Frameworks de IA

Curitiba/2023

1

5 Transfer Learning e Fine Tuning

2

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

2

3

Transfer Learning

- **Inception**: É uma CNN para classificação de imagens
- Antes as redes eram construídas como camadas de convolução cada vez mais profundas
- Inception é construída de forma mais complexa, voltada também em desempenho

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

3

4

Transfer Learning

- Tem-se várias arquiteturas famosas
  - https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d
- **Oxford VGG** : http://www.robots.ox.ac.uk/~vgg/research/very_deep/
- **Google Inception** : https://github.com/tensorflow/models/tree/master/research/inception
- **Microsoft ResNet** : https://github.com/KaimingHe/deep-residual-networks
- etc

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

4

Transfer Learning

- https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202
- Inception v1: https://arxiv.org/pdf/1409.4842v1.pdf
  - Imagens podem precisar de kernels de tamanhos diferentes
  - Aplicar na mesma camada, em paralelo
- Inception v2 e v3: https://arxiv.org/pdf/1512.00567v3.pdf
  - Diminuir o tamanho da convolução (mais eficiente)
  - Uso de outros otimizadores
  - etc
- Inception v4, Inception-ResNet: https://arxiv.org/pdf/1602.07261.pdf
  - Simplificação das camadas
- Etc

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

5

Transfer Learning

- O desenvolvimento dessas redes também foi impulsionado pelo ImageNet Challange
  - ILSVRC : ImageNet Large Scale Visual Recognition Challenge
  - http://www.image-net.org/challenges/LSVRC/
  - ImageNet DataSet : grande banco de dados de imagens, com propósito de pesquisa em visão computacional
  - Até 2017
- Agora Kaggle (Google)
  - https://www.kaggle.com/
  - Comunidade de cientistas de dados
  - Kaggle Competitions

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

6

7 Transfer Learning

- A construção dessas redes é muito complexa
- O treinamento é demorado (muitas GPUs, muito tempo de treinamento)
- O resultado é ótimo

- **Tem como usar uma dessas redes em meus projetos?**

7

8 Transfer Learning

- **SIM!!!**
- Isso é Transfer Learning
- ***Transfer Learning***
  - Usar um modelo pré-treinado sobre nossa base de dados
  - Congelar a rede (frozen)
  - Adaptar a saída para os nossos dados
    - Ex, função de ativação, número de classes, etc
- ***Fine Tuning***
  - Congelar parte da rede
  - A outra parte treinar

8

9

Transfer Learning

- Em que situação usar???

- Base GRANDE e muito DIFERENTE – Treinar Todo o Modelo
- Base GRANDE e PARECIDA – Usar ***Fine Tuning***
- Base PEQUENA e muito DIFERENTE – Usar ***Fine Tuning***
- Base PEQUENA e PARECIDA – ***Transfer Learning***

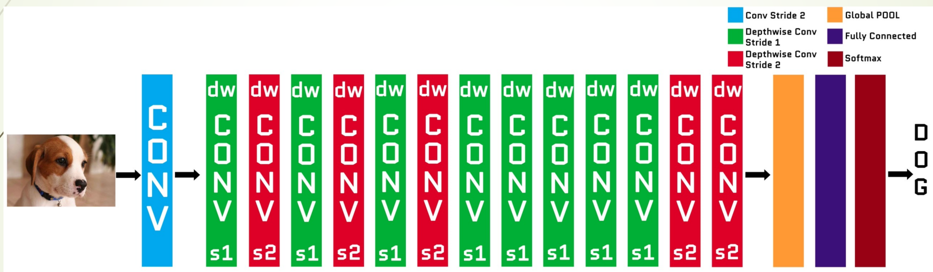
Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

