МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ

ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ

ТЕХНОЛОГІЙ

Проект №4

З дисципліни

«Штучний інтелект»

Виконав:

Студент групи ПД-44

Солов'ян Арсен

Київ – 2025

Опис проекту:

Розробка системи класифікації та аналізу зображень за допомогою нейронних мереж та/або згорткових нейронних мереж (CNN).

Створити інтелектуальну систему на основі нейронних мереж для автоматизованої обробки даних (зображень, сигналів або текстів) та виконати:

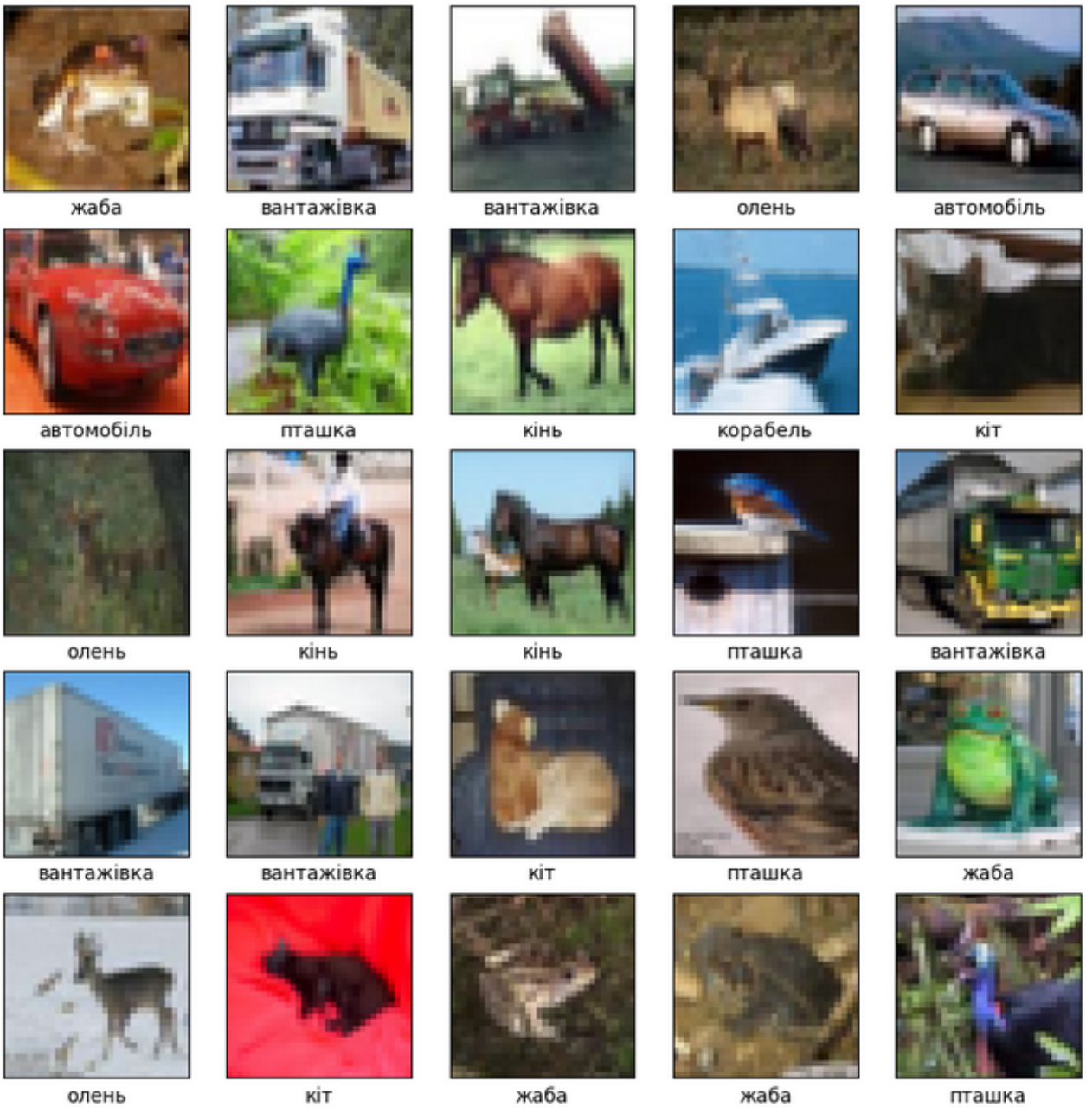
- класифікацію зображень на категорії;
- розпізнавання об’єктів на зображеннях;
- покращення якості зображень (стилізація, колоризація, реставрація, тощо).

