

```

from google.colab import drive
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Activation, Conv2D, MaxPooling2D, Flatten
from keras.optimizers import Adam
from sklearn.model_selection import train_test_split
from keras.datasets import mnist, cifar10
from keras.utils import to_categorical
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix
from PIL import Image

drive.mount('/content/drive')

# Załaduj dane mnist
(train_images_mnist, train_labels_mnist), (test_images_mnist, test_labels_mnist) = mnist.load_data()

# Załaduj dane cifar10
(train_images_cifar10, train_labels_cifar10), (test_images_cifar10, test_labels_cifar10) = cifar10.load_data()

class_names = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

# Wyświetl pierwsze 25 obrazków z danych cifar10
plt.figure(figsize=[10, 10])
for i in range(25):
    plt.subplot(5, 5, i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(train_images_cifar10[i], cmap=plt.cm.binary)
    plt.xlabel(class_names[train_labels_cifar10[i][0]])
plt.show()

print(train_images_cifar10.shape)
print(test_images_cifar10.shape)
print(train_labels_cifar10)
print(test_labels_cifar10)

# Przygotuj dane cifar10
x_train_cifar10 = train_images_cifar10.reshape((50000, 32, 32, 3))
x_test_cifar10 = test_images_cifar10.reshape((10000, 32, 32, 3))
x_train_cifar10 = x_train_cifar10.astype('float32') / 255
x_test_cifar10 = x_test_cifar10.astype('float32') / 255

y_train_cifar10 = to_categorical(train_labels_cifar10)
y_test_cifar10 = to_categorical(test_labels_cifar10)

# Zdefiniuj model
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(10, activation='softmax'))

# Kompiluj model
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])

# Trenuj model
history = model.fit(x_train_cifar10, y_train_cifar10, epochs=10, batch_size=40, verbose=0, validation_data=(x_test_cifar10, y_test_cifar10))

# Ewaluacja na danych treningowych
y_pred_train = model.predict(x_train_cifar10)
y_pred_train_rounded = np.argmax(y_pred_train, axis=1)
accuracy_train = accuracy_score(np.argmax(y_train_cifar10, axis=1), y_pred_train_rounded)
precision_train = precision_score(np.argmax(y_train_cifar10, axis=1), y_pred_train_rounded, average='macro')
recall_train = recall_score(np.argmax(y_train_cifar10, axis=1), y_pred_train_rounded, average='macro')
conf_matrix_train = confusion_matrix(np.argmax(y_train_cifar10, axis=1), y_pred_train_rounded)

print("Accuracy on training data:", accuracy_train)
print("Precision on training data:", precision_train)

```

```
print("Recall on training data:", recall_train)
print("Confusion Matrix on training data:\n", conf_matrix_train)

# Ewaluacja na danych testowych
y_pred_test = model.predict(x_test_cifar10)
y_pred_test_rounded = np.argmax(y_pred_test, axis=1)
accuracy_test = accuracy_score(np.argmax(y_test_cifar10, axis=1), y_pred_test_rounded)
precision_test = precision_score(np.argmax(y_test_cifar10, axis=1), y_pred_test_rounded, average='macro')
recall_test = recall_score(np.argmax(y_test_cifar10, axis=1), y_pred_test_rounded, average='macro')
conf_matrix_test = confusion_matrix(np.argmax(y_test_cifar10, axis=1), y_pred_test_rounded)

print("Accuracy on test data:", accuracy_test)
print("Precision on test data:", precision_test)
print("Recall on test data:", recall_test)
print("Confusion Matrix on test data:\n", conf_matrix_test)

# Przygotuj nowy obrazek do testowania
new_images = []
new_labels = []
new_labels.append(2)
image = Image.open('/content/drive/MyDrive/obrazek.png')
array = np.array(image)
new_images.append(array)

new_images = np.array(new_images)
new_labels = np.array(new_labels)

new_test = new_images.reshape((1, 32, 32, 3))
new_test = new_test.astype('float32') / 255
new_y_test = to_categorical(new_labels)

img = new_test[0]
plt.imshow(img, cmap='Greys')
plt.show()

# Ewaluacja na nowym obrazku
predict_new = model.predict(new_test)
result_new = np.argmax(predict_new, axis=1)

errors = 0
a = len(new_y_test)
b = len(new_y_test[0])
if new_y_test[0][result_new[0]] != 1:
    errors += 1

pred_new = np.round(model.predict(new_test))
print("Number of errors on the new test image:", errors, " out of ", len(new_y_test))
```

Drive already mounted at /content/drive; to attempt to forcibly remount, call `drive.mount("/content/drive", force_remount=True)`.



frog



truck



truck



deer



automobile



automobile



bird



horse



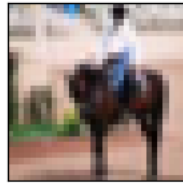
ship



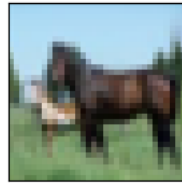
cat



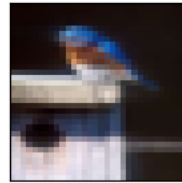
deer



horse



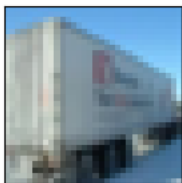
horse



bird



truck



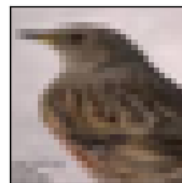
truck



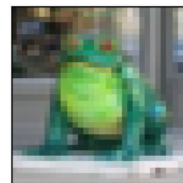
truck



cat



bird



frog



deer



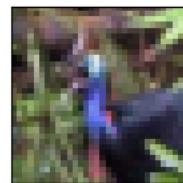
cat



frog



frog



bird

```
(50000, 32, 32, 3)
```

```
(10000, 32, 32, 3)
```

```
[[6]
```

```
[9]
```

```
[9]
```

```
...
```

```
[9]
```

```
[1]
```

```
[1]]
```

```
[[31
```