9

10

Transfer Learning

- Queremos usar a MobileNets
  - https://arxiv.org/pdf/1704.04861.pdf



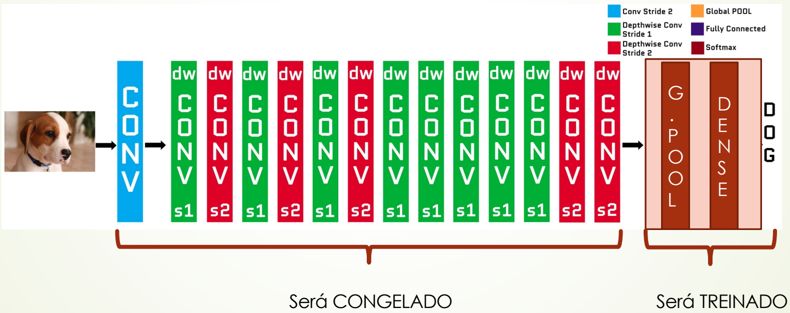
Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

10

11

Transfer Learning

- Em ***Transfer Learning***
  - Deve-se redefinir o topo: **Global Average Pooling + Dense**
  - Congela-se o modelo base
  - Treina-se as novas camadas de saída



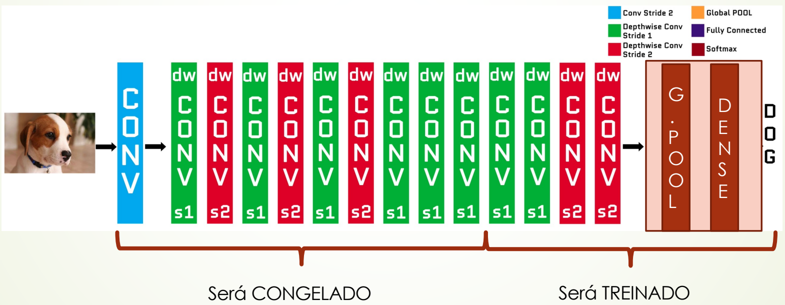
Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT   Tensorflow

11

12

Fine Tuning.

- Em ***Fine Tuning***
  - Deve-se redefinir o topo: **Global Average Pooling + Dense**
  - Congela-se a PARTE SUPERIOR (mais geral) do modelo base
  - Treina-se parte do modelo base e a nova saída



Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT   Tensorflow

12

13

Laboratório

- Classificar uma base de gatos e cachorros
- Usar uma rede pré-treinada : MobileNets
- MobileNets suporta
  - Imagens 96x96, 128x128, 160x160, 192x192, 224x224
  - 1000 classes

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

13

14

Gatos e Cachorros

- Importações

```
import os
import zipfile
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt

from tqdm import tqdm_notebook
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# plota os gráficos inline e salva no notebook
%matplotlib inline
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

14

15

Gatos e Cachorros

- Obter a base de dados

```
!wget http://www.razer.net.br/datasets/cats_and_dogs_filtered.zip
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

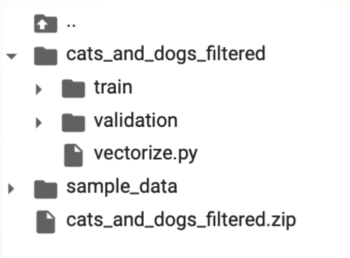
15

16

Gatos e Cachorros

- Descompactação da base

```
# descompactar a base
dataset_path = "./cats_and_dogs_filtered.zip"
zip_object = zipfile.ZipFile(file=dataset_path, mode="r")
zip_object.extractall("./")
zip_object.close()
```



Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

16

17

Gatos e Cachorros

- Diretório de Treino e Teste

```
dataset_path_new = "./cats_and_dogs_filtered"
train_dir = os.path.join(dataset_path_new, "train")
test_dir = os.path.join(dataset_path_new, "validation")
```


17

18

Gatos e Cachorros

- Carregar modelo pré-treinado: MobileNet

```
input_shape = (128, 128, 3) # tamanho da imagem entrada
base_model = tf.keras.applications.MobileNetV2(
                     input_shape=input_shape,
                     include_top=False,
                     weights="imagenet")
base_model.summary()
```

