fhtabxpa9

June 19, 2024

[1]:
```
import os, shutil
train_dir = 'C:/Users/flavi/Desktop/Projeto-20240530/train'
validation_dir = 'C:/Users/flavi/Desktop/Projeto-20240530/validation'
test_dir = 'C:/Users/flavi/Desktop/Projeto-20240530/test'
```

[2]:
```
from keras.utils import image_dataset_from_directory
IMG_SIZE = 150
train_dataset = image_dataset_from_directory(
train_dir,
image_size=(IMG_SIZE, IMG_SIZE),
batch_size=32)
validation_dataset = image_dataset_from_directory(
validation_dir,
image_size=(IMG_SIZE, IMG_SIZE),
batch_size=32)
test_dataset = image_dataset_from_directory(
test_dir,
image_size=(IMG_SIZE, IMG_SIZE),
batch_size=32)
```

```
Found 40000 files belonging to 10 classes.
Found 10000 files belonging to 10 classes.
Found 10000 files belonging to 10 classes.
```

[3]:
```
from tensorflow import keras
from keras import layers
from keras import models
from keras.preprocessing import image

inputs = keras.Input(shape=(IMG_SIZE, IMG_SIZE, 3))
x = layers.Rescaling(1./255)(inputs)

x = layers.Conv2D(filters=32, kernel_size=3, activation="relu",␣
 ↪padding='same')(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(pool_size=2)(x)
```

```
x = layers.Conv2D(filters=64, kernel_size=3, activation="relu",␣
↪padding='same')(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(pool_size=2)(x)

x = layers.Conv2D(filters=128, kernel_size=3, activation="relu",␣
↪padding='same')(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(pool_size=2)(x)

x = layers.Flatten()(x)
x = layers.Dropout(0.5)(x)
x = layers.Dense(512, activation="relu")(x)
outputs = layers.Dense(10, activation="softmax")(x)
model = keras.Model(inputs=inputs, outputs=outputs)
```

[4]:
```
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']
)
```

[5]:
```
from keras.callbacks import ReduceLROnPlateau

reduce_lr = ReduceLROnPlateau(
    monitor='val_loss',
    factor=0.2,
    patience=2,
    min_lr=0.001
)
```

[6]:
```
from keras.callbacks import EarlyStopping

early_stopping = EarlyStopping(
    monitor='val_loss',
    patience=5,
    restore_best_weights=True
)
```

[7]:
```
from keras.callbacks import ModelCheckpoint

model_checkpoint = ModelCheckpoint(
    filepath='C:/Users/flavi/Desktop/projetoClassificaoDeImagens/
↪dl_project_2201707_2211044/ModelosS/ModelS_SGDOptimizer.keras',
    save_best_only=True,
    monitor='val_loss'
)
```

[8]:

