

```
from google.colab import drive
drive.mount('/content/drive')
```

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

```
from google.colab import drive

import os
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

import cv2

from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import EarlyStopping
```

```
path = '/content/drive/MyDrive/dataset_grao_arroz/'
classes = ['graos_inteiros', 'graos_quebrados']
img_size = 64
```

```
X = []
y = []

subpastas = ['train', 'test']

for idx, classe in enumerate(classes):
    total_classe = 0
    for sub in subpastas:
        pasta = os.path.join(path, classe, sub)
        print(f"Lendo: {pasta}")
        if not os.path.exists(pasta):
            print(f"Pasta não encontrada: {pasta}")
            continue
        arquivos = os.listdir(pasta)
        print(f" - Encontrados {len(arquivos)} arquivos em {sub}")
        for file in arquivos:
            img_path = os.path.join(pasta, file)
            img = cv2.imread(img_path)
            if img is None:
                print(f"Erro ao ler imagem: {img_path}")
                continue
            img = cv2.resize(img, (img_size, img_size))
            X.append(img)
            y.append(idx)
            total_classe += 1
        print(f"Total de imagens carregadas para a classe '{classe}': {total_classe}")

print(f"\nTotal geral de imagens carregadas: {len(X)}")
```

↗ Lendo: /content/drive/MyDrive/dataset_grao_arroz/graos_inteiros/train
 - Encontrados 96 arquivos em train
 Lendo: /content/drive/MyDrive/dataset_grao_arroz/graos_inteiros/test
 - Encontrados 22 arquivos em test
 Total de imagens carregadas para a classe 'graos_inteiros': 118
 Lendo: /content/drive/MyDrive/dataset_grao_arroz/graos_quebrados/train
 - Encontrados 58 arquivos em train
 Lendo: /content/drive/MyDrive/dataset_grao_arroz/graos_quebrados/test
 - Encontrados 16 arquivos em test
 Total de imagens carregadas para a classe 'graos_quebrados': 74
 Total geral de imagens carregadas: 192

```
X = np.array(X) / 255.0
y = np.array(y)
```

```
x_train, x_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3, stratify=y, random_state=42)
x_val, x_test, y_val, y_test = train_test_split(x_temp, y_temp, test_size=0.5, stratify=y_temp, random_state=42)

print(f'Treino: {len(x_train)}, Validação: {len(x_val)}, Teste: {len(x_test)}')
```

```
datagen = ImageDataGenerator(
    rotation_range=15,
    width_shift_range=0.1,
    height_shift_range=0.1,
    zoom_range=0.1,
    horizontal_flip=True
)
datagen.fit(x_train)
```

```
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(img_size, img_size, 3)),
    MaxPooling2D(2,2),
    Dropout(0.25),


    Conv2D(64, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Dropout(0.25),

    Flatten(),
    Dense(64, activation='relu'),
    Dropout(0.5),

    Dense(1, activation='sigmoid')
])
```

```
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

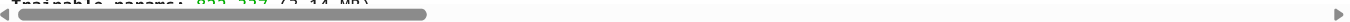
```
model.summary()
```

 /usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape` to super().__init__(activity_regularizer=activity_regularizer, **kwargs)

Model: "sequential_29"

Layer (type)	Output Shape	Param #
conv2d_79 (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d_79 (MaxPooling2D)	(None, 31, 31, 32)	0
dropout_108 (Dropout)	(None, 31, 31, 32)	0
conv2d_80 (Conv2D)	(None, 29, 29, 64)	18,496
max_pooling2d_80 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_109 (Dropout)	(None, 14, 14, 64)	0
flatten_25 (Flatten)	(None, 12544)	0
dense_58 (Dense)	(None, 64)	802,880
dropout_110 (Dropout)	(None, 64)	0
dense_59 (Dense)	(None, 1)	65

Total params: 822,337 (3.14 MB)



```
early_stop = EarlyStopping(monitor='val_accuracy', patience=30, restore_best_weights=True)
```

```
history = model.fit(
    datagen.flow(x_train, y_train, batch_size=32),
    epochs=100,
    validation_data=(x_val, y_val),
    callbacks=[early_stop]
)
```



