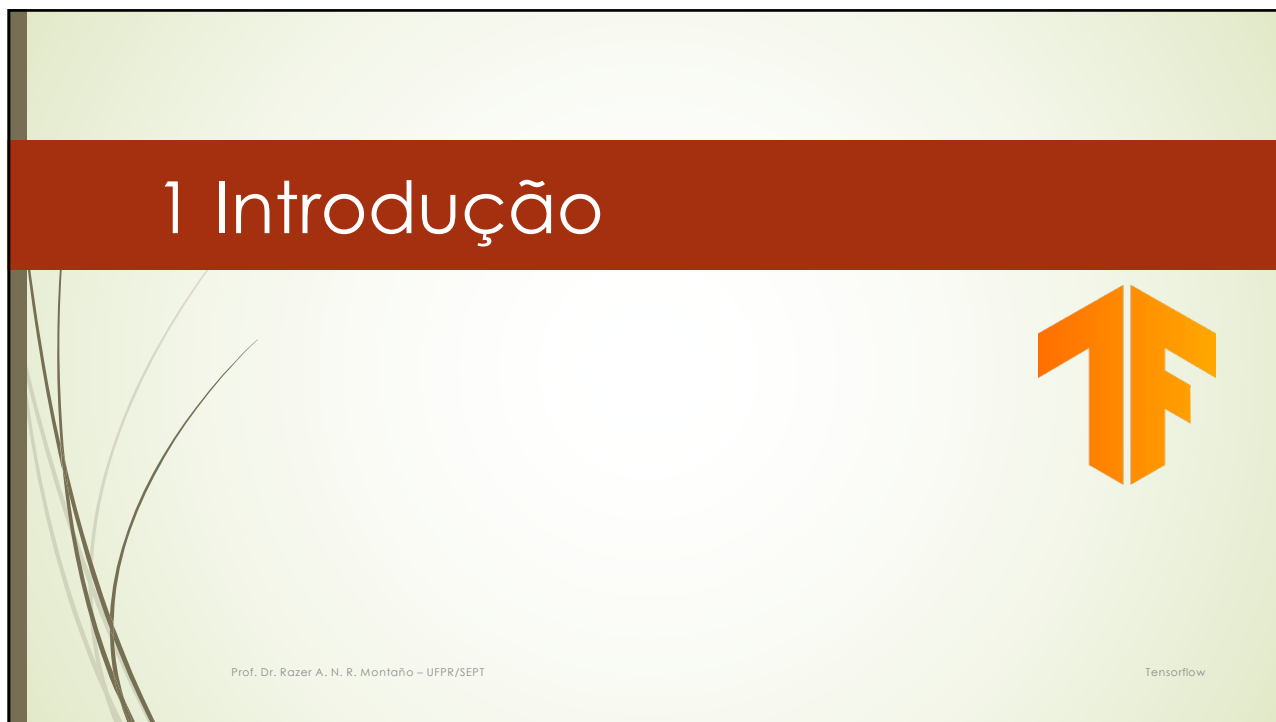




1



2



3



4

5

História

- 1921 - O termo "robô" é usado pela primeira vez por Karel Capek
- 1950 – Teste de Turing
- 1952 – Samuel escreveu o primeiro programa que aprendia : jogo de damas
- 1956 – John McCarty - Campo da IA é "fundado" em uma conferência em Dartmouth
- 1960 – Rosenblatt – Perceptron, aprendizado por tentativa e erro
- 1964 – Primeiro chatbot Eliza, capaz de manter uma conversa
- 1975 – Surge o backpropagation
- 1989 – ALVINN – primeiro carro autônomo guiado por rede neural (CMU)
- 1997 – Deep Blue vence Garry Kasparov em um torneio de xadrez
- 1997 – Pathfinder da Nasa pousa em Marte
- 1997 – Surge o Big Data

Prof. Dr. Razer A. N. R. Montano – UFPR/SEPT

Tensorflow

5

6

Introdução.

