Лабораторна робота 9 ІАД

Завдання: Створити власний набір даних для навчання та тренування моделі бінарної класифікації.

1. Створення та підготовка набору даних

Вибір теми

Тема: Класифікація зображень фруктів (яблука, банани, апельсини).

Збір даних

Ми використовуємо публічний набір даних **Fruits 360** або створюємо свій власний (у прикладі використовується URL-джерело готового набору).

```
import tensorflow as tf
import os
import zipfile
from tensorflow.keras.utils import get_file

# Завантаження набору даних
dataset_url = "https://storage.googleapis.com/download.tensorflow.org/example_images/fruit_photos.tgz"
data_dir = tf.keras.utils.get_file("fruit_photos.tgz", dataset_url, extract=True)

# Шлях до розпакованих даних
data_dir = os.path.join(os.path.dirname(data_dir), 'fruit_photos')

print(f"Дані розпаковано в: {data_dir}")
```

Перевірка та підготовка даних

```python
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Налаштування параметрів
batch_size = 32
img_height = 180
img_width = 180

# Завантаження тренувального набору
train_dataset = ImageDataGenerator(rescale=1.0/255).flow_from_directory(
    directory=os.path.join(data_dir, 'train'),
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='categorical'
)

# Завантаження тестового набору
test_dataset = ImageDataGenerator(rescale=1.0/255).flow_from_directory(
    directory=os.path.join(data_dir, 'test'),
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='categorical'
)
```

Реалізація моделі CNN

```python
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

# Побудова моделі
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(img_height, img_width, 3)),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(train_dataset.num_classes, activation='softmax')
])

# Компіляція моделі
model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

model.summary()
```

Тренування та тестування моделі

Навчання моделі

```python
# Тренування моделі
history = model.fit(
    train_dataset,
    epochs=10,
    validation_data=test_dataset
)
```

Візуалізація результатів

```python
import matplotlib.pyplot as plt

# Графік точності
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Графік точності')
plt.xlabel('Епохи')
plt.ylabel('Точність')
plt.legend()
plt.show()

# Графік втрат
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Графік втрат')
plt.xlabel('Епохи')
plt.ylabel('Втрати')
plt.legend()
plt.show()
```

Оцінка результатів

Оцінка точності

```python
# Оцінка моделі на тестовому наборі
test_loss, test_accuracy = model.evaluate(test_dataset)
print(f"Точність на тестових даних: {test_accuracy * 100:.2f}%")
```

Прогнозування

```python
import numpy as np
from tensorflow.keras.utils import load_img, img_to_array

# Прогноз для нового зображення
img_path = 'path_to_image.jpg'  # Змініть на шлях до вашого зображення
img = load_img(img_path, target_size=(img_height, img_width))
img_array = img_to_array(img) / 255.0
img_array = np.expand_dims(img_array, axis=0)

# Передбачення класу
predictions = model.predict(img_array)
predicted_class = np.argmax(predictions)
class_names = list(train_dataset.class_indices.keys())
print(f"Передбачений клас: {class_names[predicted_class]}")
```

Метрики класифікації

```python
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np

# Отримання реальних та передбачених міток
y_true = test_dataset.classes
y_pred = np.argmax(model.predict(test_dataset), axis=1)

# Звіт класифікації
print(classification_report(y_true, y_pred, target_names=test_dataset.class_indices.keys()))
```

Висновки

- **Ефективність моделі:** Показники точності залежать від якості та кількості даних.
- **Навчання:** Модель показала стабільне навчання без перенавчання (як видно з графіків).
- **Результати:** Модель здатна передбачати класи з високою точністю на тестових даних.