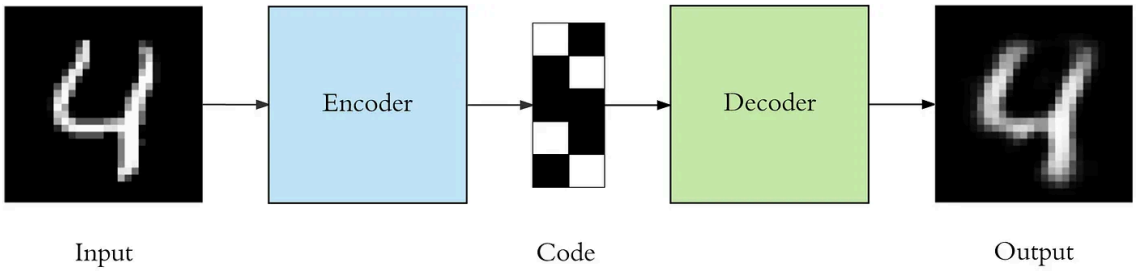
Lab 7 - Autoenkodery

1. Wstęp

Autoenkodery to specyficzny rodzaj sieci neuronowych typu feedforward, w których dane wejściowe są takie same jak dane wyjściowe. Kompensują one dane wejściowe do kodu o niższym wymiarze, a następnie rekonstruują dane wyjściowe na podstawie tej reprezentacji. Kod stanowi zwarty „zarys" lub „kompresję" danych wejściowych, nazywany także reprezentacją w przestrzeni latentnej.

Autoenkoder składa się z 3 komponentów: **enkodera, kodu** i **dekodera**. Koder kompresuje dane wejściowe i generuje kod, a dekoder rekonstruuje dane wejściowe, wykorzystując wyłącznie ten kod.



Aby zbudować autoenkoder, potrzebujemy 3 elementów: metody kodowania, metody dekodowania oraz funkcji straty do porównania wyjścia z celem.

Autoenkodery to głównie algorytm redukcji wymiarów (lub kompresji) z kilkoma istotnymi właściwościami:

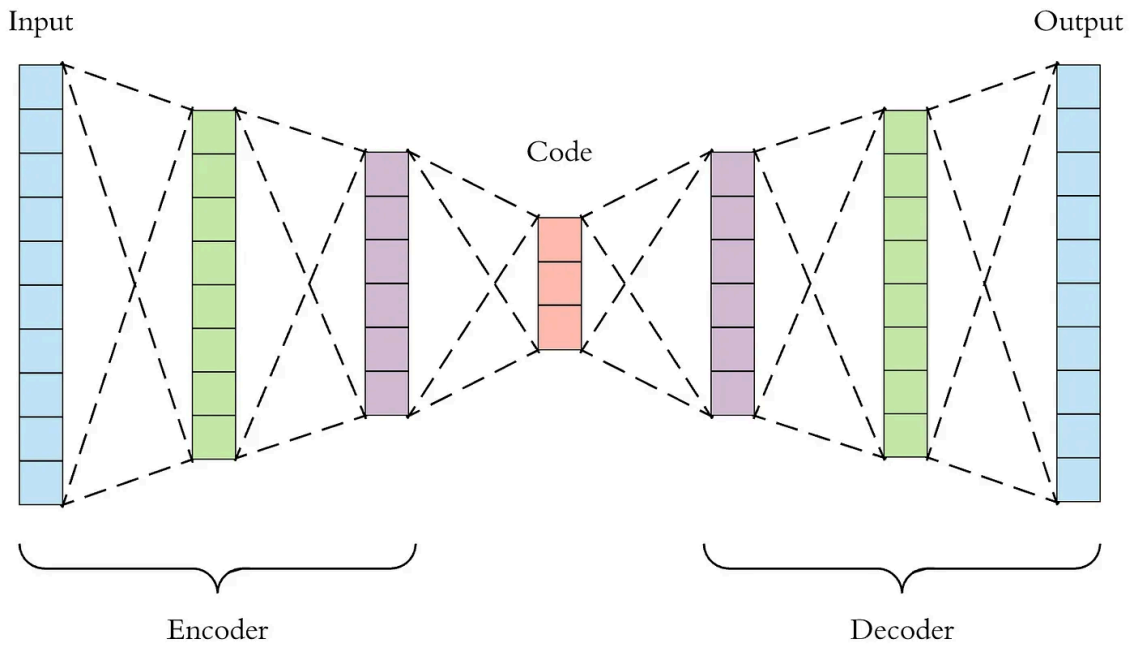
Specyficzne dla danych: Autoenkodery są w stanie skutecznie kompresować tylko dane podobne do tych, na których zostały wytrenowane. Ponieważ uczą się cech charakterystycznych dla określonego zbioru treningowego, różnią się od standardowych algorytmów kompresji, takich jak gzip.

