UNIWERSYTET RZESZOWSKI
Kolegium Nauk Przyrodniczych

Imię i nazwisko: Krystian Zbyrad

Nr albumu: 113797

Kierunek: Informatyka

Grupa laboratoryjna: 3

Przedmiot: Sztuczna inteligencja

Model sieci neuronowej pozwalający rozpoznawać narysowane odręcznie cyfry, wytrenowany na podstawie zbioru *The Digit Dataset*

Prowadzący:

mgr Wojciech Gałka

Cel projektu:

Celem projektu jest stworzenie modelu opartego na sieciach neuronowych z wykorzystaniem przetwarzania obrazów, który po wytrenowaniu przez zbiór "*The Digits Dataset*" prawidłowo rozpozna odręcznie narysowane cyfry.

Opis zbioru treningowego:

Zbiór "*The Digits Dataset*" zawiera 1797 obrazów w rozdzielczości 8x8 pikseli, przedstawiających ręcznie narysowane cyfry. Atrybut "*Images*" zbioru przechowuje tablice (o wymiarach 8x8). Atrybut "*Target*" przechowuje wartości (cyfry), jakie każdy obraz reprezentuje.

Działanie programu:

Przyjmowane są dane wejściowe:

- 1797 obrazów w rozdzielczości 8x8 pikseli przestawiających ręcznie narysowane cyfry
- Obrazy przestawiające odręcznie narysowane cyfry w różnej rozdzielczości

Przygotowanie danych treningowych:

Dane są importowane z biblioteki "sklearn.datasets" za pomocą funcji "load_digits". Następnie dane w postaci tablic dwuwymiarowych o wymiarach 8x8 są zamieniane na tablice jednowymiarowe z 64 wartościami. Ostatnim krokiem przygotowywania jest pomieszanie danych oraz podzielenie zbioru na podzbiory: treningowy I testowy przy pomocy funkcji "train_test_split" z pakietu "sklearn.model_selection".

Dodatkowe dane testowe:

Wytrenowany model jest sprawdzany na dwóch zbiorach testowych, jednym wybranym jako część zbioru "Digits" oraz drugim, specjalnie przygotowanym, składającym się z 20 zdjęć odręcznie narysowanych cyfr. Obrazy w drugim zbiorze są najpierw odpowiednio przetwarzane oraz skalowane do rozmiaru 8x8.

Przygotowanie obrazów spoza zbioru "Digits":

Obrazy spoza zbioru "Digits" są importowane z dwóch folderów oraz zamieniane na tablie numpy. W tym celu każdy obraz jest konwertowany do skali szarości. Następnie tworzony jest negatyw obrazu poprzez przekształcenie wartości każdego piksela (wartość każdego piksela jest odejmowana od 255). Kolejnym krokiem jest zmniejszenie obrazu do wymiarów 8x8 przy pomocy algorytmu resamplingu "LANCZOS". Potem tworzony jest nowy obraz 8x8 wypełniony kolorem czarnym (wartości pikseli równe 0). Następnie oryginalny przeskalowany obraz jest wklejany na środek tego nowego obrazu. Dzięki temu, jeśli oryginalny obraz nie jest dokładnie 8x8, zostanie on wyśrodkowany. Ostatnim krokiem jest konwersja do tablicy numpy oraz normalizacja wartości obrazu do przedziału [0, 150]. Przedział ten został wybrany metodą prób I błędów jako najbardziej odpowiedni do późniejszego rozpoznawania przez wytrenowany model uczący.

Inicjacja oraz trenowanie modelu:

Tworzony jest model sekwencyjny, importowany z biblioteki “tensorflow.keras”, a następnie przypisane mu zostają wartości takie jak: rozmiar danych wejściowych równy liczbie cech zbioru treningowego (64) oraz kolejne warstwy wejściowe w pełni połączone z odpowiednią ilością neuronów oraz funkcjami aktywacji. Kolejnym krokiem jest kompilacja modelu z użyciem optymalizatora “Adam”, który jest populanym algorytmem optymalizacji. Funkcja straty modelu to “sparse categorical crossentropy” odpowiednia dla problemów klasyfikacji wieloklasowej, gdzie etykiety są liczbami całkowitymi. Jako metrykę oceny modelu podczas treningu i testowania używana jest dokładność “accuracy”.