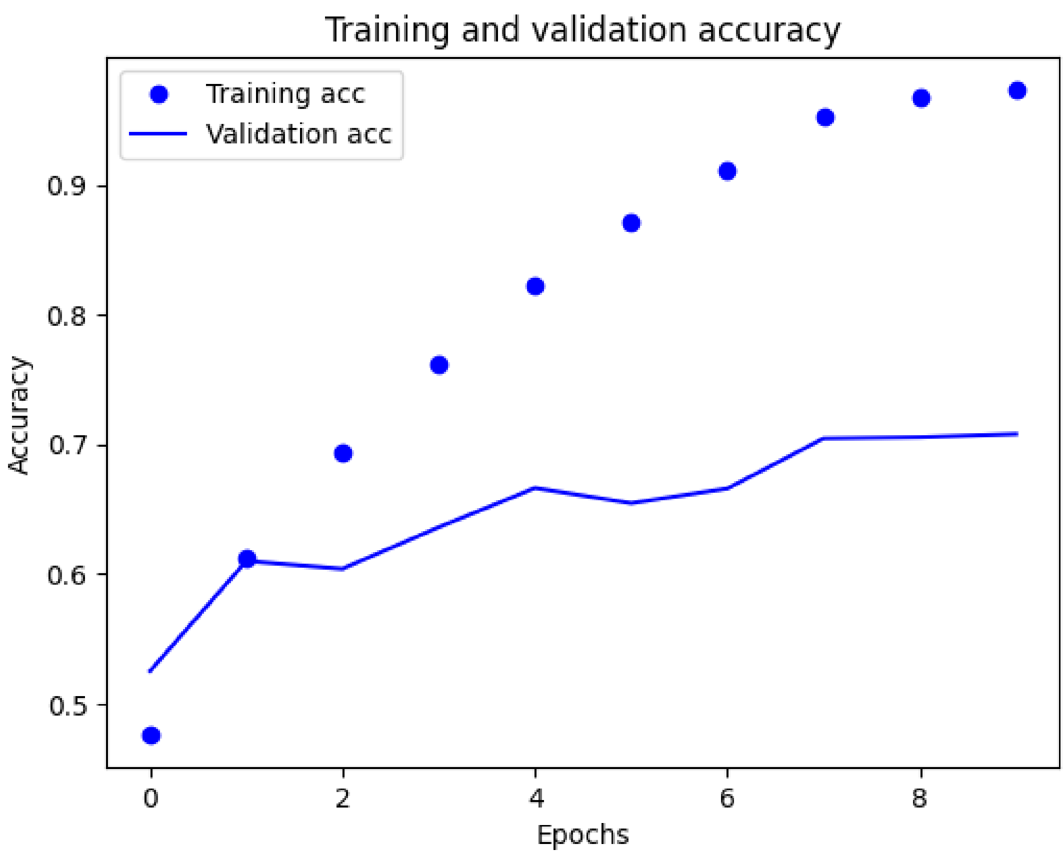
```
callbacks = [reduce_lr, early_stopping, model_checkpoint]
history = model.fit(
    train_dataset,
    epochs=30,
    validation_data=validation_dataset,
    callbacks=callbacks
)
```