- 2000 – Surge o termo Deep Learning (no contexto de redes neurais artificiais)
- 2002 – Aspirador de pó autônomo que aprende o ambiente
- 2005 – 5 veículos autônomos completam um percurso de 212km off-road
- 2011 – Surge a Siri
- 2011 – IBM Watson vence um programa de perguntas e respostas (Jeopardy)
- 2012 – Surge o Google Now em resposta à Siri
- 2012 – Deep Learning
- 2014 – DeepFace – reconhecimento facial com um grande avanço na acurácia
- 2014 – GANs – Generative Adversarial Networks – aprendizado por competição
- 2016 – AlphaGo (Google Deepmind) vence o campeão de GO pela primeira vez
- 2016 – LipNet – Leitor labial (Oxford)
- 2016 – AlphaGo
- 2017 – Deepfake pornográfico
- 2017 – Transformers – novas arquiteturas para processamento de linguagem natural (NLP)

Prof. Dr. Razer A. N. R. Montano – UFPR/SEPT

Tensorflow

6

7

Introdução.

- 2018 – AlphaZero
- 2019 – GPT-2
- 2019 – Deepfakes em Filmes (O Irlandês e Projeto Gemini)
- 2020 – GPT-3
- 2021 – Dall-e
- 2022 – ChatGPT
- 2022 – AI Bill of Rights

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

7

8

Frameworks de IA

- Accord.NET: <http://accord-framework.net>
- Apache Mahout: <http://mahout.apache.org>
- Apache MxNet: <https://mxnet.apache.org>
- Apache SystemML: <http://apache.github.io/systemml/algorithms-reference>
- AutoML: <https://cloud.google.com/automl> , <https://arxiv.org/pdf/1907.08392v1.pdf>
- AWS ML: <https://aws.amazon.com/pt/machine-learning/>
- Azure ML : <https://azure.microsoft.com/pt-br/services/machine-learning/>
- Caffe2 (Facebook): <https://caffe2.ai>
- Deeplearning4J: <https://deeplearning4j.org>

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

8

9

Frameworks de IA

- IBM Watson: <https://www.ibm.com/br-pt/cloud/machine-learning>
- H2O: <https://www.h2o.ai/products/h2o/>
- Keras: <https://keras.io>
- Microsoft CNTK: <https://github.com/Microsoft/CNTK>
- MLPack: <https://www.mlpack.org>
- Mycroft : <https://mycroft.ai>
- Neuroph : <https://github.com/neuroph/neuroph>
- ONNX (Facebook): <https://onnx.ai>
- OpenCog: <https://github.com/opencog/>
- OpenNN : <http://www.opennn.net>
- PyTorch (Facebook): <https://code.fb.com/ml-applications/announcing-pytorch-1-0-for-both-research-and-production/>

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

9

10

Frameworks de IA.

- Spark MLlib: <https://spark.apache.org/mllib/>
- Sci-kit learn: <https://scikit-learn.org/stable/>
- Torch : <http://torch.ch>
- TensorFlow (Google): <https://www.tensorflow.org>
- Theano: <https://github.com/Theano/Theano>

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

10

11

Frameworks de IA – Plataformas Cloud.

- Amazon AWS Machine Learning
 - Desde 2015
 - Suporta Tensorflow desde 2018
- Microsoft Azure Machine Learning
 - Desde 2015
- Google Cloud
 - Instâncias com TPU (*Tensor Processing Unit*)
 - Tensorflow
- IBM Watson
 - Criação de APIs

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

11

12

Frameworks de IA – Empresas.

- Top 5 Machine Learning Companies to Work for in 2020
 - <https://www.simplilearn.com/top-machine-learning-companies-article>
- Artificial Intelligence Companies You May Want To Consider (para investir)
 - <https://www.forbes.com/sites/qai/2023/02/24/artificial-intelligence-companies-you-may-want-to-invest-in-now/?sh=65c8acbf6bfb>
- Top Deep Learning Startups in 2023
 - <https://angel.co/startups/industry/deep-learning-2>
- Top 13 BEST Machine Learning Companies [Updated 2023 List]
 - <https://www.softwaretestinghelp.com/machine-learning-companies/>

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

12

1.2 Introdução ao TensorFlow

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

13

14

Introdução Tensorflow



- Framework de aprendizado de máquina
 - Projetar, Construir, Treinar modelos de aprendizado (profundo – DL)
- Desenvolvido pela Google Brain (2011+)
 - Equipe de IA do Google
 - Licença código aberto Apache 2.0 em 09/11/2015
- Pode ser executado em CPU e GPU (CUDA) e TPU (*Tensor Processing Units*)
 - CUDA: API de computação paralela (Nvidia)
 - Instruções de GPU
- Disponível em Linux, MacOS, Windows, Android, iOS
 - Cloud