```

Epoch 20/100
5/5 ————— 3s 270ms/step - accuracy: 0.8197 - loss: 0.3743 - val_accuracy: 1.0000 - val_loss: 0.2496
Epoch 21/100
5/5 ————— 1s 176ms/step - accuracy: 0.7727 - loss: 0.4125 - val_accuracy: 1.0000 - val_loss: 0.2744
Epoch 22/100
5/5 ————— 1s 181ms/step - accuracy: 0.8446 - loss: 0.3988 - val_accuracy: 0.9310 - val_loss: 0.4002
Epoch 23/100
5/5 ————— 1s 223ms/step - accuracy: 0.8730 - loss: 0.2937 - val_accuracy: 1.0000 - val_loss: 0.2495
Epoch 24/100
5/5 ————— 1s 218ms/step - accuracy: 0.8084 - loss: 0.3227 - val_accuracy: 0.8966 - val_loss: 0.3490
Epoch 25/100
5/5 ————— 1s 181ms/step - accuracy: 0.8878 - loss: 0.3466 - val_accuracy: 1.0000 - val_loss: 0.1805
Epoch 26/100
5/5 ————— 1s 179ms/step - accuracy: 0.7927 - loss: 0.3482 - val_accuracy: 0.9655 - val_loss: 0.2521
Epoch 27/100
5/5 ————— 1s 214ms/step - accuracy: 0.9027 - loss: 0.3093 - val_accuracy: 0.7931 - val_loss: 0.3544
Epoch 28/100
5/5 ————— 1s 170ms/step - accuracy: 0.9038 - loss: 0.2768 - val_accuracy: 1.0000 - val_loss: 0.1950
Epoch 29/100
5/5 ————— 1s 221ms/step - accuracy: 0.8236 - loss: 0.3872 - val_accuracy: 0.9310 - val_loss: 0.2929
Epoch 30/100
5/5 ————— 2s 310ms/step - accuracy: 0.8602 - loss: 0.3347 - val_accuracy: 0.9655 - val_loss: 0.1704
Epoch 31/100
5/5 ————— 2s 325ms/step - accuracy: 0.9138 - loss: 0.2401 - val_accuracy: 0.9310 - val_loss: 0.2527
Epoch 32/100
5/5 ————— 2s 180ms/step - accuracy: 0.9252 - loss: 0.2225 - val_accuracy: 0.9655 - val_loss: 0.1341
Epoch 33/100
5/5 ————— 1s 175ms/step - accuracy: 0.9242 - loss: 0.1939 - val_accuracy: 0.9655 - val_loss: 0.1336
Epoch 34/100
5/5 ————— 1s 176ms/step - accuracy: 0.8820 - loss: 0.2673 - val_accuracy: 0.9655 - val_loss: 0.1949
Epoch 35/100
5/5 ————— 1s 176ms/step - accuracy: 0.9227 - loss: 0.2094 - val_accuracy: 1.0000 - val_loss: 0.0955
Epoch 36/100
5/5 ————— 1s 231ms/step - accuracy: 0.8955 - loss: 0.2735 - val_accuracy: 0.9655 - val_loss: 0.1415
Epoch 37/100
5/5 ————— 1s 188ms/step - accuracy: 0.9375 - loss: 0.1965 - val_accuracy: 0.7586 - val_loss: 0.3281
Epoch 38/100
5/5 ————— 1s 186ms/step - accuracy: 0.9217 - loss: 0.1931 - val_accuracy: 1.0000 - val_loss: 0.0761
Epoch 39/100
5/5 ————— 1s 218ms/step - accuracy: 0.9473 - loss: 0.2052 - val_accuracy: 1.0000 - val_loss: 0.0785
Epoch 40/100
5/5 ————— 1s 277ms/step - accuracy: 0.9261 - loss: 0.2063 - val_accuracy: 0.9655 - val_loss: 0.1292
Epoch 41/100
5/5 ————— 2s 303ms/step - accuracy: 0.9544 - loss: 0.1464 - val_accuracy: 0.7586 - val_loss: 0.4177
Epoch 42/100
5/5 ————— 2s 195ms/step - accuracy: 0.9598 - loss: 0.1853 - val_accuracy: 0.7586 - val_loss: 0.3317
Epoch 43/100
5/5 ————— 1s 215ms/step - accuracy: 0.8992 - loss: 0.2658 - val_accuracy: 0.9310 - val_loss: 0.1358
Epoch 44/100
5/5 ————— 1s 232ms/step - accuracy: 0.8564 - loss: 0.2327 - val_accuracy: 0.9655 - val_loss: 0.1375
Epoch 45/100
5/5 ————— 1s 175ms/step - accuracy: 0.9501 - loss: 0.1672 - val_accuracy: 0.9655 - val_loss: 0.1266

```

```

plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'], label='Treino')
plt.plot(history.history['val_accuracy'], label='Validação')
plt.title('Acurácia')
plt.legend()

