Universidade Federal Fluminense

TÓPICOS ESP. SIST. INFORMAÇÃO

Prof.ªLeila Weitzel

3. Treinamento de Redes

Etapas da classificação com Python

6. Treinar o modelo

```
model.fit(X_train, y_train, epochs=100, batch_size=10):
```

- **X_train:**
  - É o conjunto de dados de treinamento (recursos).
  - Contém as entradas para o modelo.
- **y_train:**
  - É o conjunto de rótulos correspondentes aos dados de treinamento.
  - Contém as saídas esperadas para o modelo.
- **epochs:**
  - Número de vezes que o modelo percorre todo o conjunto de treinamento.
  - Cada época atualiza os pesos do modelo com base no erro calculado.
- **batch_size:**
  - define o número de amostras que serão propagadas pela rede em cada iteração (ou época) durante o treinamento. Em outras palavras, é o tamanho do lote de dados usado para atualizar os pesos da rede.
  - O modelo calcula o gradiente com base em um lote de dados e ajusta os pesos.

Etapas da classificação com Python

6. Treinar o modelo

   - **validation_data:**

     - Conjunto de dados de validação usado para avaliar o desempenho do modelo durante o treinamento.

Etapas da classificação com Python

6. Treinar o modelo

```
model.fit(X_train, y_train, epochs=100, batch_size=10);
```

– callbacks:

- Em deep learning, uma função de callback é um conjunto de funções que são aplicadas em determinados estágios do treinamento do modelo neural.
- Elas são úteis para realizar ações como salvar checkpoints do modelo, ajustar a taxa de aprendizado durante o treinamento, realizar avaliações periódicas do modelo em um conjunto de dados de validação, entre outras tarefas.

Etapas da classificação com Python

6. Treinar o modelo

```
model.fit(X_train, y_train, epochs=100, batch_size=10)
```

- **callbacks:**
- Exemplos:
  - **ModelCheckpoint:** Salva o modelo a cada época ou a cada melhoria no desempenho do modelo em um conjunto de validação. Isso permite retomar o treinamento a partir de um ponto específico caso o treinamento seja interrompido.
  - **EarlyStopping:** Interrompe o treinamento do modelo caso não haja melhoria significativa no desempenho em um conjunto de validação após um número especificado de épocas.
  - **ReduceLROnPlateau:** Reduz a taxa de aprendizado do otimizador quando o desempenho do modelo para de melhorar em um conjunto de validação.
  - **TensorBoard:** Permite a visualização dinâmica do treinamento e do desempenho do modelo utilizando a ferramenta TensorBoard, que é parte do TensorFlow.
  - **CSVLogger:** Registra o histórico do treinamento em um arquivo CSV para posterior análise.