- `include_top = False` : não vai carregar o final da rede, que deverá ser definida por nós

18

19

Gatos e Cachorros

- Congelar modelo base

```
# Congelar modelo base
base_model.trainable = False
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

19

20

Gatos e Cachorros

- Adicionar Cabeçalho Personalizado

```
# Adicionar cabeçalho personalizado
print(base_model.output.shape)

# reduzir a dimensionalidade (4 x 4 x 1280 = 20480 pesos para treinar!!)
# Global Average Pooling
global_average_layer = tf.keras.layers.GlobalAveragePooling2D()(base_mod
el.output)

print(global_average_layer.shape)

prediction_layer = tf.keras.layers.Dense(units=1, activation="sigmoid")(
global_average_layer)
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

20

21

Gatos e Cachorros

- Definição do Modelo

```
# Definição do modelo (unir o base com o personalizado)
model = tf.keras.models.Model(base_model.input,
                              prediction_layer)
model.summary()
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

21

22

Gatos e Cachorros

- Compilar o modelo

```
# compilar o modelo
# testar com o Adam também

model.compile(optimizer=tf.keras.optimizers.RMSprop(
                                 learning_rate=0.0001),
              loss="binary_crossentropy",
              metrics = ["accuracy"])
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

22

23

Gatos e Cachorros

- Carregar as imagens

```
# carga das imagens
# Pré-processamento /255
data_gen_train = ImageDataGenerator(rescale=1/255.)
data_gen_test  = ImageDataGenerator(rescale=1/255.)

train_generator = data_gen_train.flow_from_directory(
                          train_dir,
                          target_size = (128,128),
                          batch_size  = 128,
                          class_mode  = "binary")

test_generator = data_gen_train.flow_from_directory(
                          test_dir,
                          target_size = (128,128),
                          batch_size  = 128,
                          class_mode  = "binary")
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

23

24

Gatos e Cachorros

- Treinar o modelo

```
# Treinar o modelo
EPOCHS = 5

r = model.fit(train_generator,
              epochs          = EPOCHS,
              validation_data = test_generator)
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

24

25 Gatos e Cachorros

- Plotar a *Loss-function*

```
plt.plot(r.history["loss"], label="loss")
plt.plot(r.history["val_loss"], label="val_loss")
plt.xlabel("Épocas")
plt.ylabel("Loss")
plt.xticks( np.arange(0, EPOCHS, step=1) )
plt.legend()
```


25

26 Gatos e Cachorros

- Plotar a Acurácia

```
plt.plot(r.history["accuracy"], label="accuracy")
plt.plot(r.history["val_accuracy"], label="val_accuracy")
plt.xlabel("Épocas")
plt.ylabel("Acurácia")
plt.xticks( np.arange(0, EPOCHS, step=1) )
plt.legend()
```


26

27

Gatos e Cachorros.

- Avaliar o Modelo

```
# Avaliação do Modelo
val_loss, val_accuracy = model.evaluate (test_generator)

print(val_accuracy)
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

27

28

Fine Tuning: Gatos e Cachorros

- Efetuar o ***Fine Tuning***: Congelar as camadas até a 100

```
# Efetuar o Fine Tuning
# Primeiro fazer o Transfer Learning
# Depois o Fine Tuning

# descongelar algumas camadas
base_model.trainable = True
len(base_model.layers)

# deixar o início da base
# congelar o final
fine_tuning_at = 100

for l in base_model.layers[:fine_tuning_at]:
  l.trainable = False
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

