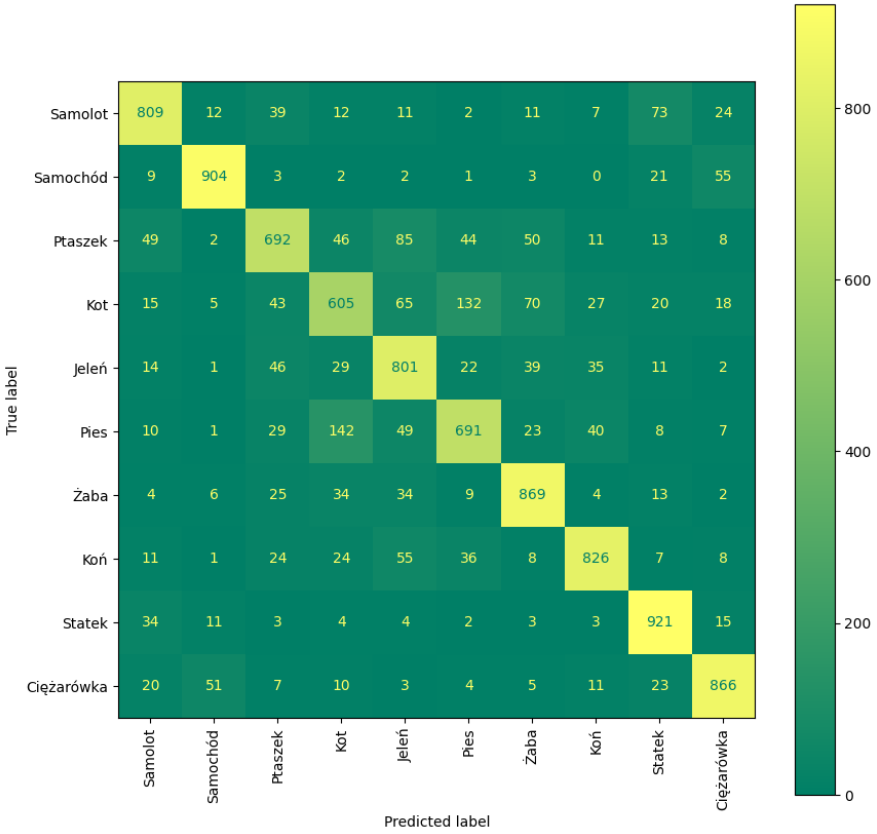
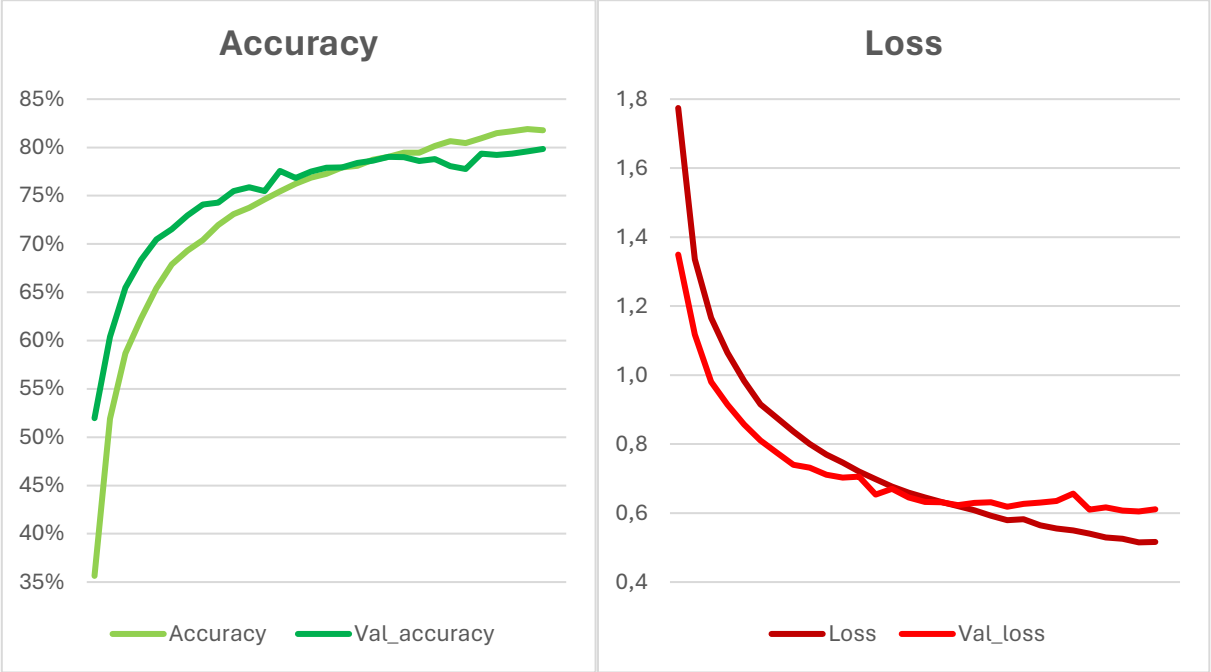
Macierz Konfuzji