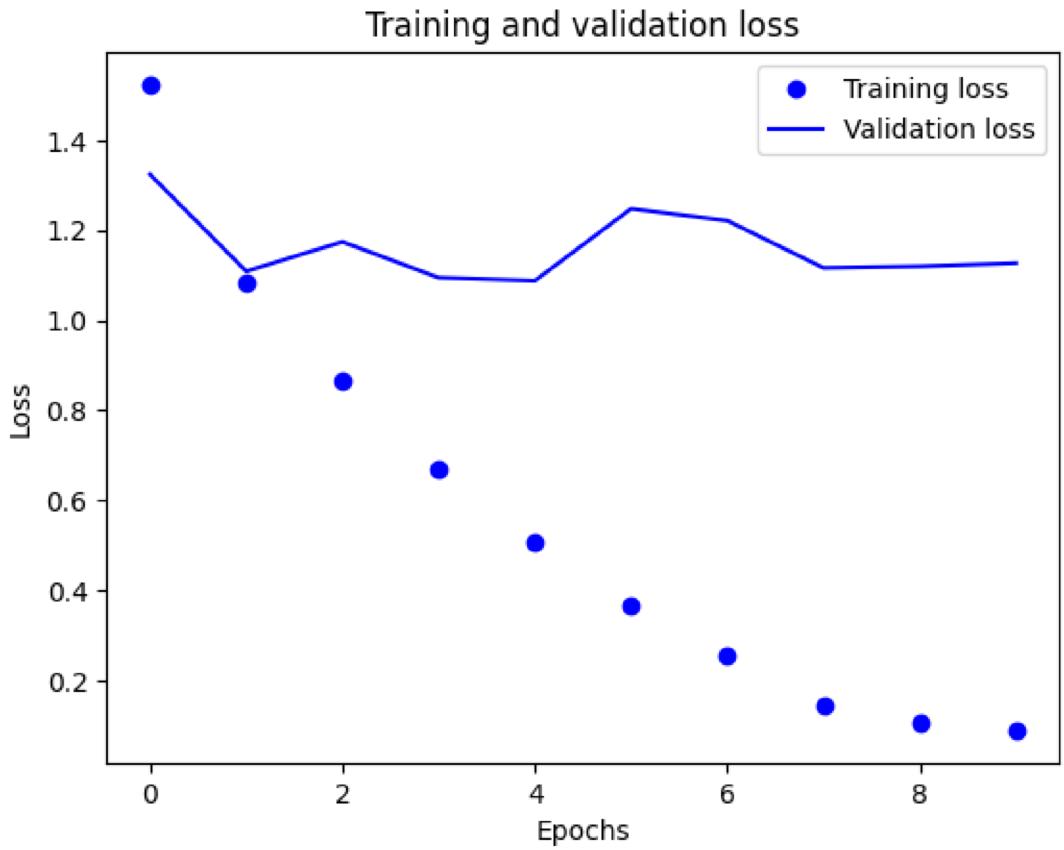
```
Epoch 1/30
1250/1250          1092s
869ms/step - accuracy: 0.4040 - loss: 1.9449 - val_accuracy: 0.5251 - val_loss:
1.3241 - learning_rate: 0.0100
Epoch 2/30
1250/1250          1091s
873ms/step - accuracy: 0.5952 - loss: 1.1380 - val_accuracy: 0.6099 - val_loss:
1.1089 - learning_rate: 0.0100
Epoch 3/30
1250/1250          1131s
905ms/step - accuracy: 0.6770 - loss: 0.9113 - val_accuracy: 0.6040 - val_loss:
1.1744 - learning_rate: 0.0100
Epoch 4/30
1250/1250          1199s
933ms/step - accuracy: 0.7436 - loss: 0.7177 - val_accuracy: 0.6360 - val_loss:
1.0946 - learning_rate: 0.0100
Epoch 5/30
1250/1250          1165s
932ms/step - accuracy: 0.8082 - loss: 0.5436 - val_accuracy: 0.6662 - val_loss:
1.0878 - learning_rate: 0.0100
Epoch 6/30
1250/1250          1149s
919ms/step - accuracy: 0.8600 - loss: 0.3930 - val_accuracy: 0.6546 - val_loss:
1.2478 - learning_rate: 0.0100
Epoch 7/30
1250/1250          906s 725ms/step
- accuracy: 0.9039 - loss: 0.2750 - val_accuracy: 0.6658 - val_loss: 1.2213 -
learning_rate: 0.0100
Epoch 8/30
1250/1250          542s 434ms/step
- accuracy: 0.9402 - loss: 0.1761 - val_accuracy: 0.7045 - val_loss: 1.1164 -
learning_rate: 0.0020
Epoch 9/30
1250/1250          535s 428ms/step
- accuracy: 0.9619 - loss: 0.1190 - val_accuracy: 0.7054 - val_loss: 1.1198 -
learning_rate: 0.0020
Epoch 10/30
1250/1250          552s 442ms/step
- accuracy: 0.9696 - loss: 0.0996 - val_accuracy: 0.7079 - val_loss: 1.1270 -
```

learning_rate: 0.0010

[9]:
```
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'], 'bo', label='Training acc')
plt.plot(history.history['val_accuracy'], 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

plt.plot(history.history['loss'], 'bo', label='Training loss')
plt.plot(history.history['val_loss'], 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```





[10]:
```
val_loss, val_acc = model.evaluate(validation_dataset)
print('Validation Accuracy:', val_acc)
```

```
313/313              21s 67ms/step -
accuracy: 0.6692 - loss: 1.0758
Validation Accuracy: 0.6661999821662903
```

[11]:
```
loss, accuracy = model.evaluate(test_dataset)
print(f"Loss: {loss}, Accuracy: {accuracy}")
```