Model jest trenowany na zbiorze testowym z odpowiednimi etykietami przez odpowiednią liczbę epok. Jest ustalany również “batch_size”, czyli wielkość partii po określoną liczbę próbek. Ostatnim przekazanym parametrem jest “validation_split” określający jaka część zbioru treningowego zostanie użyta jako zbiór walidacyjny służący do monitorowania wydajności modelu w trakcie treningu.

Predykcje:

Model dokonuje predykcji na danych testowych zwracając prawdopodobieństwa przynależności do klas dla każdej próbki. Następnie otrzymane predykcje są konwertowane na konkretne klasy poprzez wybranie indeksu z najwyższym prawdopodobieńswem dla każdej próbki.

Raport klasyfikacji oraz macierze błędów:

Ostatnim krokiem jest wygenerowanie szczegółowego raportu klasyfikacji zawierającego takie metryki jak: precyzja, czułość, “F1-score”, (średnia harmoniczna pomiędzy precyzją a czułością) oraz dokładność dla każdej klasy.

W celu sprawdzenia zgodności generowana jest również macierz pomyłek, która na podstawie predykcji oraz właściwych etykiet pokazuje, ile próbek z każdej klasy zostało poprawnie sklasyfikowanych (wartości na przekątnej) oraz ile zostało błędnie sklasyfikowanych (wartości poza przekątną).

Struktura modelu sieci neuronowej:

- Rozmiar danych wejściowych odpowiadający liczbie cech zbioru treningowego – równy 64
- Warstwy sieci:
    - Warstwa “gęsta” (Dense) – wszystkie neurony są połączone ze wszystkimi neuronami wartwy kolejnej – zawierająca 128 neuronów oraz z sigmoidalną funcją aktywacji
    - Warstwa “gęsta” (Dense) – wszystkie neurony są połączone ze wszystkimi neuronami wartwy kolejnej – zawierająca 64 neurony oraz z funcją aktywacji “relu”
    - Warstwa “gęsta” (Dense) – wszystkie neurony są połączone ze wszystkimi neuronami wartwy kolejnej – zawierająca 10 neuronów oraz z funcją aktywacji “softmax”

- Funkcje aktywacji:
    - Sigmoidalna – jej dziedziną jest zbiór wszystkich liczb rzeczywistych, ma nieujemną pochodną w każdym punkcie oraz pojedynczy punkt przegięcia (tz. Zmiana wypukłości funkcji), jest opisana wzorem $f(x) = \frac{1}{1+e^{-x}}$
    - ReLU – "Rectified Linear Unit", jest zdefiniowana wzorem $f(x) = \max(0,\ x)$
    - Sofmax – jest to znormalizowana funkcja wykładnicza, przekształca wektor K liczb rzeczywistych w rozkład prawdopodobieństwa K możliwych wyników, jest zdefiniowana wzorem $\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$ gdzie σ: $\mathbb{R}^K \longrightarrow (0,1)^K, K \geq 1$ oraz K przyjmuje wektor $z = (z_1, \dots, z_K) \in \mathbb{R}^K$
- Kompilacja modelu:
    - Optymalizator Adam (Adaptive Moment Estimation) - stochastyczna metoda gradientowego opadania, która opiera się na adaptacyjnej estymacji momentów pierwszego i drugiego rzędu. Łączy w sobie zalety dwóch innych optymalizatorów: RMSprop i klasycznego momentum, dzięki czemu efektywnie i skutecznie prowadzi do minimalizacji funkcji kosztu.
    - Funkcja straty modelu "sparse categorical crossentropy" – odpowiednia dla problemów klasyfikacji wieloklasowej, etykiety są liczbami całkowitymi
    - Metryka oceny modelu – dokładność (accuracy), jeden z najczęściej stosowanych wskaźników w ocenie jakości modelu klasyfikacyjnego. Jest obliczana wzorem $\frac{TP+TN}{TP+TN+FP+FN}$ gdzie:
        - TP (True Positives) – liczba prawdziwie pozytywnych przypadków
        - TN (True Negatives) – liczba prawdziwie negatywnych przypadków
        - FP (False Positives) – liczba fałszywie pozytywnych przypadków
        - FN (False Negatives) – liczba fałszywie negatywnych przypadków

