

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики
Кафедра прикладної математики

Звіт
із лабораторної роботи
із дисципліни «СИСТЕМИ ГЛИБИННОГО НАВЧАННЯ»

на тему:
Розпізнавання двовимірних кольорових об'єктів за допомогою згорткової
нейронної мережі

Виконав:
студент групи КМ-13
Онищенко В.С.

Перевірів:
Терейковський І. А.

Київ — 2024

ЗМІСТ

1	Мета роботи	3
2	Теорія	4
3	Код програми	5
3.1	Код навчання нейронної мережі	5
3.2	Код розпізнавання об'єктів	8
4	Результати	10
4.1	Точність моделі	10
4.2	Скріншот результату	10
5	Висновки	12

1. Мета роботи

Навчитися створювати та застосовувати згорткову нейронну мережу для розпізнавання двовимірних кольорових об'єктів з використанням набору даних CIFAR-10.

2. Теорія

Згорткова нейронна мережа (CNN) — це тип глибокої нейронної мережі, який використовується для обробки структурованих даних, таких як зображення.

Основні компоненти CNN:

- **Згорткові шари (Conv2D):** визначають просторові характеристики об'єктів на зображеннях.
- **Шари пулінгу (MaxPooling2D):** зменшують розмірність даних, зберігаючи суттєву інформацію.
- **Dropout:** запобігає перенавчанню моделі шляхом випадкового обнулення нейронів.
- **Щільно з'єднані шари (Dense):** виконують класифікацію після обробки даних згортковими шарами.

3. Код програми

3.1. Код навчання нейронної мережі

```
1 import numpy as np
2 from keras.datasets import cifar10
3 from keras.models import Sequential
4 from keras.layers import Dense, Flatten, Dropout, Conv2D, MaxPooling2D
5 from tensorflow.keras.utils import to_categorical
6 from keras.optimizers import SGD
7
8 # Встановлення фіксованого значення для генератора випадкових чисел
9 np.random.seed(42)
10
11 # Завантаження даних
12 (X_train, y_train), (X_test, y_test) = cifar10.load_data()
13
14 # Параметри моделі
15 batch_size = 32
16 nb_classes = 10
17 nb_epoch = 25
18 img_rows, img_cols = 32, 32
19 img_channels = 3
20
21 # Нормалізація даних
22 X_train = X_train.astype('float32') / 255
23 X_test = X_test.astype('float32') / 255
24
25 # Перетворення міток у формат one-hot
26 Y_train = to_categorical(y_train, nb_classes)
27 Y_test = to_categorical(y_test, nb_classes)
28
29 # Створення моделі
30 model = Sequential()
31
32 # Додавання шарів згорткової нейромережі
33 model.add(Conv2D(32, (3, 3), padding='same', activation='relu',
```

```

34         input_shape=(img_rows, img_cols, img_channels)))
35 model.add(Conv2D(32, (3, 3), padding='same', activation='relu'))
36 model.add(MaxPooling2D(pool_size=(2, 2)))
37 model.add(Dropout(0.25))
38
39 model.add(Conv2D(64, (3, 3), padding='same', activation='relu'))
40 model.add(Conv2D(64, (3, 3), activation='relu'))
41 model.add(MaxPooling2D(pool_size=(2, 2)))
42 model.add(Dropout(0.25))
43
44 model.add(Flatten())
45 model.add(Dense(512, activation='relu'))
46 model.add(Dropout(0.5))
47 model.add(Dense(nb_classes, activation='softmax'))
48
49 # Компіляція моделі
50 sgd = SGD(learning_rate=0.01, decay=1e-6, momentum=0.9, nesterov=True)
51 model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
52
53 # Навчання моделі
54 model.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch,
55         validation_split=0.1, shuffle=True, verbose=2)
56
57 # Оцінка на тестових даних
58 scores = model.evaluate(X_test, Y_test, verbose=0)
59 print(f"Accuracy on test data: {scores[1] * 100:.2f}%")
60
61 # Збереження моделі
62 model.save('my_model.h5')

```

Пояснення:

- Імпорт бібліотек забезпечує доступ до інструментів для роботи з нейронними мережами.
- Нормалізація даних масштабує значення пікселів у діапазон [0, 1].
- Згорткові шари аналізують характеристики зображень.

- Dropout мінімізує ризик перенавчання.
- `model.fit` навчає модель на основі тренувального набору.

3.2. Код розпізнавання об'єктів

```
1 import numpy as np
2 from keras.models import load_model
3 from keras.datasets import cifar10
4 import matplotlib.pyplot as plt
5
6 # Завантаження моделі
7 model = load_model('my_model.h5')
8
9 # Завантаження даних для перевірки
10 (_, _), (X_test, y_test) = cifar10.load_data()
11
12 # Нормалізація даних
13 X_test = X_test.astype('float32') / 255
14
15 # Словник класів CIFAR-10
16 class_labels = ['Airplane', 'Automobile', 'Bird', 'Cat', 'Deer',
17                 'Dog', 'Frog', 'Horse', 'Ship', 'Truck']
18
19 # Вибір випадкового зображення для тесту
20 idx = np.random.randint(0, X_test.shape[0])
21 test_image = X_test[idx]
22 test_label = y_test[idx][0]
23
24 # Підготовка зображення для моделі
25 test_image_input = np.expand_dims(test_image, axis=0)
26
27 # Прогноз моделі
28 predictions = model.predict(test_image_input)
29 predicted_class = np.argmax(predictions)
30
31 # Вивід результатів
32 print(f"True class: {class_labels[test_label]}")
33 print(f"Predicted class: {class_labels[predicted_class]}")
34
35 # Вивід зображення
36 plt.imshow(test_image)
```



```
37 plt.title(f"True: {class_labels[test_label]},  
38           Predicted: {class_labels[predicted_class]}")  
39 plt.axis('off')  
40 plt.show()
```

Пояснення:

- Завантажується збережена модель ('my_model.h5').
- Випадково обирається зображення для тестування.
- 'model.predict' визначає ймовірності класів, а 'np.argmax' визначає клас з найвищою ймовірністю.

4. Результати

4.1. Точність моделі

Точність моделі на тестових даних становить **75.60%**. Це свідчить про високу якість навчання моделі на наборі даних CIFAR-10.

4.2. Скріншот результату

На рисунку 4.2.1 представлено зображення, що демонструє результат розпізнавання моделі. Модель правильно визначила об'єкт як **Truck**.

True class: Truck
Predicted class: Truck

True: Truck, Predicted: Truck



Рисунок 4.2.1 – Результат розпізнавання зображення.

Опис скріншота: Зображення представляє вантажівку з набору CIFAR-10.

Наша модель правильно визначила об'єкт як вантажівку (**Truck**). Підпис до зображення відображає фактичний і передбачений клас, що підтверджує успішність роботи нейронної мережі.

5. Висновки

У ході виконання роботи було:

- Розроблено згорткову нейронну мережу для класифікації зображень.
- Проведено навчання моделі на наборі даних CIFAR-10 із точністю 75.60% на тестовій вибірці.
- Візуалізовано результат роботи моделі на прикладі зображення вантажівки.

Модель демонструє ефективність у задачах класифікації зображень. Надалі можна покращити результати за рахунок збільшення кількості епох, модифікації архітектури або використання складніших підходів для попередньої обробки даних.