```
313/313              21s 67ms/step -
accuracy: 0.6583 - loss: 1.0930
Loss: 1.0986255407333374, Accuracy: 0.654699981212616
```

[12]:
```
import numpy as np
from sklearn.metrics import confusion_matrix, classification_report
import seaborn as sns
import matplotlib.pyplot as plt

# Function to evaluate the model and get true and predicted labels
```

```python
def evaluate_model(model, dataset):
    all_labels = []
    all_predictions = []

    for images, labels in dataset:
        predictions = model.predict(images)
        predicted_labels = np.argmax(predictions, axis=1)
        true_labels = labels.numpy()  # Convert to numpy array if not already

        all_labels.extend(true_labels)
        all_predictions.extend(predicted_labels)

    return np.array(all_labels), np.array(all_predictions)

# Get true and predicted labels for the test dataset
true_labels, predicted_labels = evaluate_model(model, test_dataset)

# Compute the confusion matrix
conf_matrix = confusion_matrix(true_labels, predicted_labels)

# Plot the confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=range(10), yticklabels=range(10))
plt.title('Matriz de Confusão')
plt.xlabel('Previsão')
plt.ylabel('Realidade')
plt.show()

# Print classification report
class_names = [str(i) for i in range(10)]  # Define class names based on your dataset
print(classification_report(true_labels, predicted_labels, target_names=class_names))

# Extract precision, recall, and F1-score for each class from classification report
report = classification_report(true_labels, predicted_labels, target_names=class_names, output_dict=True)

metrics = {'precision': [], 'recall': [], 'f1-score': []}
for cls in class_names:
    metrics['precision'].append(report[cls]['precision'])
    metrics['recall'].append(report[cls]['recall'])
    metrics['f1-score'].append(report[cls]['f1-score'])

# Plot precision, recall, and F1-score
```

```python
plt.figure(figsize=(10, 6))
bar_width = 0.2
index = np.arange(len(class_names))

plt.bar(index, metrics['precision'], bar_width, label='Precision')
plt.bar(index + bar_width, metrics['recall'], bar_width, label='Recall')
plt.bar(index + 2*bar_width, metrics['f1-score'], bar_width, label='F1-score')

plt.xlabel('Class')
plt.ylabel('Scores')
plt.title('Precision, Recall e F1-score para cada classe')
plt.xticks(index + bar_width, class_names)
plt.legend()

plt.tight_layout()
plt.show()
```

```
1/1               0s 140ms/step
1/1               0s 64ms/step
1/1               0s 78ms/step
1/1               0s 84ms/step
1/1               0s 73ms/step
1/1               0s 63ms/step
1/1               0s 78ms/step
1/1               0s 74ms/step
1/1               0s 77ms/step
1/1               0s 79ms/step
1/1               0s 68ms/step
1/1               0s 66ms/step
1/1               0s 73ms/step
1/1               0s 71ms/step
1/1               0s 73ms/step
1/1               0s 70ms/step
1/1               0s 71ms/step
1/1               0s 62ms/step
1/1               0s 78ms/step
1/1               0s 82ms/step
1/1               0s 68ms/step
1/1               0s 66ms/step
1/1               0s 78ms/step
1/1               0s 77ms/step
1/1               0s 67ms/step
1/1               0s 78ms/step
1/1               0s 78ms/step
1/1               0s 77ms/step
1/1               0s 78ms/step
1/1               0s 84ms/step
```