1. Аналіз даних:

1) Підготовка датасету: Датасет CIFAR-10 був успішно завантажений за допомогою функції cifar10.load_data(). Це забезпечило набір зображень для навчання та тестування моделі.

2) Анотація та маркування даних: Датасет CIFAR-10 є вже анотованим та маркованим. Кожне зображення має відповідну мітку, що вказує на його категорію (наприклад жаба, автомобіль). Це підтверджується візуалізацією перших 25 зображень з їхніми мітками:

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 ━━━━━━━━━━━━━━━━━━━━ 2s 0us/step
Розмір навчального набору зображень: (50000, 32, 32, 3)
Розмір навчального набору міток: (50000, 1)
Розмір тестового набору зображень: (10000, 32, 32, 3)
Розмір тестового набору міток: (10000, 1)
```



2. Збір та обробка даних:
1) Збір вихідного датасету: Вихідний датасет CIFAR-10 був зібраний шляхом його завантаження за допомогою вбудованої функції Keras.
2) Попередня обробка даних: Була виконана попередня обробка даних, яка включала нормалізацію значень пікселів зображень до діапазону [0, 1] шляхом ділення на 255.0. Також було виконано перетворення міток у формат one-hot encoding за допомогою функції to_categorical.
3) Формування навчальної, валідаційної та тестової вибірок: Датасет CIFAR-10 автоматично розділяється на навчальну (50,000 зображень) та тестову (10,000 зображень) вибірки при завантаженні. Крім того, під час навчання моделі було налаштовано використання 20% навчальних даних як валідаційної вибірки (validation_split=0.2).

3. Модель нейронної мережі:
1) Проектування архітектури CNN: Була спроектована послідовна архітектура згорткової нейронної мережі (CNN) з кількома згортковими шарами, шарами максполінгу, шарами пакетної нормалізації, шаром згладжування та повністю зв'язаними шарами. Детальна архітектура моделі була виведена за допомогою model.summary():

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
batch_normalization (BatchNormalization)	(None, 15, 15, 32)	128
