfr ze zbioru MNIST, czyli rozpoznawanie pisanych cyfr.

- Rozwiązanie

Moja sieć neuronowa składa się z 4 warstw: wejściowego obrazka (która składa się z 784 neuronów odpowiadającym 784 pikselom), dwom warstwom ukrytym, oraz wyjściowej, składającej się z 10 neuronów, każdy odpowiada jednej cyfrze. Każdy neuron ma połączenie z każdym neuronem z kolejnej warstwy. Każde takie połączenie ma swoją wagę. Celem sieci neuronowej jest takie dobranie wszystkich wag, żeby po otrzymaniu obrazka na wejściu, na końcu najmocniej zapalił się neuron oznaczający właściwą cyfrę. Na początku wszystkie wagi mają losowe wartości.

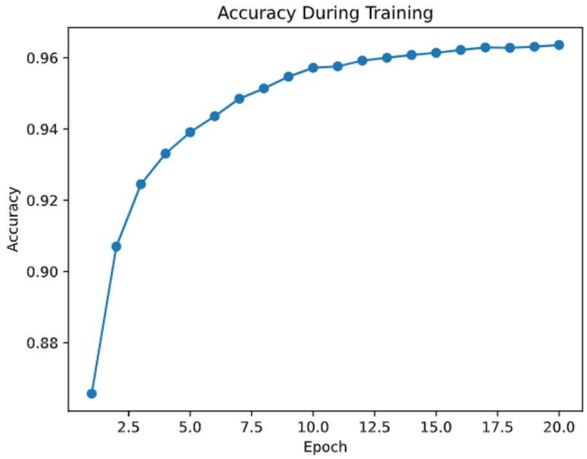
Sieć uczy się poprzez przepuszczeniu przez nią wszystkich 60000 obrazków. Każdy taki zestaw treningowy można nazwać epoką. Po podaniu obrazka, kolejne warstwy zapalają się poprzez sumę iloczynów wartości wszystkich poprzednich neuronów i wagi na tym połączeniu. Później należy „spłaszczyć" wynik, aby mieścił się w przedziale od 0 do 1. Można użyć do tego np. funkcji *sigmoid*. W ten sam sposób liczymy wartości neuronów, aż do ostatniej warstwy. Tam nasz wynik zostaje przekształcony funkcją *softmax*, która zamienia wartości neuronów w prawdopodobieństwa poszczególnych wartości.

Następnie tworzymy sobie funkcję kosztu, składającej się z różnicy wyniku osiągniętego i docelowego. Taką funkcję chcemy jak najbardziej zminimalizować, aby uzyskać jak najlepszy wynik. Do minimalizacji funkcji używamy gradientu prostego, czyli wykonujemy krok w przeciwną stronę do pochodnej funkcji kosztu. W ten sposób liczymy żądaną zmianę dla poprzedniej warstwy. Poprzednia warstwa otrzymuje ten żądany wynik, liczy własną żądaną zmianę używając pochodnej funkcji sigmoid. W ten sposób propagujemy żądane zmiany wstecz, aż nie dojdziemy do początkowego obrazka. Wtedy odpowiednio modyfikujemy wszystkie wagi, mnożąc każdą żądaną zmianę przez parametr learning rate. Po wielu przeprocesowanych obrazach, sieć dostosuje swoje wagi, aby wartość funkcji kosztu była jak najmniejsza, czyli wynik był bliższy prawidłowemu.

- Wyniki

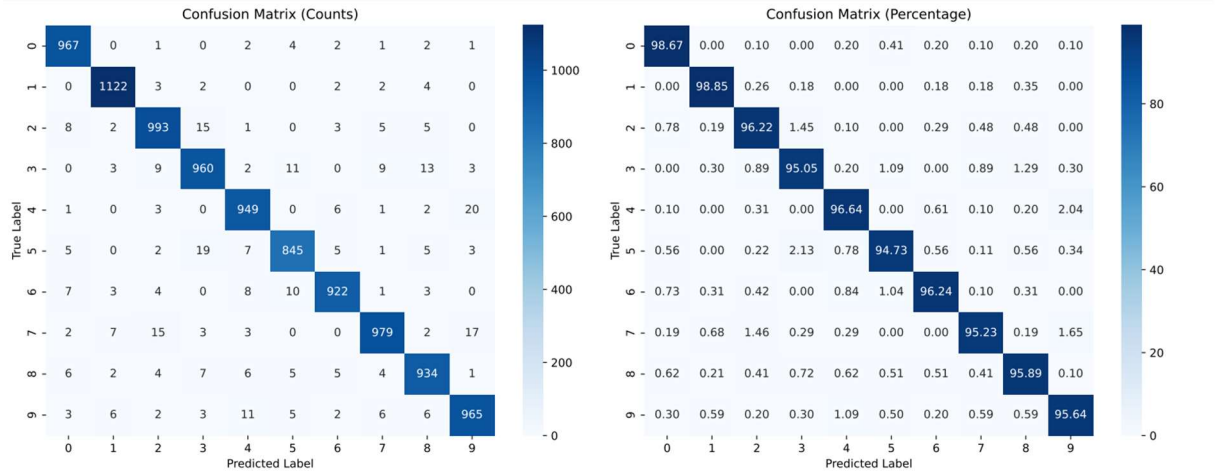
Poniższa sieć neuronowa została uruchomiona z parametrami:

  - Rozmiar 1 ukrytej warstwy – 128
  - Rozmiar 2 ukrytej warstwy - 64
  - Ilość epok – 20
  - Learning_rate – 0.1



Tak wygląda wykres skuteczności, czyli jaki procent testowych obrazków został zakwalifikowany poprawnie. Już po pierwszej epoce sieć osiągnęła nienajgorszy wynik rozpoznawania cyfr na poziomie ok. 87%, po drugiej było to już ponad 90%, a ostatecznie osiągnęło mniej więcej 96%.

Confusion-martix wygląda następująco:



Na jej podstawie można zauważyć, że sieć najlepiej rozpoznawała cyfrę 1, a najgorzej – 5.

Inną metodą na ocenianie wydajności sieci jest policzenie jej precyzji, rozrzutu, oraz f1-score dla każdej cyfry. Precyzję liczymy dzieląc ilość prawidłowo zakwalifikowanych przypadków przez sumę prawidłowo zakwalifikowanych przypadków, oraz ilości sytuacji, w których sieć zakwalifikowała inną cyfrę jako badaną. Rozrzut jest podobny, tylko w mianowniku do sumy prawidłowo zakwalifikowanych przypadków dodajemy ilość sytuacji, w której badana liczba została zakwalifikowana jako inna. F1-score jest zaś średnią geometryczną obu tych wartości. Po wyliczeniu tych parametrów dla każdej liczby okazuje się, że najwyższy f1-score dostała liczba 1, zaś najmniejszy – co ciekawe – 3 (5 miało wyższy wynik rozrzutu o ok. 1%.