stratne : Wynik działania autoenkodera nie będzie identyczny z danymi wejściowymi; będzie to ich bliska, ale nieco zdegradowana reprezentacja.

Nadzorowane pośrednio: Są uważane za technikę uczenia nienadzorowanego, ponieważ nie wymagają jawnych etykiet do nauki. Jednak bardziej precyzyjnie można je określić jako samonadzorowane, ponieważ generują własne etykiety na podstawie danych treningowych.

2. Architektura

Zarówno enkoder, jak i dekoder to w pełni połączone, jednokierunkowe sieci neuronowe. **Kod (przestrzeń latentna)** to pojedyncza warstwa sieci neuronowej o ustalonej liczbie wymiarów. Liczba neuronów w warstwie kodu (rozmiar kodu) to hiperparametr, który ustalamy przed rozpoczęciem trenowania autoenkodera.



Najpierw dane wejściowe przechodzą przez enkoder, który jest w pełni połączoną siecią neuronową, w celu wygenerowania kodu. Dekoder, mający podobną strukturę jak enkoder, generuje dane wyjściowe wyłącznie na podstawie kodu. Celem jest uzyskanie wyjścia identycznego z wejściem. Należy zauważyć, że architektura dekodera jest lustrzanym odbiciem enkodera.

Istnieją 4 hiperparametry, które musimy ustalić przed trenowaniem autoenkodera:

Rozmiar kodu (przestrzeni latentnej): liczba neuronów w warstwie środkowej. Mniejszy rozmiar oznacza większą kompresję.

Liczba warstw: autoenkoder może mieć dowolną liczbę warstw.

Liczba neuronów na warstwę: liczba neuronów w kolejnych warstwach enkodera maleje, a w dekoderze ponownie rośnie. Ponadto dekoder jest symetryczny względem enkodera pod względem struktury warstw.

Funkcja straty: MSE lub binarna entropia krzyżowa.

Autoenkodery są trenowane za pomocą algorytmu propagacji wstecznej.

3. **Zadanie**

Stwórz autoenkoder, który potrafi kompresować dane MNIST do przestrzeni latentnej o wymiarze 2, a następnie zrekonstruować obrazy oraz klasyfikować je na podstawie przestrzeni latentnej. Dodatkowo:

- Dodaj wizualizacje oryginalnych obrazów oraz ich rekonstrukcji,
- Dodaj wizualizację przestrzeni latentnej
- Przeprowadź test dekodowania na macierzy wartości losowych.

Protips:

- *Enkoder powinien przekształcać wejściowy obraz o rozmiarze 28x28x1 na wektor latentny o wymiarze 2, wykorzystując warstwę spłaszczającą Flatten, następnie warstwy gęste z aktywacją ReLU - np. 2 po 200, a na końcu warstwę wyjściową z aktywacją tanh.*
- *Dekoder powinien odtwarzać obraz o rozmiarze 28x28x1 z wektora latentnego o wymiarze 2, wykorzystując warstwy gęste z aktywacją ReLU, a na końcu warstwę wyjściową też z ReLU o rozmiarze 28x28x1*
- *Przetrenuj autoenkoder jako jeden model - połącz enkoder i dekoder za jako model sekwencyjny:*

  enc = encoder(n)
  dec = decoder(n)
  ae_model = Sequential([enc, dec])

  Można wykorzystać optymalizator Adam oraz MSE jako funkcje straty.

- *Przestrzeń latentną najlepiej zwizualizować za pomocą wykresu rozrzutu (scatter plot), z różnymi kolorami reprezentującymi poszczególne etykiety.*