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

14

15

Introdução Tensorflow

- Cálculos matemáticos feitos em grafos e fluxos de dados
- Grafo
 - Nós: representam operações matemáticas
 - Arestas: representam os dados : arrays ou tensores
 - Tensores: vetores n-dimensionais
 - Acíclico e dirigido
 - Compilado e otimizado
 - Executado nos dispositivos disponíveis (CPU, GPU, TPU)
 - Dados (tensores) fluem pelo grafo

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

15

16

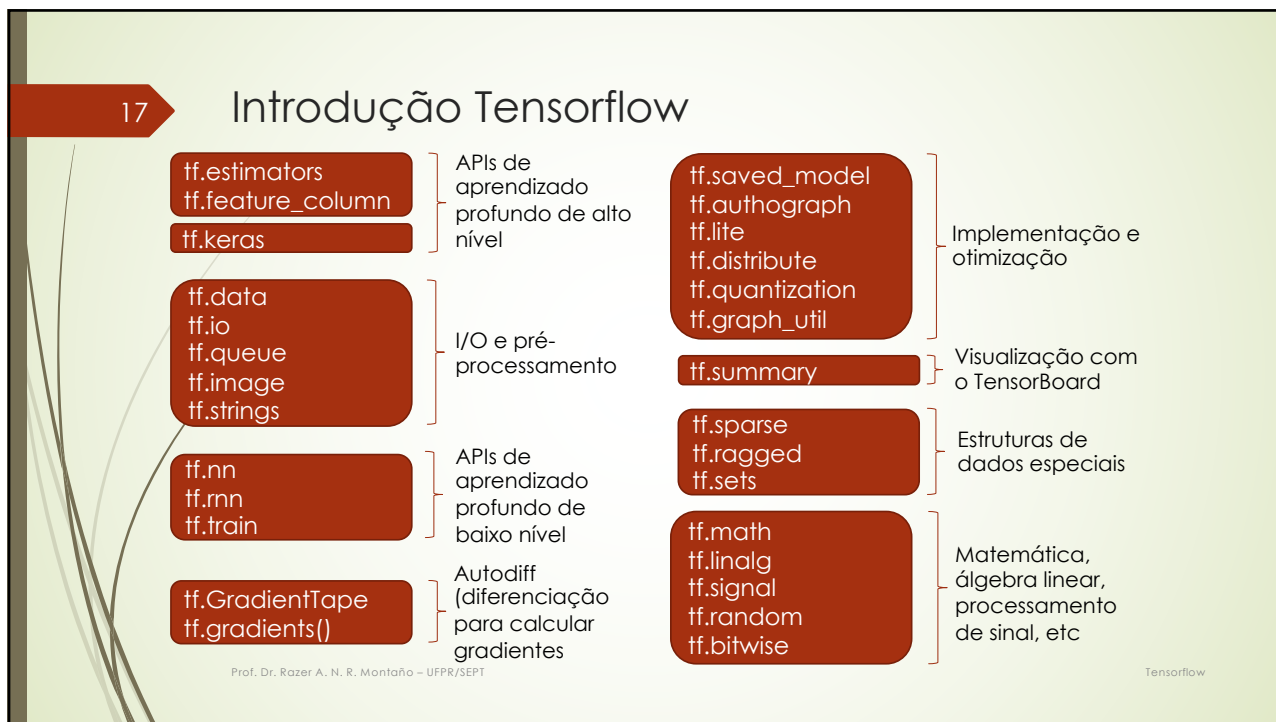
Introdução Tensorflow

- Algumas bibliotecas serão usadas
 - KERAS (integrada ao TF 2.0)
 - PANDAS: carregar dados (CSV) de forma tabular
 - NUMPY: representação de tensores/arrays, aritmética matricial,
 - MATPLOTLIB: gráficos
 - SCIPY: funções de mais alto nível
 - SCIKIT-LEARN: *machine learning*
 - etc

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

16



17

18

Introdução Tensorflow.