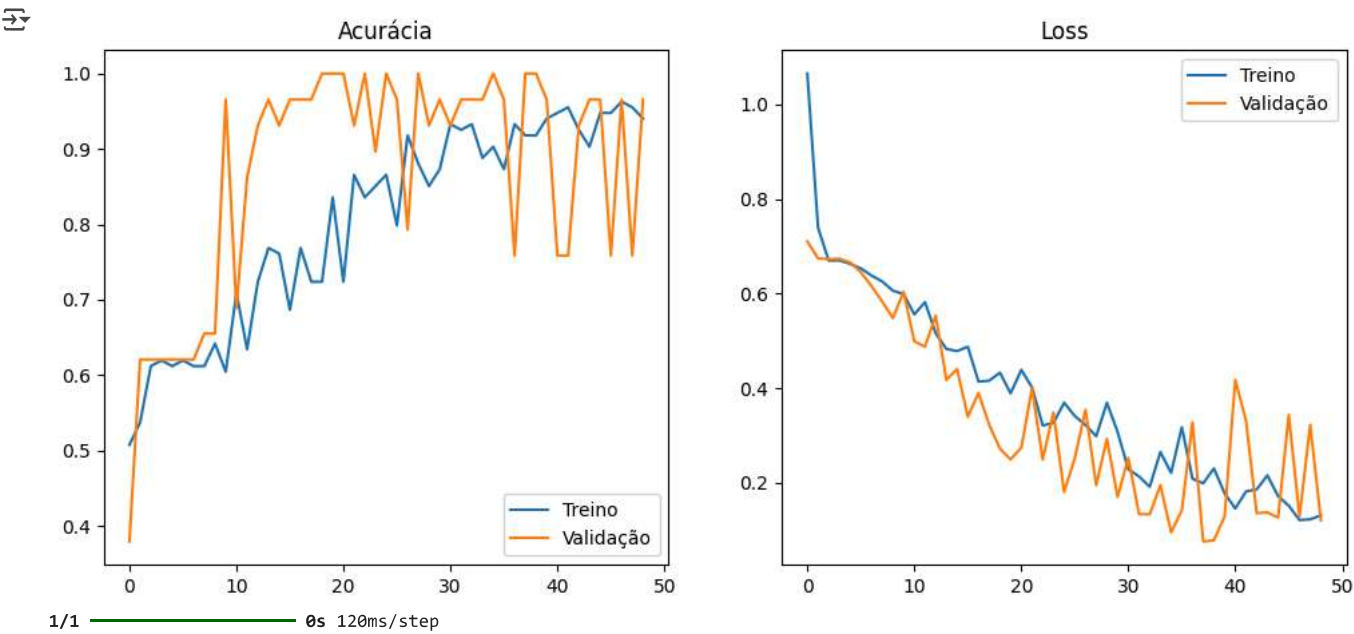
plt.subplot(1,2,2)
plt.plot(history.history['loss'], label='Treino')
plt.plot(history.history['val_loss'], label='Validação')
plt.title('Loss')
plt.legend()
plt.show()

y_pred_prob = model.predict(x_test)
y_pred = (y_pred_prob > 0.5).astype(int)

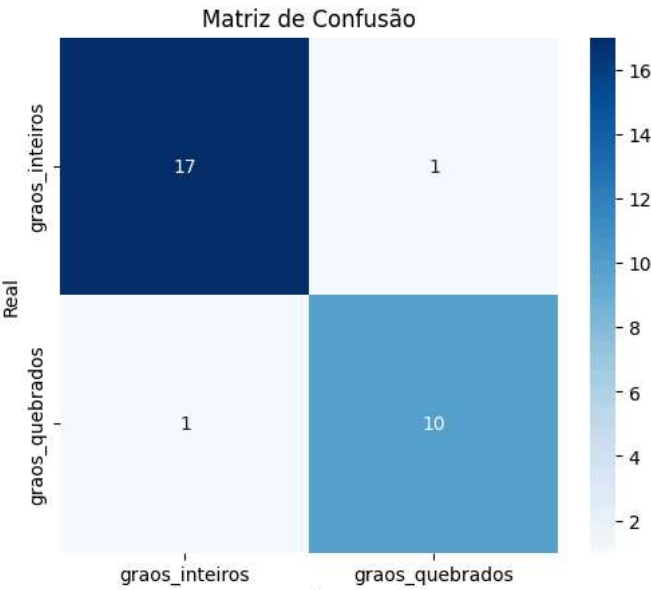
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt="d", cmap='Blues', xticklabels=classes, yticklabels=classes)
plt.title("Matriz de Confusão")
plt.xlabel("Predito")
plt.ylabel("Real")
plt.show()

print(classification_report(y_test, y_pred, target_names=classes))

```



1/1 0s 120ms/step



	precision	recall	f1-score	support
graos_inteiros	0.94	0.94	0.94	18
graos_quebrados	0.91	0.91	0.91	11
accuracy			0.93	29
macro avg	0.93	0.93	0.93	29
weighted avg	0.93	0.93	0.93	29