```
1/1                0s 74ms/step
1/1                0s 63ms/step
1/1                0s 71ms/step
1/1                0s 71ms/step
1/1                0s 76ms/step
1/1                0s 74ms/step
1/1                0s 73ms/step
1/1                0s 73ms/step
1/1                0s 64ms/step
1/1                0s 64ms/step
1/1                0s 63ms/step
1/1                0s 82ms/step
1/1                0s 82ms/step
1/1                0s 78ms/step
1/1                0s 61ms/step
1/1                0s 73ms/step
1/1                0s 71ms/step
1/1                0s 70ms/step
1/1                0s 68ms/step
1/1                0s 63ms/step
1/1                0s 62ms/step
1/1                0s 78ms/step
1/1                0s 69ms/step
1/1                0s 76ms/step
1/1                0s 64ms/step
1/1                0s 67ms/step
1/1                0s 66ms/step
1/1                0s 75ms/step
1/1                0s 85ms/step
1/1                0s 78ms/step
1/1                0s 67ms/step
1/1                0s 78ms/step
1/1                0s 72ms/step
1/1                0s 72ms/step
1/1                0s 68ms/step
1/1                0s 78ms/step
1/1                0s 87ms/step
1/1                0s 82ms/step
1/1                0s 69ms/step
1/1                0s 78ms/step
1/1                0s 79ms/step
1/1                0s 73ms/step
1/1                0s 76ms/step
1/1                0s 68ms/step
1/1                0s 67ms/step
1/1                0s 75ms/step
1/1                0s 78ms/step
1/1                0s 78ms/step
```

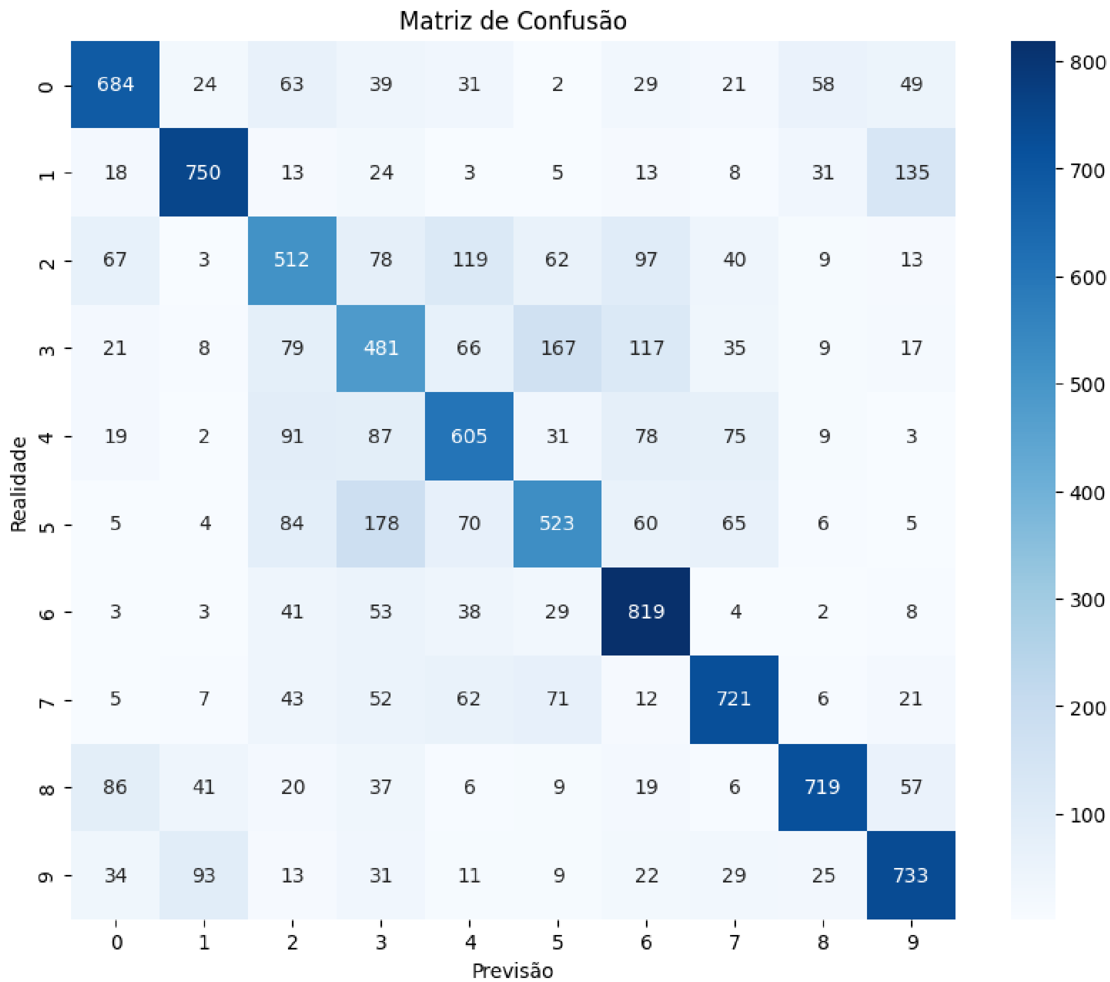
```
1/1              0s 78ms/step
1/1              0s 81ms/step
1/1              0s 76ms/step
1/1              0s 75ms/step
1/1              0s 78ms/step
1/1              0s 64ms/step
1/1              0s 74ms/step
1/1              0s 72ms/step
1/1              0s 78ms/step
1/1              0s 71ms/step
1/1              0s 81ms/step
1/1              0s 71ms/step
1/1              0s 71ms/step
1/1              0s 72ms/step
1/1              0s 78ms/step
1/1              0s 62ms/step
1/1              0s 63ms/step
1/1              0s 82ms/step
1/1              0s 77ms/step
1/1              0s 78ms/step
1/1              0s 72ms/step
1/1              0s 66ms/step
1/1              0s 72ms/step
1/1              0s 77ms/step
1/1              0s 71ms/step
1/1              0s 75ms/step
1/1              0s 74ms/step
1/1              0s 68ms/step
1/1              0s 70ms/step
1/1              0s 72ms/step
1/1              0s 78ms/step
1/1              0s 60ms/step
1/1              0s 71ms/step
1/1              0s 78ms/step
1/1              0s 74ms/step
1/1              0s 72ms/step
1/1              0s 67ms/step
1/1              0s 71ms/step
1/1              0s 80ms/step
1/1              0s 79ms/step
1/1              0s 70ms/step
1/1              0s 68ms/step
1/1              0s 70ms/step
1/1              0s 75ms/step
1/1              0s 79ms/step
1/1              0s 58ms/step
1/1              0s 72ms/step
1/1              0s 79ms/step
```