► Casos de Uso

- <https://www.tensorflow.org/about/case-studies>
- Airbnb
- Airbus
- Coca-cola
- Google
- Intel
- Lenovo
- Paypal
- Qualcomm
- Twitter

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

18

19

Instalação do TensorFlow

- Acessar:

<https://www.tensorflow.org/install/pip>

- Necessários Python 3 instalado

```
raser@Neumann:~$ python --version  
Python 3.7.4
```

```
$ pip3 install tensorflow==2.0.0-rc0
```

- Executa-se

```
$ pip3 list | grep tensorflow
```

- Para ver a versão instalada do TensorFlow

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

19

20

Primeiro Programa

- Escreva um script: **programa1.py**

```
# Importando o tensorflow  
import tensorflow as tf  
  
# Inicializando as constantes  
x1 = tf.constant([1,2,3,4])  
x2 = tf.constant([5,6,7,8])  
  
# Multiplicando os Tensores  
result = tf.multiply(x1, x2)  
  
# Print the result  
print(result)
```

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

20

21

Primeiro Programa

➡ Execute

```
$ python program1.py
```

➡ Resultado

```
2019-08-27 11:09:52.187838: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x7fe490a21c10 executing computation
s on platform Host. Devices:
2019-08-27 11:09:52.187871: I tensorflow/compiler/xla/service/service.cc:175] StreamExecutor device (0): Host, Default Versi
on
tf.Tensor([ 5 12 21 32], shape=(4,), dtype=int32)
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

21

22

Primeiro Programa