      Przykład obliczania: Załóżmy, że mamy zestaw danych testowych składający się z 100 próbek, z których 70 zostało poprawnie sklasyfikowanych jako pozytywne (TP) i 20 jako negatywne (TN). Pozostałe 10 próbek zostało błędnie sklasyfikowanych: 5 jako pozytywne (FP) i 5 jako negatywne (FN). Wówczas dokładność będzie wynosić: $\frac{70+20}{70+20+5+5} = \frac{90}{100} = 0.9$ czyli 90%.
- Trenowanie modelu:
    - Dane treningowe wraz z etykietami są przekazywane do modelu
    - Model jest trenowany przez 40 epok
    - W każdej epoce model przechodzi przez cały zbiór treningowy w podzbiorach (batchach) po 32 próbki. Oznacza to, że gradinety będą obliczane, a wagi aktualizowane po każdym podzbiorze 32 próbek.
    - 10% danych treningowych jest zachowywane do walidacji modelu po każdej epoce trenowania, co pozwala na monitorowanie wydajności modelu podczas procesu trenowania.
- Predykcje:

  Wytrenowany model jest sprawdzany na zbiorze testowym, a następnie prawdopodobieństwa są przekształcane na etykiety klas.

Macierze błędów oraz raporty klasyfikacji:

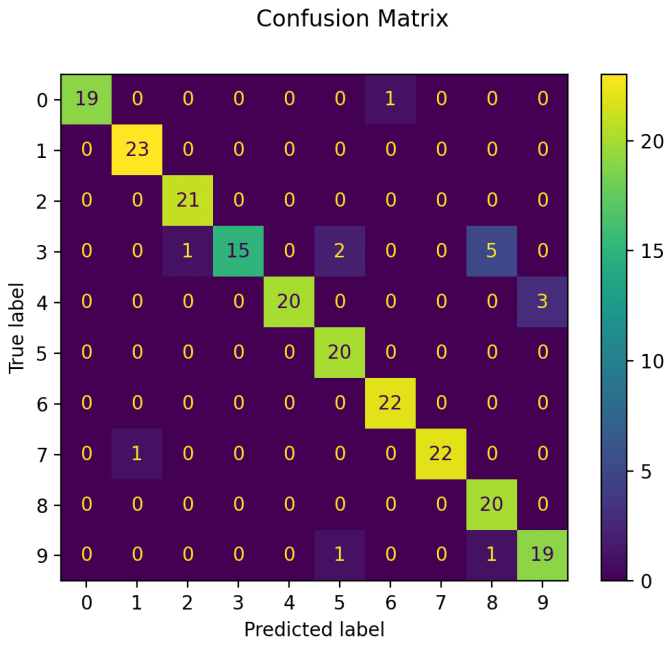
W celu sprawdzenia dokładności sieci generowany jest raport klasyfikacji obrazujący dokładność dla każdej klasy oraz zawierającego takie metryki jak: precyzja, czułość, “F1-score”, (średnia harmoniczna pomiędzy precyzją a czułością) i liczba próbek.

	precision	recall	f1-score	support
0	1.00	0.95	0.97	20
1	1.00	1.00	1.00	23
2	0.95	0.95	0.95	21
3	0.94	0.65	0.77	23
4	0.91	0.87	0.89	23
5	0.83	1.00	0.91	20
6	0.96	1.00	0.98	22
7	0.96	0.96	0.96	23
8	0.79	0.95	0.86	20
9	0.86	0.86	0.86	21
accuracy			0.92	216
macro avg	0.92	0.92	0.91	216
weighted avg	0.92	0.92	0.91	216

Rysunek 1. Raport klasyfikacji dla zbioru testowego wybranego ze zbioru "Digits"

Na Rysunku 1 przedstawione zostały miary oceny modelu dla każdej z klas w zbiorze testowym. Ogólna dokładność modelu wynosi 92%, a średnie wartości makro I ważone dla precyzji, czułości I miary F1 są również zbliżone do 92% co wskazuje na dobrą wydajność modelu dla większości klas.

Macierz pomyłek opisuje wydajność modelu klasyfikacyjnego. Przedstawia ona wizualnie jak często wyniki klasyfikacji są poprawne oraz błędne. Macierz ta zawiera informacje o przewidywanych klasyfikacjach dokonanych przez model oraz rzeczywistych.