```
1/1                0s 79ms/step
1/1                0s 70ms/step
1/1                0s 72ms/step
1/1                0s 79ms/step
1/1                0s 73ms/step
1/1                0s 72ms/step
1/1                0s 73ms/step
1/1                0s 80ms/step
1/1                0s 73ms/step
1/1                0s 58ms/step
1/1                0s 73ms/step
1/1                0s 79ms/step
1/1                0s 77ms/step
1/1                0s 65ms/step
1/1                0s 67ms/step
1/1                0s 71ms/step
1/1                0s 77ms/step
1/1                0s 78ms/step
1/1                0s 75ms/step
1/1                0s 71ms/step
1/1                0s 71ms/step
1/1                0s 81ms/step
1/1                0s 74ms/step
1/1                0s 72ms/step
1/1                0s 71ms/step
1/1                0s 83ms/step
1/1                0s 83ms/step
1/1                0s 80ms/step
1/1                0s 71ms/step
1/1                0s 72ms/step
1/1                0s 74ms/step
1/1                0s 79ms/step
1/1                0s 72ms/step
1/1                0s 71ms/step
1/1                0s 70ms/step
1/1                0s 76ms/step
1/1                0s 84ms/step
1/1                0s 71ms/step
1/1                0s 68ms/step
1/1                0s 76ms/step
1/1                0s 81ms/step
1/1                0s 71ms/step
1/1                0s 72ms/step
1/1                0s 72ms/step
1/1                0s 80ms/step
1/1                0s 63ms/step
1/1                0s 71ms/step
1/1                0s 74ms/step
```

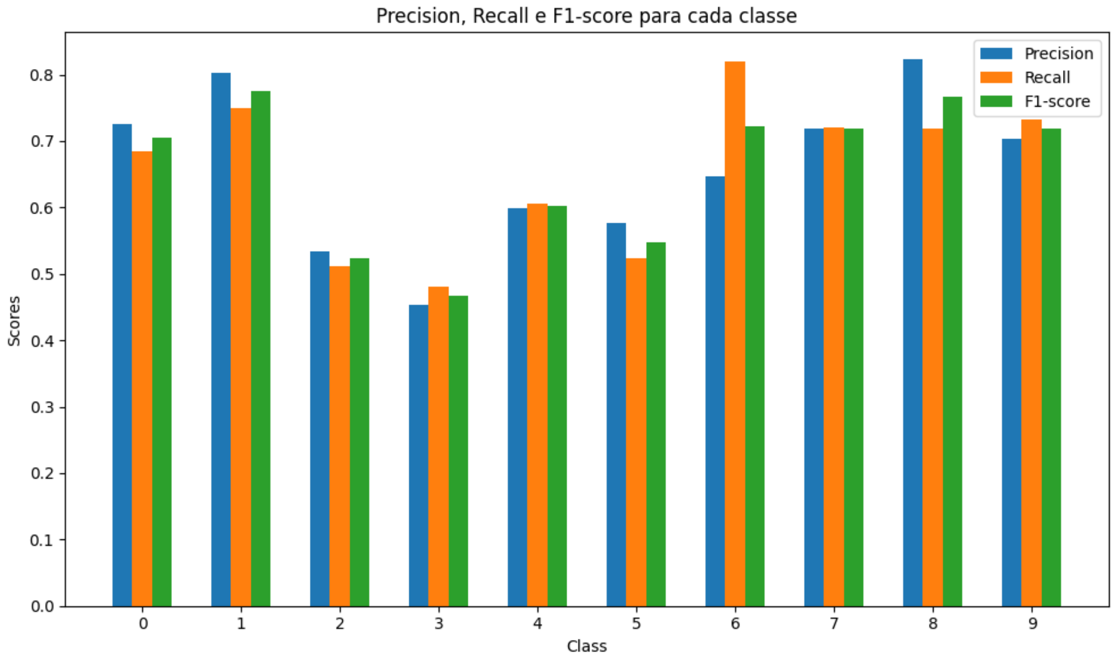
```
1/1            0s 70ms/step
1/1            0s 74ms/step
1/1            0s 70ms/step
1/1            0s 69ms/step
1/1            0s 77ms/step
1/1            0s 72ms/step
1/1            0s 71ms/step
1/1            0s 70ms/step
1/1            0s 83ms/step
1/1            0s 68ms/step
1/1            0s 68ms/step
1/1            0s 77ms/step
1/1            0s 77ms/step
1/1            0s 76ms/step
1/1            0s 79ms/step
1/1            0s 64ms/step
1/1            0s 71ms/step
1/1            0s 77ms/step
1/1            0s 66ms/step
1/1            0s 69ms/step
1/1            0s 80ms/step
1/1            0s 74ms/step
1/1            0s 74ms/step
1/1            0s 71ms/step
1/1            0s 81ms/step
1/1            0s 80ms/step
1/1            0s 74ms/step
1/1            0s 73ms/step
1/1            0s 70ms/step
1/1            0s 81ms/step
1/1            0s 80ms/step
1/1            0s 71ms/step
1/1            0s 71ms/step
1/1            0s 74ms/step
1/1            0s 78ms/step
1/1            0s 64ms/step
1/1            0s 68ms/step
1/1            0s 72ms/step
1/1            0s 77ms/step
1/1            0s 73ms/step
1/1            0s 71ms/step
1/1            0s 68ms/step
1/1            0s 73ms/step
1/1            0s 73ms/step
1/1            0s 78ms/step
1/1            0s 58ms/step
1/1            0s 72ms/step
1/1            0s 75ms/step
```