```
# Importando o tensorflow
import tensorflow as tf

# Inicializando as constantes
x1 = tf.constant([1,2,3,4])
x2 = tf.constant([5,6,7,8])

# Multiplicando os Tensores
result = tf.multiply(x1, x2)

# Print the result
print(result)
```

$x1 = [1, 2, 3, 4]$
 $x2 = [5, 6, 7, 8]$

$$\begin{matrix} [1, 2, 3, 4] \\ * * * * \\ [5, 6, 7, 8] \end{matrix} = [5, 12, 21, 32]$$

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

22

23

Google Colaboratory

- Serviço na Nuvem
- Incentivar o estudo em Aprendizado de Máquina
- Suporta a Python 2 e 3
- Aceleração de GPU (GPU Tesla K80)
- Bibliotecas pré-instaladas
- Com base no Jupyter Notebook no Google Drive
 - Armazenados em seu Google Drive
- Suporta comandos bash

Prof. Dr. Razer A. N. R. Montano – UFPR/SEPT

Tensorflow

23

24

Google Colaboratory

- Serviço na Nuvem
 - <https://colab.research.google.com/>
 - <https://www.youtube.com/watch?v=inN8seMm7UI>
- Gratuito: 12 GB de RAM, GPU Tesla K80, Até 12 horas de execução
 - Pro: R\$ 58,00, 32 GB de RAM, GPU NVIDIA T4, Tesla P100
 - Pro+: R\$ 258,00, 52 GB de RAM, GPU NVIDIA T4, Tesla P100, com maior prioridade

	Colab Free	Colab Pro	Colab Pro +
Guarantee of resources	Low	High	Even Higher
GPU	K80	K80, T4 and P100	K80, T4 and P100
RAM	16 GB	32 GB	52 GB
Runtime	12 hours	24 hours	24 hours
Background execution	No	No	Yes
Costs	Free	9.99\$ per month	49.99\$ per month
Target group	Casual user	Regular user	Heavy user

FONTE: <https://towardsdatascience.com/google-colab-pro-is-it-worth-49-99-c542770b8e56>

Prof. Dr. Razer A. N. R. Montano – UFPR/SEPT

Tensorflow

24

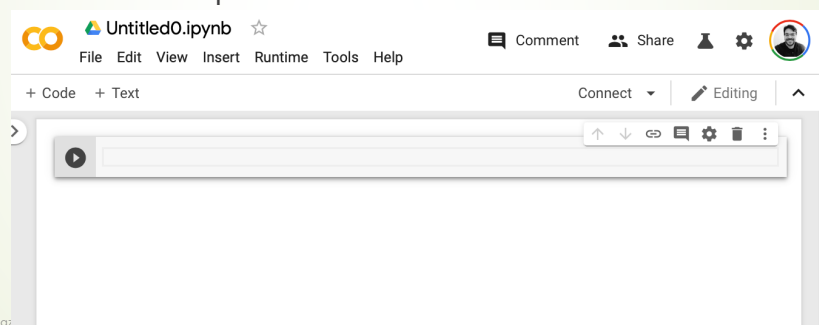
25

Criar um Notebook (Exemplo 1)

➤ PASSO 1: Logado no seu Drive, abra o Google Colab

➤ <https://colab.research.google.com>

➤ PASSO 2: Clique em “Novo Notebook”



25

26

Criar um Notebook (Exemplo 2)

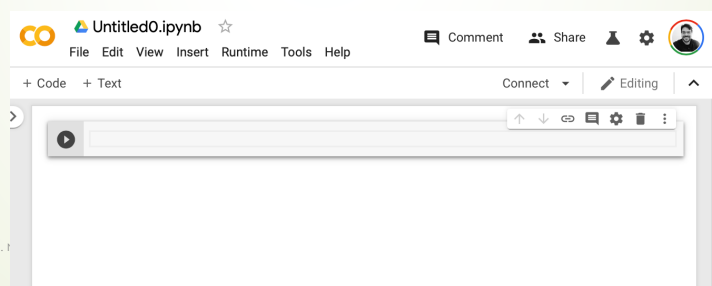
➤ PASSO 1: Abra o Google Drive

➤ <http://drive.google.com>

➤ PASSO 2: Crie uma nova pasta para o projeto

➤ PASSO 3: Entre na pasta

➤ PASSO 4: Clique em Novo | Mais | Colaboratório

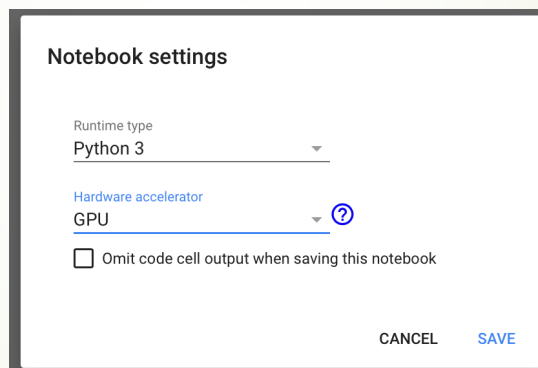


26

27

Configurar a Aceleração de GPU

- Clique em **Edit | Notebook Settings**
- Na opção **Hardware accelerator** escolha **GPU**



Prof. Dr. Razer A. N. R. Montano – UFPR/SEPT

Tensorflow

27

28

Executar uma célula

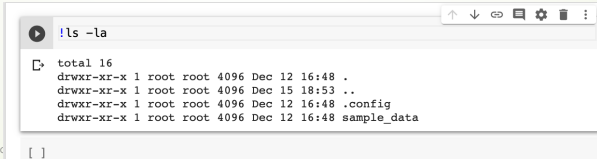
- Em uma célula digite o comando
 - Comandos bash iniciam com “!”
- Para executar pressione o botão “play” ou SHIFT+ENTER

```
!ls -la
```



```
!ls -la
```

- Como resultado



```
!ls -la
total 16
drwxr-xr-x 1 root root 4096 Dec 12 16:48 .
drwxr-xr-x 1 root root 4096 Dec 15 18:53 ..
drwxr-xr-x 1 root root 4096 Dec 12 16:48 .config
drwxr-xr-x 1 root root 4096 Dec 12 16:48 sample_data
```

Prof. Dr. Razer A. N. R. Montano

Tensorflow

28

29

Instalar o TensorFlow 2.0

- Para instalar uma versão específica
- Execute o comando:

```
!pip install tensorflow-gpu==2.1.0.rc2
```

- Como resultado:

