Штучний Інтелект в задачах обробки зображень, Лабораторна робота №6.

Виконав: студент групи ІП-13, Кисельов Микита Євгенович.

Перевірив: Нікітін Валерій Андрійович.

17.02.2024

Тема роботи: Реалізація архітектури AlexNet CNN за допомогою TensorFlow і Keras.

Мета роботи: отримати навички реалізації архітектури AlexNet CNN з використанням бібліотек TensorFlow та Keras.

Завдання на лабораторну роботу:

1. Реалізувати засобами TensorFlow та Keras AlexNet;

2. Отримати оцінку точності навченої мережі.

1. Реалізувати засобами TensorFlow та Keras AlexNet;

1.1. Підготовка датасету

In [ ]:
```
batch_size = 128
epochs_count = 100
test_size = 1000
val_size = min(50_000, batch_size * epochs_count)
checkpoint_filepath = "best_model.h5"
```

In [ ]:
```
from tensorflow.keras import datasets
from tensorflow import data


(train_images, train_labels), (test_images, test_labels) = datasets.cifar10.load_data()
test_images, test_labels = test_images[:test_size], test_labels[:test_size]
validation_images, validation_labels = train_images[:val_size], train_labels[:val_size]
train_images, train_labels = train_images[val_size:2*val_size], train_labels[val_size:2*val_size]

train_ds = data.Dataset.from_tensor_slices((train_images, train_labels))
test_ds = data.Dataset.from_tensor_slices((test_images, test_labels))
validation_ds = data.Dataset.from_tensor_slices((validation_images, validation_labels))
```

In [ ]:
```
import matplotlib.pyplot as plt
from numpy import argmax


CLASS_NAMES = [
    "airplane", "automobile", "bird", "cat",
    "deer", "dog", "frog", "horse", "ship", "truck",
]

plt.figure(figsize=(20, 20))
for i, (image, label) in enumerate(train_ds.take(5)):
    ax = plt.subplot(5, 5, i+1)
    plt.imshow(image)
    plt.title(CLASS_NAMES[argmax(label)])
```

```
    plt.axis("off")

plt.show()
```

In [ ]:
```
from tensorflow import image as tf_image


def process_images(image, label):
    image = tf_image.per_image_standardization(image)
    image = tf_image.resize(image, (227, 227))
    return image, label


train_ds_size = data.experimental.cardinality(train_ds).numpy()
test_ds_size = data.experimental.cardinality(test_ds).numpy()
validation_ds_size = data.experimental.cardinality(validation_ds).numpy()

print(f"{train_ds_size = }")
print(f"{test_ds_size = }")
print(f"{validation_ds_size = }")

train_ds = (train_ds
            .map(process_images)
            .shuffle(buffer_size=train_ds_size)
            .batch(batch_size=batch_size, drop_remainder=True))

test_ds = (test_ds
            .map(process_images)
            .shuffle(buffer_size=test_ds_size)
            .batch(batch_size=batch_size, drop_remainder=True))

validation_ds = (validation_ds
            .map(process_images)
            .shuffle(buffer_size=validation_ds_size)
            .batch(batch_size=batch_size, drop_remainder=True))
```

1.2. Побудова та компіляція моделі

In [ ]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, BatchNormalization, MaxPooling2D, Flatten, Dense, Dropout


model = Sequential([
    Conv2D(96, (11, 11), (4, 4), activation="relu", input_shape=(227, 227, 3)),
    BatchNormalization(),
    MaxPooling2D((3, 3), (2, 2)),

    Conv2D(256, (5, 5), (1, 1), activation="relu", input_shape=(227, 227, 3), padding="same"),
    BatchNormalization(),
    MaxPooling2D((3, 3), (2, 2)),

    Conv2D(384, (3, 3), (1, 1), activation="relu", input_shape=(227, 227, 3), padding="same"),
    BatchNormalization(),

    Conv2D(384, (3, 3), (1, 1), activation="relu", input_shape=(227, 227, 3), padding="same"),
    BatchNormalization(),

    Conv2D(256, (3, 3), (1, 1), activation="relu", input_shape=(227, 227, 3), padding="same"),
    BatchNormalization(),
    MaxPooling2D((3, 3), (2, 2)),

    Flatten(),

    Dense(4096, activation="relu"),
    Dropout(.5),

    Dense(4096, activation="relu"),
    Dropout(.5),

    Dense(10, activation="softmax")
])
```

In [ ]:
```
from os import path, curdir


run_index = 1
run_logdir = path.join(curdir, "my_cifar10_logs", "run_{:03d}".format(run_index))

def get_run_logdir():
  import time
  run_id = time.strftime("run_%Y_%m_%d-%H_%M_%S")
  return path.join(run_logdir, run_id)
```

In [ ]:
```
from tensorflow.keras import callbacks


run_logdir = get_run_logdir()
tensorboard_cb = callbacks.TensorBoard(run_logdir)
```

In [ ]:
```
from tensorflow import optimizers


model.compile(
    loss="sparse_categorical_crossentropy",
    optimizer=optimizers.SGD(learning_rate=.001),
    metrics=["accuracy"]
)
model.summary()
```

1.3. Тренування моделі

In [ ]:
```
from tensorflow.keras.callbacks import ModelCheckpoint


checkpoint_callback = ModelCheckpoint(
    filepath=checkpoint_filepath,
    save_best_only=True,
    monitor="val_loss",
```

```
    mode="min",
    verbose=True
)

model.fit(
    train_ds,
    epochs=epochs_count,
    validation_data=validation_ds,
    validation_freq=1,
    callbacks=[tensorboard_cb, checkpoint_callback]
)
```

1.3. Візуалізація моделі

In [ ]:
```
# !tensorboard --logdir logs
```

In [ ]:
```
from tensorflow.keras.models import load_model


loaded_model = load_model(checkpoint_filepath)
```

In [ ]:
```
from os import path, makedirs
from tensorflow.keras.utils import plot_model


output_dir = "./tmp"
makedirs(output_dir, exist_ok=True)
img_file = path.join(output_dir, checkpoint_filepath + ".jpg")
plot_model(loaded_model, to_file=img_file, show_shapes=True)
```

In [ ]:
```
from visualkeras import layered_view


layered_view(loaded_model)
```

2. Отримати оцінку точності навченої мережі.

In [ ]:
```
loss, accuracy = loaded_model.evaluate(test_ds)
```

Висновок

В ході виконання даної лабораторної роботи я реалізував архітектуру AlexNet CNN з використанням бібліотек TensorFlow та Keras. Під час побудови моделі були використані згорткові шари, пакетна нормалізація, шари пулінгу, повнозв`язні шари та шари dropout для запобігання перенавчання. Модель була скомпільована з використанням оптимізатора SGD та функції втрати категоріальної крос-ентропії.

Після тренування моделі протягом ... епох було отримано значення точності приблизно ... на тестовому наборі даних.

Таким чином, лабораторна робота дозволила отримати практичні навички з реалізації та оцінки ефективності нейронних мереж в обробці зображень за допомогою штучного інтелекту.

In [ ]:
```
```