Sprawozdanie Jakub Świerczyński, Karolina Starzec

1. **Cel**

Celem było zbudowanie lekkiej aplikacji Streamlit, która klasyfikuje obrazy owoców/warzyw (zbiór Fruit-360 100 × 100 px) przy użyciu modelu w formacie ONNX. Końcowy produkt uruchamiany jest zarówno lokalnie, jak i w kontenerze Docker.

1. **Zbiór danych i ich przygotowanie**

- Pozyskanie danych

Model został wytrenowany na otwartym zbiorze Fruit-360, który zawiera surowe zdjęcia owoców w formacie RGB o rozdzielczości 100×100 pikseli. Dane są zorganizowane w podfolderach odpowiadających konkretnym odmianom owoców (np. *Apple Braeburn*, *Banana*, *Avocado*), co ułatwia wczytywanie ich z użyciem narzędzi typu ImageDataGenerator.

- Pogrupowanie danych

Oryginalny zbiór zawiera wiele szczegółowych odmian jednego gatunku (np. różne typy jabłek: *Apple Red 1*, *Apple Granny Smith*). W celu uproszczenia problemu klasyfikacji i zwiększenia liczebności klas, dokonano mapowania tych odmian na 82 ogólne kategorie (np. apple, banana, blueberry).

- Podział danych

Z każdej z 82 klas losowo wydzielono:

- 70% danych do zbioru treningowego – wykorzystywanego podczas nauki modelu,

- 15% do zbioru walidacyjnego – używanego do monitorowania jakości modelu w trakcie treningu i optymalizacji hiperparametrów,

- 15% do zbioru testowego – używanego tylko po zakończeniu treningu, aby uzyskać niezależną ocenę skuteczności modelu.

- Dlaczego dzielimy dane na zbiory?

- Zbiór treningowy umożliwia modelowi naukę reprezentacji i wzorców charakterystycznych dla poszczególnych klas.

- Zbiór walidacyjny pozwala ocenić postęp treningu w sposób niezależny, zapobiegając przeuczeniu i umożliwiając dobór parametrów takich jak: współczynnik uczenia (learning rate), liczba epok, dropout czy architektura sieci.

- Zbiór testowy pozostaje „niewidoczny” aż do końca i daje obiektywną informację o generalizacji.  
Podział 70/15/15 jest kompromisem: 70 % zapewnia wystarczającą różnorodność do nauki, a po 15 % daje stabilną statystykę dla walidacji i testu, nie marnując zbyt wielu przykładów.

1. **Wybór i implementacja modelu AI**

- Wybór modelu

Wybrano model prostej sieci konwolucyjnej (CNN), który dobrze sprawdza się w zadaniach klasyfikacji obrazów z warstwami Conv, MaxPooling2D, Flatten, Dense oraz Dropout, tworząc klasyczną architekturę do rozpoznawania obrazów.

- Architektura

|  |  |  |
| --- | --- | --- |
| Blok | Parametry | Uzasadnienie |
| Conv 1 | Conv2d(3 → 32, k=3, p=1) + ReLU | mały filtr 3 × 3 wychwytuje lokalne krawędzie przy minimalnej liczbie wag (0.9 k) |
| Pool 1 | MaxPool2d(2) | redukcja wymiaru do 50 × 50, tłumi przesunięcia |
| Conv 2 | Conv2d(32 → 64, k=3, p=1) + ReLU | podwaja kanały → głębsze cechy (tekstury) |
| Pool 2 | MaxPool2d(2) | 25 × 25 px, dalsza redukcja |
| Conv 3 | Conv2d(64 → 128, k=3, p=1) + ReLU | 128 kanałów wystarcza, by uchwycić zróżnicowanie 82 gatunków |
| Pool 3 | MaxPool2d(2) → tensor 128 × 12 × 12 | przygotowanie do warstwy w pełni połączonej |
| Classifier | Flatten → Linear(18432 → 256) + ReLU → Dropout(0.3) → Linear(256 → 82) | *FC 256* pozwala na kompresję cech; *Drop-out 0.3* (losowe „znikanie” neuronów) ogranicza przeuczenie. |

- Hiperparametry

|  |  |  |
| --- | --- | --- |
| Kategoria | Wartość | Dlaczego tak? |
| Optymalizator | Adam (β₁ = 0.9, β₂ = 0.999) | adaptacyjne LR przyspiesza zbieżność na małych batchach. |
| Learning Rate | 1 × 10⁻³ | przy LR = 1e-2 model divergował, przy 1e-4 zbiegał zbyt wolno. |
| Batch size | 32 | mieści się w VRAM 8 GB i stabilizuje estymację gradientu. |
| Epoki | 20 + *early-stop* (zapis „best.pth” gdy val\_loss ↓) | po 17 epokach poprawa < 0.1 pp → zatrzymano. |
| Augmentacje | RandomHorizontalFlip(), RandomRotation(10°) (komórka 5) | zwiększają wariancję bez przesuwania klasy (owoce symetryczne). |
| Normalizacja | (x–0.5)/0.5 ⇒ [-1,1] | środek 0, jednostkowa skala kanałów – stabilniejsze gradienty. |
| Dropout | 0.3 | empiria: 0.2 skutkowało przeuczeniem, 0.4 spowalniało konwergencję. |
| Seed | 42 | zapewnia powtarzalność wyników przy losowym shufflu i augmentacjach. |

1. **Ocena wyników modelu**

Skuteczność na zbiorze testowym (6 896 obrazów)

Accuracy = 99.94 % – tylko 4 błędy na cały zbiór.

Jednostkowe pomyłki dotyczą wyłącznie klas *Banana*, *Strawberry* i *Nut Pecan* (recall 0.99).

Macierz pomyłekA chart with a purple square with text

AI-generated content may be incorrect.

1. **Wdrożenie modelu i monitorowanie**

W celu uruchomienia modelu w środowisku produkcyjnym dokonano jego eksportu do formatu ONNX.

Aplikacja do klasyfikacji obrazów została stworzona przy użyciu frameworka Streamlit, który pozwala na szybkie prototypowanie interaktywnych interfejsów użytkownika. Użytkownik może przesłać obraz w formacie JPG lub PNG, a model ONNX dokona predykcji i wyświetli najbardziej prawdopodobną klasę wraz z poziomem pewności.

Całość została umieszczona w lekkim kontenerze Docker, co umożliwia szybkie i powtarzalne wdrożenie systemu na dowolnej maszynie (lokalnie, w chmurze, na serwerze).

A screenshot of a computer

AI-generated content may be incorrect.

A blueberry on a white background

AI-generated content may be incorrect.