conv2d_1 (Conv2D)	(None, 13, 13, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 6, 6, 64)	0
batch_normalization_1 (BatchNormalization)	(None, 6, 6, 64)	256
conv2d_2 (Conv2D)	(None, 4, 4, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 2, 2, 128)	0
batch_normalization_2 (BatchNormalization)	(None, 2, 2, 128)	512
flatten (Flatten)	(None, 512)	0
dense (Dense)	(None, 128)	65,664
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 10)	1,290

Total params: 161,098 (629.29 KB)
Trainable params: 160,650 (627.54 KB)
Non-trainable params: 448 (1.75 KB)

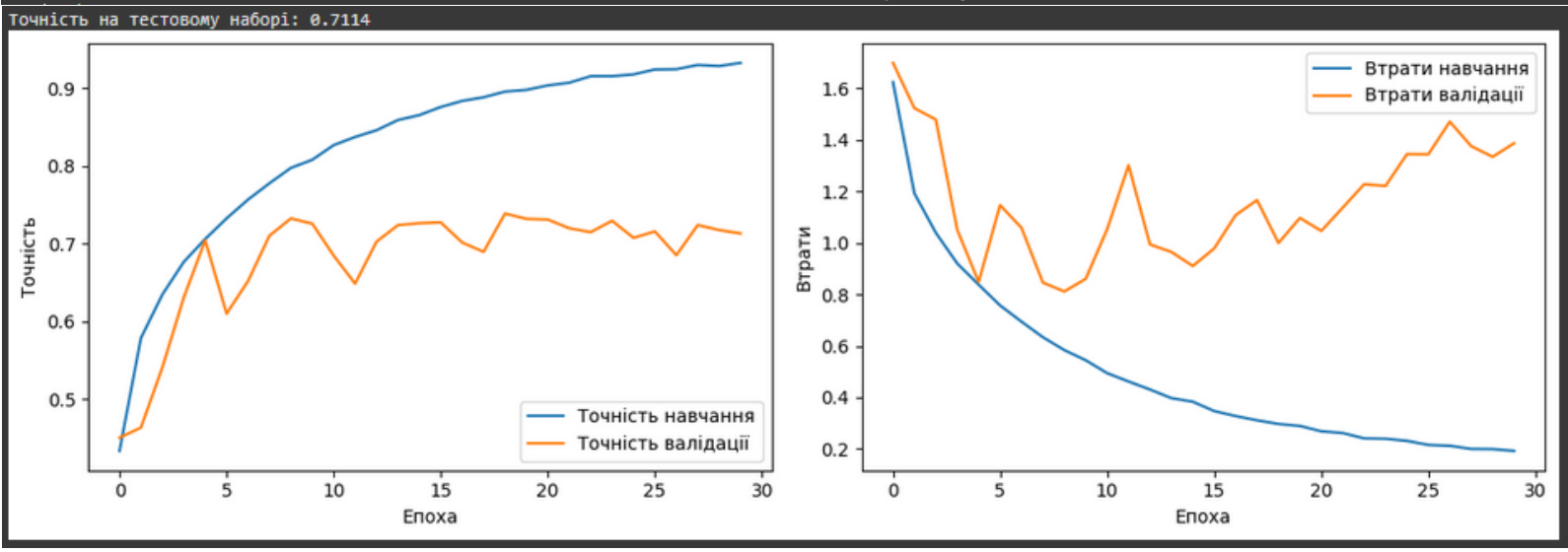
2) Реалізація регуляризації (Batch Normalization): У спроектованій архітектурі CNN було використано шари BatchNormalization після кожного згорткового шару для запобігання перенавчанню.

4. Обробка даних:
1) Застосування попередньої обробки даних: Застосовано попередню обробку даних, включаючи нормалізацію.
2) Реалізація функцій для читання, обробки та візуалізації даних: Для читання даних використовувалася функція cifar10.load_data(); Для обробки застосовувалася нормалізація та one-hot encoding; Для візуалізації використовувалася бібліотека matplotlib.pyplot для відображення перших кількох зображень з їхніми мітками;

5. Навчання моделі:
1) Використання готових бібліотек: Для реалізації та навчання моделі використовувалися готові бібліотеки TensorFlow та Keras.
2) Налаштування гіперпараметрів: Були налаштовані наступні гіперпараметри: кількість епох навчання (epochs=30), розмір міні-пакету (batch_size=64), оптимізатор (adam), функція втрат (categorical_crossentropy) та метрика оцінки (accuracy).

6. Оцінка моделі:
1) Проведення валідації на тестовому наборі: Під час навчання моделі використовувалася валідаційна вибірка (20% навчальних даних) для оцінки її продуктивності на кожній епосі.
2) Оцінка метрик якості: Після завершення навчання модель була оцінена на тестовому наборі даних за допомогою функції model.evaluate(). В якості метрики якості використовувалася точність (accuracy).