```
1/1                0s 72ms/step
1/1                0s 67ms/step
1/1                0s 77ms/step
1/1                0s 83ms/step
1/1                0s 72ms/step
1/1                0s 78ms/step
1/1                0s 70ms/step
1/1                0s 70ms/step
1/1                0s 67ms/step
1/1                0s 77ms/step
1/1                0s 75ms/step
1/1                0s 73ms/step
1/1                0s 70ms/step
1/1                0s 70ms/step
1/1                0s 83ms/step
1/1                0s 80ms/step
1/1                0s 72ms/step
1/1                0s 71ms/step
1/1                0s 71ms/step
1/1                0s 78ms/step
1/1                0s 65ms/step
1/1                0s 70ms/step
1/1                0s 82ms/step
1/1                0s 70ms/step
1/1                0s 71ms/step
1/1                0s 71ms/step
1/1                0s 66ms/step
1/1                0s 72ms/step
1/1                0s 77ms/step
1/1                0s 65ms/step
1/1                0s 67ms/step
1/1                0s 73ms/step
1/1                0s 73ms/step
1/1                0s 81ms/step
1/1                0s 79ms/step
1/1                0s 74ms/step
1/1                0s 71ms/step
1/1                0s 76ms/step
1/1                0s 70ms/step
1/1                0s 68ms/step
1/1                0s 77ms/step
1/1                0s 77ms/step
1/1                0s 72ms/step
1/1                0s 70ms/step
1/1                0s 71ms/step
1/1                0s 77ms/step
1/1                0s 77ms/step
1/1                0s 71ms/step
```

