Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Sieci neuronowe w zastosowaniach biomedycznych

Projekt

Dokumentacja wstępna

Kamil Gabryjelski, Antoni Różański

Prowadzący: mgr inż. Piotr Płoński

Warszawa, 2017

1. Opis projektu i jego rozwiązania

1.1. Temat i analiza zadania projektowego

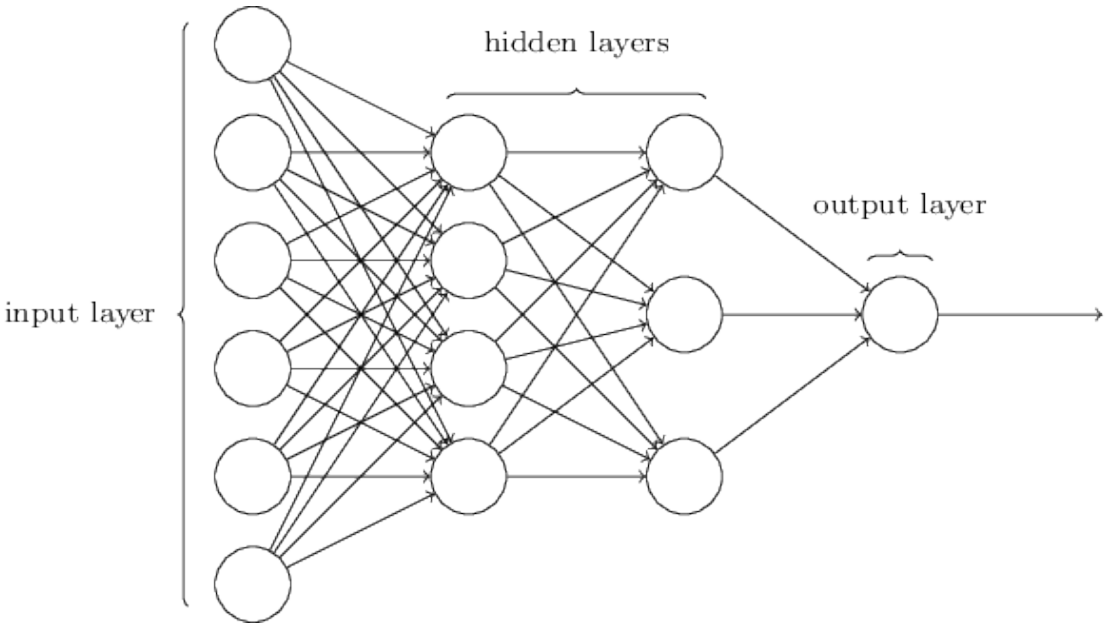
Zadaniem projektowym jest stworzenie systemu do rozpoznawania cyfr pisanych ręcznie z użyciem sieci MLP. Dane będą pochodzić ze zbioru MNIST.

Dane zadanie jest problemem odpowiedniego przyporządkowania obrazków do odpowiadających im kategorii na podstawie cyfr tworzonych przez ich piksele. Należy więc stworzyć klasyfikator, który poprawnie przydzieli dany obrazek do jednej z 10 klas (każda cyfra z przedziału $0-9$ tworzy jedną klasę). Zadanie zostanie rozwiązane poprzez stworzenie sieci neuronowej przy pomocy biblioteki TensorFlow.

1.2. Techniczne szczegóły zastosowanem sieci

1.2.1. Ogólna struktura sieci

Zadanie to rozwiążemy stosując sieć neuronową MLP (Multi Layer Perceptron), to jest sieć neuronową typu *feedforward*, zawierającą jedną lub więcej wartw ukrytych sztucznych neuronów. Jej struktura pokazana jest na rys. 1.1.



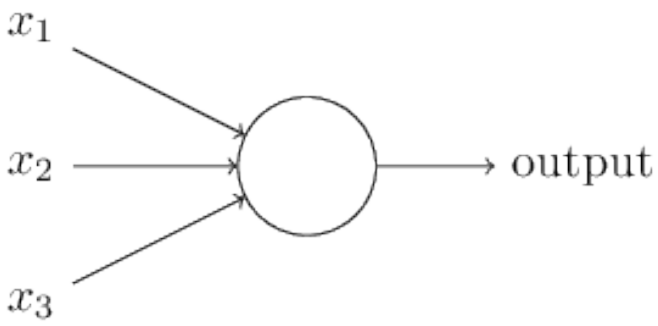
Rys. 1.1: Struktura sieci MLP

1.2.2. Sztuczny neuron

Pojedynczy neuron pokazany jest na rys. 1.2. Wykonuje on iloczyn skalarny wektorów tworzonych przez wartości wejść x_j i odpowiadające im wagi w_j po czym dodaje do niego wielkość *bias*:

$$\sum w_j x_j + bias \tag{1.1}$$

Tak otrzymana wartość jest podawana jako argument funkcji wzbudzenia neuronu. Dopiero wynik tej funkcji jest wynikiem końcowym neuronu, *output* na rys. 1.2.

