Kuanysheva

```
# Импорт необходимых библиотек
import numpy as np
import tensorflow as tf
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.metrics import classification_report, ConfusionMatrixDisplay
import matplotlib.pyplot as plt
# Загрузка и нормализация данных
train_data = np.load('C:/Users/Larin/Desktop/Asem/train_small.npz')
test_data = np.load('C:/Users/Larin/Desktop/Asem/test_small.npz')
X_train = train_data['data'].astype(np.float32) / 255.0
y_train = train_data['labels']
X_test = test_data['data'].astype(np.float32) / 255.0
y_test = test_data['labels']
# Аугментация данных
datagen = ImageDataGenerator(
rotation_range=20,
width_shift_range=0.2,
height_shift_range=0.2,
zoom_range=0.2,
horizontal_flip=True
)
datagen.fit(X_train)
# Создание улучшенной модели CNN
model = Sequential([
Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
```

```
MaxPooling2D((2, 2)),
Dropout(0.25),
Conv2D(64, (3, 3), activation='relu'),
MaxPooling2D((2, 2)),
Dropout(0.25),
Conv2D(128, (3, 3), activation='relu'),
MaxPooling2D((2, 2)),
Dropout(0.25),
Flatten(),
Dense(256, activation='relu'),
Dropout(0.5),
Dense(10, activation='softmax') # Убедитесь, что у вас 10 классов
])
# Компиляция модели
model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy',
metrics=['accuracy'])
# Обучение модели с использованием аугментации данных
history = model.fit(datagen.flow(X_train, y_train, batch_size=32),
epochs=50, validation_data=(X_test, y_test))
# Оценка точности
loss, accuracy = model.evaluate(X_test, y_test)
print(f"Точность улучшенной модели CNN: {accuracy:.2f}")
# Предсказание на тестовой выборке
y_pred = model.predict(X_test)
y_pred_classes = np.argmax(y_pred, axis=1)
# Отчет классификации
print("Отчет классификации:")
print(classification_report(y_test, y_pred_classes, zero_division=0))
# Построение матрицы ошибок
```

```
ConfusionMatrixDisplay.from_predictions(y_test, y_pred_classes, cmap="Blues")
plt.title("Confusion Matrix")
plt.show()
```