Eksport sieci

Warstwa (typ)	Wymiar wyjściowy	Liczba parametrów
InputLayer	(None, 32, 32, 3)	0
Conv2D	(None, 32, 32, 32)	896
MaxPooling2D	(None, 16, 16, 32)	0
Dropout	(None, 16, 16, 32)	0
Conv2D	(None, 16, 16, 64)	18,496
MaxPooling2D	(None, 8, 8, 64)	0
Dropout	(None, 8, 8, 64)	0
Conv2D	(None, 8, 8, 128)	73,856
MaxPooling2D	(None, 4, 4, 128)	0
Dropout	(None, 4, 4, 128)	0
Flatten	(None, 2048)	0
Dense	(None, 512)	1,049,088
Dropout	(None, 512)	0
Dense	(None, 10)	5,13
SUMA		**1,147,466**

Historia Trenowania



Epoch	Loss	Accuracy	Val_loss	Val_accuracy
1	1.7743	0.3564	1.3493	0.5197
2	1.3360	0.5190	1.1172	0.6041
3	1.1657	0.5868	0.9806	0.6547
4	1.0646	0.6229	0.9142	0.6836
5	0.9830	0.6543	0.8569	0.7049
6	0.9157	0.6789	0.8102	0.7152
7	0.8766	0.6932	0.7756	0.7297
8	0.8369	0.7042	0.7398	0.7407
9	0.8007	0.7198	0.7313	0.7427
10	0.7698	0.7308	0.7109	0.7549
11	0.7465	0.7374	0.7030	0.7589
12	0.7207	0.7462	0.7060	0.7549
13	0.6986	0.7544	0.6532	0.7756
14	0.6772	0.7625	0.6704	0.7683
15	0.6593	0.7688	0.6449	0.7751
16	0.6449	0.7726	0.6325	0.7791
17	0.6327	0.7794	0.6315	0.7794
18	0.6205	0.7811	0.6227	0.7840
19	0.6082	0.7874	0.6295	0.7863
20	0.5926	0.7901	0.6317	0.7901
21	0.5798	0.7946	0.6188	0.7900
22	0.5826	0.7945	0.6264	0.7859
23	0.5647	0.8016	0.6307	0.7878
24	0.5557	0.8065	0.6352	0.7806
25	0.5500	0.8046	0.6560	0.7775
26	0.5403	0.8096	0.6098	0.7935
27	0.5295	0.8147	0.6170	0.7921
28	0.5255	0.8169	0.6076	0.7934
29	0.5151	0.8190	0.6047	0.7960
30	0.5160	0.8177	**0.6113**	**0.7984**