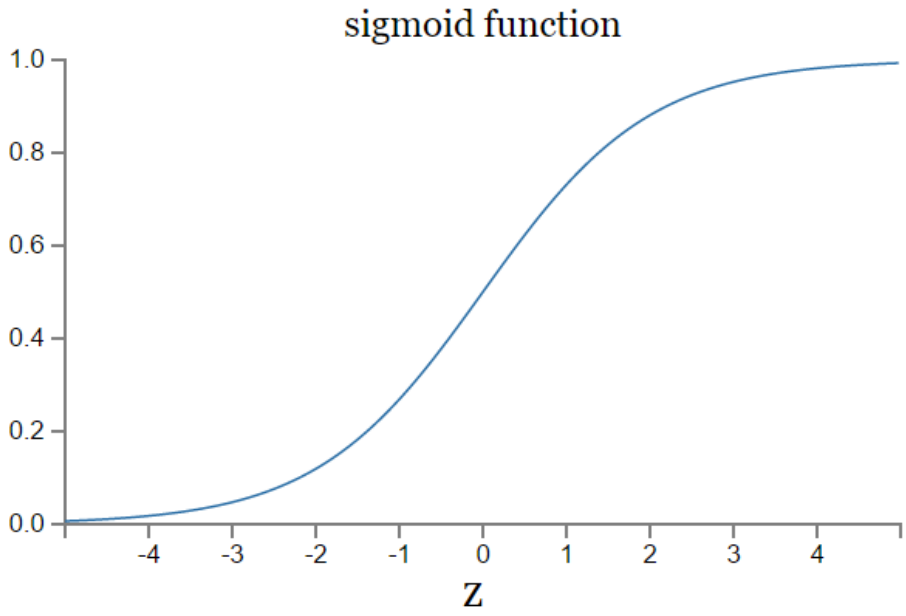


Rys. 1.2: Pojedyńczy neuron

1.2.3. Funkcja wzbudzenia neuronów

Zastosowana funkcja wzbudzenia naszej sieci MLP będzie miała postać funkcji sigmoidalnej, przedstawionej na rys. 1.3. Funkcja tej postaci zapewnia nam dwie korzyści:

1. Małe zmiany wejść neuronów będą skutkowały niewielkimi zmianami na wyjściu funkcji sigmoidalnej; pozwoli to kontrolować proces uczenia (w przypadku np. funkcji pobudzenia postaci funkcji skokowej, nawet małe zmiany wejść mogą powodować nieprzewidywalne zachowanie całej sieci);
2. Funkcja ta jest różniczkowalna, co jest kluczowe w algorytmie uczenia wykorzystującym mechanizm wstecznej propagacji (ang. *backpropagation*).



Rys. 1.3: Postać funkcji sigmoidalnej

1.2.4. Warstwa wejściowa

Zbiór MNIST składa się z 70,000 obrazków przedstawiających pisane ręcznie cyfry. Każdy z obrazków ma wymiary 28x28, co daje ilość $28 * 28 = 784$ pikseli na każdej ilustracji. Nasza sieć będzie posiadała dokładnie tyle wejść - każdy z neuronów warstwy wejściowej (*input layer* na rys. 1.1) będzie otrzymywał dane z jednego i tego samego dla wszystkich danych pikselu obrazka.

1.2.5. Warstwa wyjściowa

Warstwa wyjściowa natomiast (*output layer* na rys. 1.1) będzie składała się z 10 neuronów: każdy z nich będzie sygnalizował stopień przynależności obrazka do danej klasy.

Stopień przynależności, jako konsekwencja zastosowania sigmoidalnej funkcji wzbudzeń neuronów, będzie zawierał się w przedziale $< 0,1 >$. Największa wartości ze zbioru neuronów wyjściowych będzie identyfikowała aktualnie identyfikowaną cyfrę.

1.2.6. Proces uczenia

Do nauki sieci zostanie wykorzystany algorytm wstecznej propagacji (ang. *backpropagation*). Zadanie uczenie sieci przy pomocy tego algorytmu polaga na minimalizacji funckji straty (ang. *loss funtion*) jako funkcji wag i parametrów bias. Funkcja ta została przedstawiona jako wzór 1.2, gdzie: w - wagi, b - bias, n - liczebność zbioru uczącego, $y(x)$ - pożądane wyjście dla danego wejścia, a - rzeczywisty wynik działania sieci dla danego wejścia.

Algorytm ten polega na aplikowaniu do wag drobnych poprawek Δw wyliczonych ze wzoru 1.3, gdzie ∇C jest gradientem funkcji strat a η jest hiperparametrem wyrażającym wielkość kroku.

Skrótowo mówiąc, po każdej iteracji, to jest obliczeniu wyniku dla kolejno wszystkich danych uczących, obliczana jest funkcja $C(w, b)$ i jej gradient, a następnie zmieniane są wagi w kierunku przeciwnym do gradientu funcji strat o krok η.

$$C(w, b) \equiv \frac{1}{2n} \sum_x \|y(x) - a\|^2. \tag{1.2}$$

$$\Delta w = -\eta \nabla C, \tag{1.3}$$

1.3. Funkcjonalność

Nasz program będzie na podstawie dostarczonych danych (zbioru MNIST) uczył się rozpoznawać ręcznie pisane cyfry. Proces ten będzie raportowany za pomocą wykresów i statystyk. Gdy sieć zostanie już nauczona, będzie można wprowadzić jako dane wejściowe naszego programu dowolny obrazek, a ten zakomunikuje użytkownikowi, jaka cyfra znajduje się na tym obrazku.

1.4. Narzędzia

Narzędzia, z których zamierzamy korzystać:

1. Język programowania: Python
2. Zewnętrzne biblioteki: Tensorflow
3. Środowisko programistyczne - JetBrains Pycharm
4. Kontrola wersji - git/GitHub
5. Dokumentacja - narzędzia LaTeX
6. Komunikacja - Slack