```
!pip install tensorflow-gpu==2.0.0.alpha0
Collecting tensorflow-gpu==2.0.0.alpha0
  Downloading https://files.pythonhosted.org/packages/1a/66/32cfa089523219d53f6bdc2a16437bde1ac/...
Requirement already satisfied: gast==0.2.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Collecting tf-estimator-nightly==1.14.0.dev20190301 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/13/82/f16063bdeed210dc2ab857930ac1da4fba1e51b/...
Requirement already satisfied: wheel==0.30 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: google-pasta==0.1.2 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: astor==0.6.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Collecting tb-nightly==1.14.0a20190302 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/89/51/ae1d73644b4642c03846115e4c058eff27ac/...
Requirement already satisfied: six==1.10.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: keras-preprocessing==1.0.5 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: keras-applications==1.0.6 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: numpy==1.14.5 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: grpcio==1.8.6 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: absl-py==0.7.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: termcolor==1.1.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: protobuf==3.6.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: werkzeug==0.11.15 in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: h5py in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from tensorflow)
Installing collected packages: tf-estimator-nightly, tb-nightly, tensorflow-gpu
Successfully installed tb-nightly-1.14.0a20190301 tensorflow-gpu-2.0.0a0 tf-estimator-nightly-1.14.0
WARNING: The following packages were previously imported in this runtime:
(tensorboard,tensorflow,tensorflow_estimator)
You must restart the runtime in order to use newly installed versions.

RESTART RUNTIME
```

- Clicar em “Restart Runtime”

Tensorflow

29

30

Instalar o TensorFlow 2.0

- Para adicionar mais uma célula
- Passe o mouse no final da célula anterior e clique em “+ Code”

```
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from protobuf>=
Installing collected packages: tf-estimator-nightly, tb-nightly, tensorflow-gpu
Successfully installed tb-nightly-1.14.0a20190301 tensorflow-gpu-2.0.0a0 tf-estimator-nightly-1.14.0
WARNING: The following packages were previously imported in this runtime:
(tensorboard,tensorflow,tensorflow_estimator)
You must restart the runtime in order to use newly installed versions.

RESTART RUNTIME
```

Code

Text

↑

Tensorflow

30

31

Instalar o TensorFlow 2.0

Verificar a versão do TF

```
import tensorflow as tf
tf.__version__
```

Como resultado:

```
[2] import tensorflow as tf
tf.__version__
/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/dtypes.py
_np_qint8 = np.dtype([('qint8', np.int8, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/dtypes.py
_np_qint8 = np.dtype([('qint8', np.uint8, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/dtypes.py
_np_qint16 = np.dtype([('qint16', np.int16, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/dtypes.py
_np_qint16 = np.dtype([('qint16', np.uint16, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/dtypes.py
_np_qint32 = np.dtype([('qint32', np.int32, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/dtypes.py
_np_resource = np.dtype([('resource', np.ubyte, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/compat/tensorflow_stub/dt
_np_qint8 = np.dtype([('qint8', np.int8, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/compat/tensorflow_stub/dt
_np_qint16 = np.dtype([('qint16', np.int16, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/compat/tensorflow_stub/dt
_np_qint16 = np.dtype([('qint16', np.uint16, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/compat/tensorflow_stub/dt
_np_qint32 = np.dtype([('qint32', np.int32, 1)])
/usr/local/lib/python3.6/dist-packages/tensorflow/compat/tensorflow_stub/dt
_np_resource = np.dtype([('resource', np.ubyte, 1)])
'2.0.0-alpha0'
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

31

32

Executar o Primeiro Programa

Adicione mais uma Célula

Digite o código

```
# Inicializando as constantes
x1 = tf.constant([1,2,3,4])
x2 = tf.constant([5,6,7,8])

# Multiplicando os Tensores
result = tf.multiply(x1, x2)

# Print the result
print(result)
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

32

33

Executar o Primeiro Programa.

➡ Execute a Célula

```
# Inicializando as constantes
x1 = tf.constant([1,2,3,4])
x2 = tf.constant([5,6,7,8])

# Multiplicando os Tensores
result = tf.multiply(x1, x2)

# Print the result
print(result)
```

tf.Tensor([5 12 21 32], shape=(4,), dtype=int32)

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

33

34

Subir dados para o Google Colab

➡ Usando WGET

```
!wget https://archive.ics.uci.edu/ml/machine-learning-databases/arrhythmia/arrhythmia.data
```

➡ Resultado