Końcowa precyzja dla danych testowych: loss: **0.61**, accuracy: **79%**

Kod Źródłowy:

```
# Importuj biblioteki
import numpy as np
import keras
import matplotlib.pyplot as plt
import pandas as pd
from keras.models import Model
from keras.layers import Input, Dense, Flatten, Conv2D, MaxPooling2D, Dropout
from keras.utils import plot_model
from keras.datasets import cifar10
from sklearn.metrics import ConfusionMatrixDisplay, confusion_matrix

# Definicja etykiet klas
labels = ["Samolot", "Samochód", "Ptaszek", "Kot", "Jeleń", "Pies", "Żaba", "Koń",
"Statek", "Ciężarówka"]

# Załaduj dane
(x_train, y_train), (x_test, y_test) = cifar10.load_data()

# Podgląd pierwszego obrazka w zbiorze treningowym
print("Etykieta: ", labels[y_train[0][0]])
plt.imshow(x_train[0])
plt.show()

# Przetwórz dane
x_train = x_train.astype('float32') / 255
x_test = x_test.astype('float32') / 255

# Konwertuj etykiety na kategorie
num_classes = 10
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)

# Zdefiniuj model za pomocą Functional API
input_shape = (32, 32, 3)
inputs = Input(shape=input_shape)

x = Conv2D(32, kernel_size=(3, 3), activation='relu', padding='same',
kernel_initializer='he_normal')(inputs)
x = MaxPooling2D(pool_size=(2, 2))(x)
x = Dropout(0.25)(x)

x = Conv2D(64, kernel_size=(3, 3), activation='relu', padding='same',
kernel_initializer='he_normal')(x)
x = MaxPooling2D(pool_size=(2, 2))(x)
x = Dropout(0.25)(x)

x = Conv2D(128, kernel_size=(3, 3), activation='relu', padding='same',
kernel_initializer='he_normal')(x)
x = MaxPooling2D(pool_size=(2, 2))(x)
x = Dropout(0.25)(x)

x = Flatten()(x)
x = Dense(512, activation='relu', kernel_initializer='he_normal')(x)
x = Dropout(0.5)(x)
outputs = Dense(num_classes, activation='softmax')(x)

model = Model(inputs=inputs, outputs=outputs)

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

# Trenuj model
history = model.fit(x_train, y_train, batch_size=64, epochs=30, verbose=1,
validation_data=(x_test, y_test))
```

```
# Tworzenie wykresu z historią
df = pd.DataFrame(history.history)
ax = df.plot()
# Zapisywanie wykresu do pliku
fig = ax.get_figure()
fig.savefig('history_plot.png')

# Ocena modelu
loss, accuracy = model.evaluate(x_test, y_test, verbose=1)
print('Test loss:', loss)
print('Test accuracy:', accuracy)

# Wizualizacja modelu
model.summary()
plot_model(model, to_file='model_plot.png', show_shapes=True, show_layer_names=True)

# export modelu CNN
model.save("my_model_cifar10.keras")

# Wyświetlanie przykładów źle sklasyfikowanych
predictions = np.argmax(model.predict(x_test), axis=1)
y_test_flat = np.argmax(y_test, axis=1)
incorrect_indices = np.nonzero(predictions != y_test_flat)[0]

for i in range(5):
    idx = incorrect_indices[i]
    print("Przykład źle sklasyfikowany nr", i+1)
    plt.imshow(x_test[idx])
    plt.xlabel(f"True label: {labels[y_test_flat[idx]]}, Predicted label:
{labels[predictions[idx]]}")
    plt.show()

# Tworzenie macierzy pomyłek
cm = confusion_matrix(y_test_flat, predictions)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=labels)

# Tworzenie wykresu
fig, ax = plt.subplots(figsize=(10, 10))
disp.plot(xticks_rotation='vertical', ax=ax, cmap='summer')

# Zapisywanie wykresu do pliku
plt.savefig('confusion_matrix.png')
plt.show()
```