28

29

Fine Tuning: Gatos e Cachorros

- Compilar e Treinar o Modelo (fine tuning)

```
#compilar o modelo
# testar com o Adam tb

model.compile(optimizer = tf.keras.optimizers.RMSprop(
                                   learning_rate=0.0001),
              loss      = "binary_crossentropy",
              metrics   = ["accuracy"])

# Treinar o modelo
EPOCHS = 5
r = model.fit(train_generator,
              epochs = EPOCHS,
              validation_data = test_generator)
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

29

30

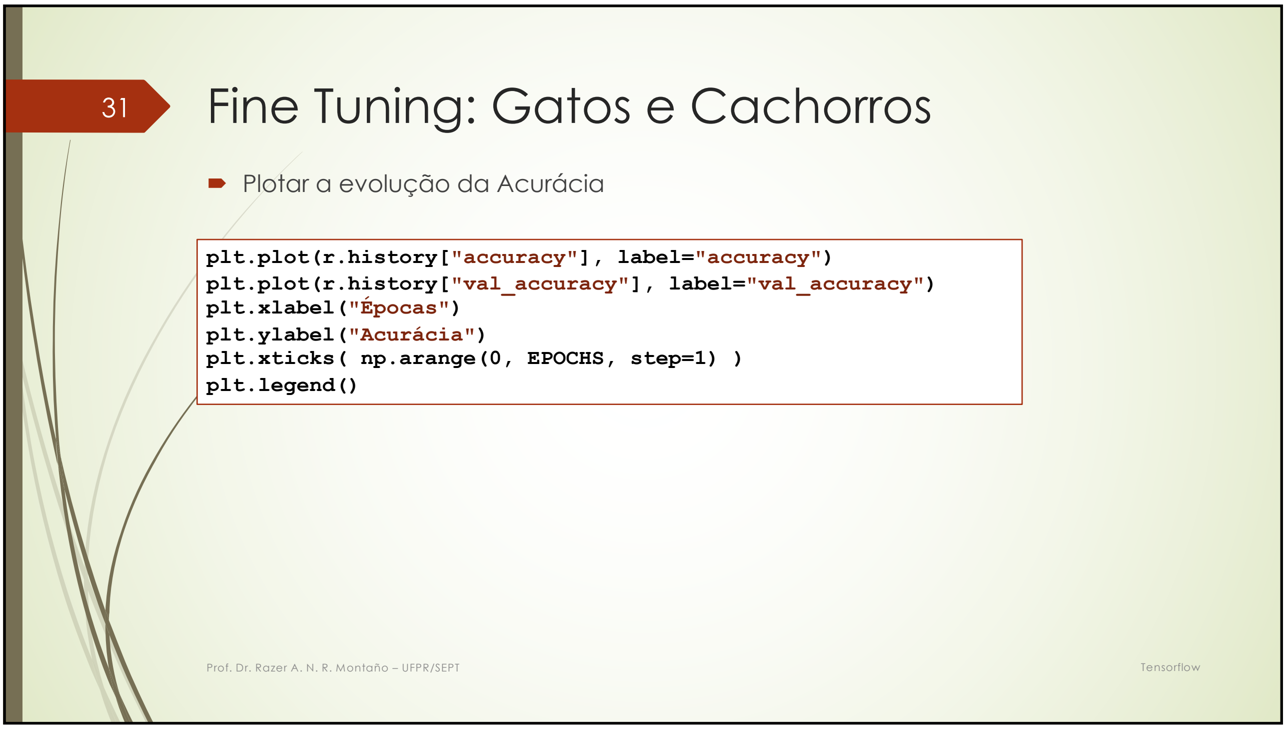
Fine Tuning: Gatos e Cachorros

- Plotar a evolução da *Loss-Function*

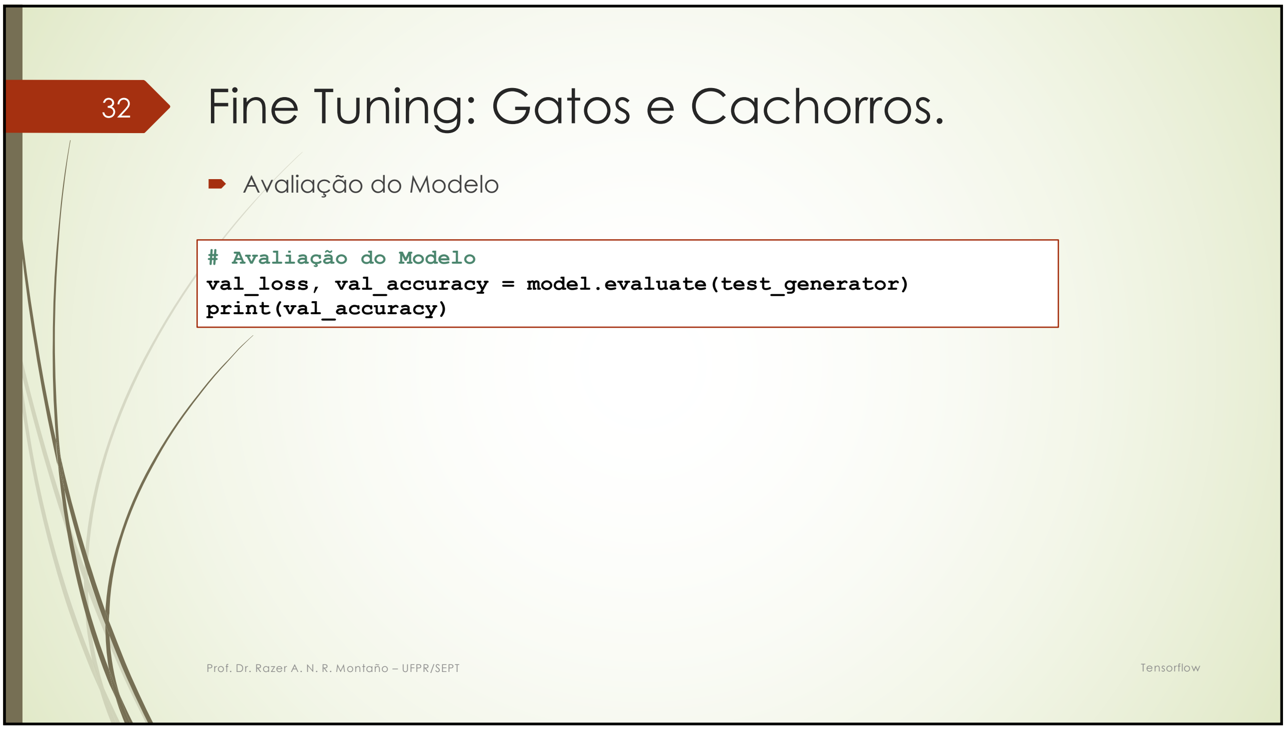
```
plt.plot(r.history["loss"], label="loss")
plt.plot(r.history["val_loss"], label="val_loss")
plt.xlabel("Épocas")
plt.ylabel("Loss")
plt.xticks( np.arange(0, EPOCHS, step=1) )
plt.legend()
```

Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT Tensorflow

30


31
Fine Tuning: Gatos e Cachorros
Plotar a evolução da Acurácia
plt.plot(r.history["accuracy"], label="accuracy")
plt.plot(r.history["val_accuracy"], label="val_accuracy")
plt.xlabel("Épocas")
plt.ylabel("Acurácia")
plt.xticks(np.arange(0, EPOCHS, step=1))
plt.legend()
Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT
Tensorflow

31


32
Fine Tuning: Gatos e Cachorros.
Avaliação do Modelo
Avaliação do Modelo
val_loss, val_accuracy = model.evaluate(test_generator)
print(val_accuracy)
Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT
Tensorflow

32


33
EXERCÍCIO..
Executar o exercício de Transfer Learning e Fine Tuning.
Prof. Dr. Razer A. N. R. Montaño – UFPR/SEPT
Tensorflow

33