```
Epoch 1/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 69s 102ms/step - accuracy: 0.3526 - loss: 1.9237 - val_accuracy:
Epoch 2/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 80s 99ms/step - accuracy: 0.5654 - loss: 1.2286 - val_accuracy:
Epoch 3/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 81s 98ms/step - accuracy: 0.6337 - loss: 1.0450 - val_accuracy:
Epoch 4/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 83s 100ms/step - accuracy: 0.6767 - loss: 0.9154 - val_accuracy:
Epoch 5/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 62s 99ms/step - accuracy: 0.7074 - loss: 0.8281 - val_accuracy:
Epoch 6/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 83s 102ms/step - accuracy: 0.7389 - loss: 0.7418 - val_accuracy:
Epoch 7/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 62s 99ms/step - accuracy: 0.7637 - loss: 0.6776 - val_accuracy:
Epoch 8/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 62s 99ms/step - accuracy: 0.7855 - loss: 0.6060 - val_accuracy:
Epoch 9/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 82s 98ms/step - accuracy: 0.8043 - loss: 0.5678 - val_accuracy:
Epoch 10/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 83s 100ms/step - accuracy: 0.8160 - loss: 0.5291 - val_accuracy:
Epoch 11/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 83s 101ms/step - accuracy: 0.8325 - loss: 0.4723 - val_accuracy:
Epoch 12/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 62s 99ms/step - accuracy: 0.8427 - loss: 0.4430 - val_accuracy:
Epoch 13/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 82s 99ms/step - accuracy: 0.8531 - loss: 0.4103 - val_accuracy:
Epoch 14/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 84s 102ms/step - accuracy: 0.8669 - loss: 0.3748 - val_accuracy:
Epoch 15/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 81s 100ms/step - accuracy: 0.8739 - loss: 0.3591 - val_accuracy:
Epoch 16/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 63s 100ms/step - accuracy: 0.8819 - loss: 0.3252 - val_accuracy: 0.7272 - val_loss: 0.9788
Epoch 17/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 81s 99ms/step - accuracy: 0.8898 - loss: 0.3093 - val_accuracy: 0.7014 - val_loss: 1.1082
Epoch 18/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 81s 98ms/step - accuracy: 0.8931 - loss: 0.2950 - val_accuracy: 0.6894 - val_loss: 1.1666
Epoch 19/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 85s 102ms/step - accuracy: 0.9021 - loss: 0.2775 - val_accuracy: 0.7386 - val_loss: 1.0000
Epoch 20/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 81s 100ms/step - accuracy: 0.9055 - loss: 0.2671 - val_accuracy: 0.7318 - val_loss: 1.0971
Epoch 21/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 83s 103ms/step - accuracy: 0.9140 - loss: 0.2410 - val_accuracy: 0.7309 - val_loss: 1.0469
Epoch 22/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 79s 98ms/step - accuracy: 0.9133 - loss: 0.2432 - val_accuracy: 0.7197 - val_loss: 1.1373
Epoch 23/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 83s 100ms/step - accuracy: 0.9204 - loss: 0.2209 - val_accuracy: 0.7147 - val_loss: 1.2282
Epoch 24/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 62s 99ms/step - accuracy: 0.9207 - loss: 0.2247 - val_accuracy: 0.7293 - val_loss: 1.2215
Epoch 25/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 82s 99ms/step - accuracy: 0.9265 - loss: 0.2047 - val_accuracy: 0.7074 - val_loss: 1.3454
Epoch 26/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 64s 102ms/step - accuracy: 0.9265 - loss: 0.2069 - val_accuracy: 0.7156 - val_loss: 1.3447
Epoch 27/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 62s 99ms/step - accuracy: 0.9325 - loss: 0.1900 - val_accuracy: 0.6851 - val_loss: 1.4713
Epoch 28/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 62s 100ms/step - accuracy: 0.9329 - loss: 0.1928 - val_accuracy: 0.7238 - val_loss: 1.3761
Epoch 29/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 83s 102ms/step - accuracy: 0.9330 - loss: 0.1870 - val_accuracy: 0.7174 - val_loss: 1.3347
Epoch 30/30
625/625 ━━━━━━━━━━━━━━━━━━━━ 62s 100ms/step - accuracy: 0.9396 - loss: 0.1728 - val_accuracy: 0.7130 - val_loss: 1.3873
313/313 - 4s - 13ms/step - accuracy: 0.7114 - loss: 1.3948
Точність на тестовому наборі: 0.7114
```



Аналіз:
1) Точність навчання: Точність на навчальній вибірці зростала протягом 30 епох і досягла значення близько 94%. Це свідчить про те, що модель добре навчилася розпізнавати образи на навчальних даних.
2) Точність валідації: Точність на валідаційній вибірці також зростала, але досягла плато приблизно на рівні 73% і навіть почала дещо знижуватися на останніх епохах. Це може бути ознакою перенавчання, коли модель починає занадто добре запам'ятовувати навчальні дані, але гірше узагальнює нові, невідомі дані.
3) Втрати навчання: Втрати на навчальній вибірці постійно зменшувалися протягом навчання, що є очікуваним результатом.
4) Втрати валідації: Втрати на валідаційній вибірці спочатку зменшувалися, але потім почали зростати після певної кількості епох. Це також підтверджує можливість перенавчання.
5) Точність на тестовому наборі: Після завершення навчання модель була оцінена на тестовому наборі, і досягнута точність склала близько 71.14%. Це є показником того, як модель буде працювати на абсолютно нових, раніше не бачених даних.

Висновок:
Модель досягла досить високої точності на навчальних даних, але її здатність до узагальнення є нижчою. Ймовірно, модель почала перенавчатися після певної кількості епох. Незважаючи на ознаки перенавчання, досягнута точність на тестовому наборі (71.14%) є досить непоганим результатом для базової CNN на датасеті CIFAR-10 після 30-и епох навчання.