```
1/1              0s 68ms/step
1/1              0s 66ms/step
1/1              0s 74ms/step
1/1              0s 72ms/step
1/1              0s 72ms/step
1/1              0s 79ms/step
1/1              0s 73ms/step
1/1              0s 72ms/step
1/1              0s 76ms/step
1/1              0s 77ms/step
1/1              0s 76ms/step
1/1              0s 70ms/step
1/1              0s 70ms/step
1/1              0s 83ms/step
1/1              0s 84ms/step
1/1              0s 75ms/step
1/1              0s 73ms/step
1/1              0s 68ms/step
1/1              0s 79ms/step
1/1              0s 84ms/step
1/1              0s 73ms/step
1/1              0s 59ms/step
1/1              0s 67ms/step
1/1              0s 77ms/step
1/1              0s 75ms/step
1/1              0s 72ms/step
1/1              0s 84ms/step
1/1              0s 83ms/step
1/1              0s 75ms/step
1/1              0s 73ms/step
1/1              0s 73ms/step
1/1              0s 68ms/step
1/1              0s 75ms/step
1/1              0s 67ms/step
1/1              0s 69ms/step
1/1              0s 67ms/step
1/1              0s 68ms/step
1/1              0s 66ms/step
1/1              0s 69ms/step
1/1              0s 75ms/step
1/1              0s 68ms/step
1/1              0s 70ms/step
1/1              0s 98ms/step
```



	precision	recall	f1-score	support
0	0.73	0.68	0.70	1000
1	0.80	0.75	0.78	1000
2	0.53	0.51	0.52	1000
3	0.45	0.48	0.47	1000
4	0.60	0.60	0.60	1000
5	0.58	0.52	0.55	1000
6	0.65	0.82	0.72	1000
7	0.72	0.72	0.72	1000
8	0.82	0.72	0.77	1000
9	0.70	0.73	0.72	1000
accuracy			0.65	10000
macro avg	0.66	0.65	0.65	10000
weighted avg	0.66	0.65	0.65	10000


Precision, Recall e F1-score para cada classe
Precision
Recall
F1-score
Scores
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
1
2
3
4
5
6
7
8
9
Class