Rysunek 2. Macierz pomylek dla zbioru testowego wybranego ze zbioru "Digits"

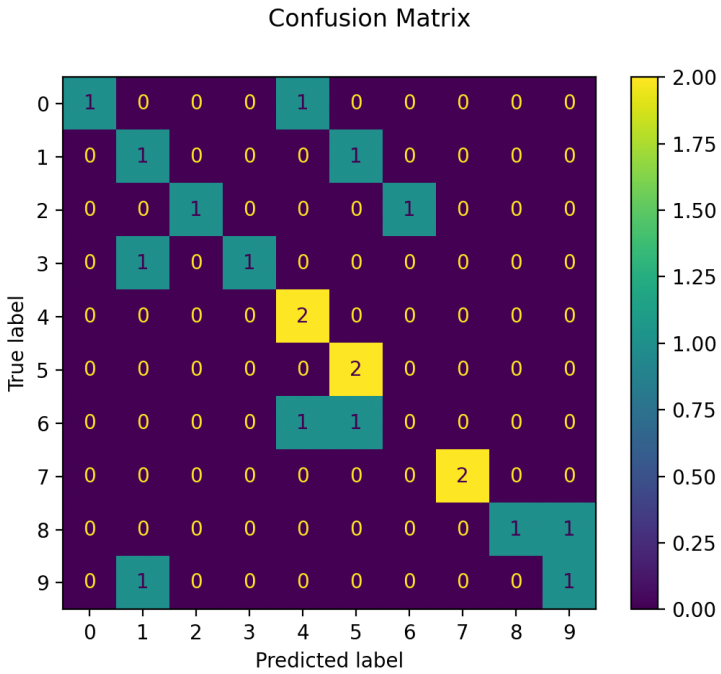
Macierz pomyłek przedstawiona na Rysunku 2 ilustruje, że model klasyfikacyjny w większości przypadków poprawnie przewiduje klasy, co widać po wysokich wartościach na przekątnej macierzy, takich jak 19, 23, 21 itd. Niewielka liczba pomyłek występuje poza przekątną, na przykład jedna próbka klasy 0 została błędnie sklasyfikowana jako klasa 9, a pięć próbek klasy 3 zostało błędnie sklasyfikowanych jako klasa 8.

Na Rysunku 3 został przedstawiony raport klasyfikacji dla obrazów dla zbioru testowego nie należącego do zbioru “Digits”.

	precision	recall	f1-score	support
0	1.00	0.50	0.67	2
1	0.33	0.50	0.40	2
2	1.00	1.00	1.00	2
3	1.00	0.50	0.67	2
4	0.50	1.00	0.67	2
5	0.67	1.00	0.80	2
6	0.00	0.00	0.00	2
7	1.00	1.00	1.00	2
8	1.00	0.50	0.67	2
9	0.50	0.50	0.50	2
accuracy			0.65	20
macro avg	0.70	0.65	0.64	20
weighted avg	0.70	0.65	0.64	20

Rysunek 3. Raport klasyfikacji dla zbioru testowego spoza zbioru "Digits"

Powyższy raport klasyfikacji pokazuje, że model osiągnął ogólną dokładność (accuracy) wynoszącą 65%. Precyzja, czułość i miara F1 są różne dla poszczególnych klas: niektóre klasy, jak klasa 2, 7 i 5, mają wysokie wartości wszystkich miar, podczas gdy klasy takie jak 1 i 6 wykazują niską wydajność, z zerową czułością dla klasy 6. Średnie wartości makro i ważone (macro avg i weighted avg) dla precyzji, czułości i miary F1 wynoszą około 0.65, co wskazuje na zróżnicowaną wydajność modelu w zależności od klasy.



Rysunek 4. Macierz pomylek dla zbioru testowego spoza zbioru "Digits"

Macierz pomyłek zamieszczona na Rysunku 4 przedstawia klasyfikację dla 10 klas, z każdą klasą reprezentowaną przez 2 próbki (łącznie 20 próbek). Większość próbek została poprawnie sklasyfikowana, co widać po wartościach na przekątnej, takich jak 1 lub 2, jednakże występuje kilka błędnych klasyfikacji.

Wnioski:

Projekt wykorzystujący sieci neuronowe do rozpoznawania ręcznie pisanych cyfr na zbiorze "The Digits Dataset" osiągnął wysoką dokładność na zbiorze testowym wylosowanym ze zbioru "Digits" (92%), co potwierdzają wartości precyzji, czułości i F1-score. Dodatkowy test na obrazach spoza zbioru "Digits" wykazał niższą dokładność równą 65%. Jest to prawdopodobnie spowodowane zbyt małym zróżnicowaniem zbioru uczącego. Użycie różnych funkcji aktywacji i optymalizatora Adam podczas trenowania modelu zapewniło skuteczne uczenie się dzięki czemu uzyskano wysoką dokładność.