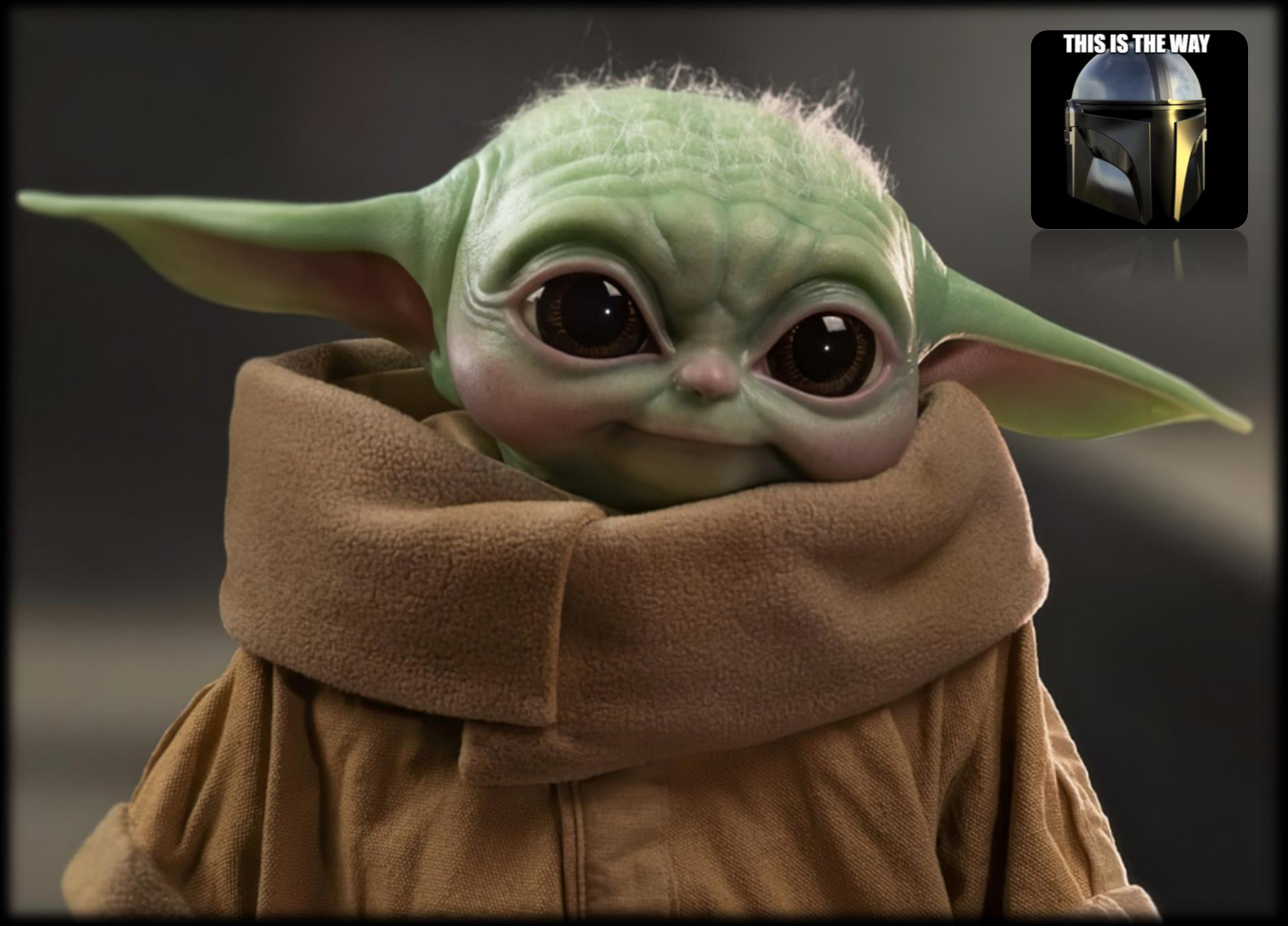
- **filepath=**'melhor_modelo.h5': Especifica o caminho onde o modelo será salvo. Neste caso, o modelo será salvo no arquivo 'melhor_modelo.h5' no diretório atual.
- **monitor='val_accuracy':** A métrica que será monitorada para determinar a melhoria do modelo. Neste caso, o callback irá verificar a acurácia no conjunto de validação ('val_accuracy').
- **save_best_only**=True: Indica se o callback deve salvar apenas o melhor modelo observado até o momento, de acordo com a métrica monitorada. Se True, o callback irá substituir o arquivo do modelo salvo apenas se o novo modelo for melhor do que o anterior.
- **verbose=1:** Controla a quantidade de informações impressas durante o treinamento. Um valor de 1 significa que mensagens informativas serão impressas, como a indicação de que o modelo foi salvo. Se definido como 0, nenhuma mensagem será impressa.

Chamada

```
from tensorflow.keras.callbacks import ModelCheckpoint

# Criar um callback para salvar o melhor modelo
checkpoint_callback = ModelCheckpoint(filepath='melhor_modelo.h5',
                                      monitor='val_accuracy',
                                      save_best_only=True,
                                      verbose=1)

# Treinar o modelo utilizando o callback
model.fit(X_train, y_train,
          epochs=10,
          batch_size=32,
          validation_data=(X_val, y_val),
          callbacks=[checkpoint_callback])
```



See You
Next time