```
[12] !wget https://archive.ics.uci.edu/ml/machine-learning-databases/arrhythmia/arrhythmia.data
--2019-12-15 21:52:29-- https://archive.ics.uci.edu/ml/machine-learning-databases/arrhythmia/arrhythmia.data
Resolving archive.ics.uci.edu (archive.ics.uci.edu)... 128.195.10.252
Connecting to archive.ics.uci.edu (archive.ics.uci.edu)|128.195.10.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 402355 (393K) [application/x-httpd-php]
Saving to: 'arrhythmia.data'

arrhythmia.data 100%[=====] 392.92K --.-KB/s in 0.03s
2019-12-15 21:52:29 (12.8 MB/s) - 'arrhythmia.data' saved [402355/402355]
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

34

35

Subir dados para o Google Colab

Verificando o dataset obtido

```
!ls -l
```

Resultado

```
[14] !ls -l
```

```
total 400  
-rw-r--r-- 1 root root 402355 Apr  1 1998 arrhythmia.data  
drwxr-xr-x 1 root root  4096 Dec 12 16:48 sample_data
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

35

36

Subir dados para o Google Colab

Pode-se carregar os dados com PANDAS

```
import pandas as pd  
df = pd.read_csv("arrhythmia.data", header=None)
```

Cortar algumas colunas e nomeá-las

```
data = df[[0,1,2,3,4,5]]  
data.columns = ['age', 'sex', 'height', 'weight', 'QRS duration',  
                'P-R interval']
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

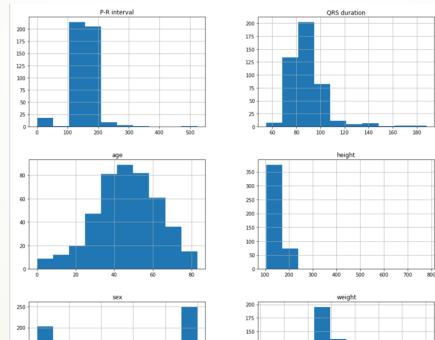
36

37

Subir dados para o Google Colab

- Pode-se gerar histogramas dos dados

```
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [15, 15]
data.hist();
```



Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

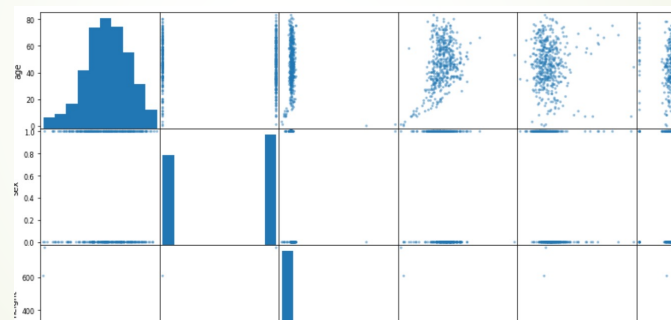
37

38

Subir dados para o Google Colab

- Pode-se gerar gráfico de dispersão dos dados

```
from pandas.plotting import scatter_matrix
scatter_matrix(data);
```



Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

38

39

Subir datos para o Google Colab

➤ Usando Keras

```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data"
tf.keras.utils.get_file('auto-mpg.data', url)
```

➤ Resultado

```
[29] url = "https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data"
      tf.keras.utils.get_file('auto-mpg.data', url)

↳ Downloading data from https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data
32768/30286 [=====] - 0s 0us/step
'/root/.keras/datasets/auto-mpg.data'
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

39

40

Subir datos para o Google Colab

➤ Verificando o dataset obtido

```
!head /root/.keras/datasets/auto-mpg.data
```

➤ Resultado

```
[31] !head /root/.keras/datasets/auto-mpg.data

↳ 18.0  8  307.0    130.0    3504.    12.0  70  1  "chevrolet chevelle malibu"
15.0  8  350.0    165.0    3693.    11.5  70  1  "buick skylark 320"
16.0  8  318.0    150.0    3436.    11.0  70  1  "plymouth satellite"
17.0  8  304.0    150.0    3433.    12.0  70  1  "amc rebel sst"
17.0  8  302.0    140.0    3449.    10.5  70  1  "ford torino"
15.0  8  429.0    198.0    4341.    10.0  70  1  "ford galaxie 500"
14.0  8  454.0    220.0    4354.    9.0  70  1  "chevrolet impala"
14.0  8  440.0    215.0    4312.    8.5  70  1  "plymouth fury iii"
14.0  8  455.0    225.0    4425.    10.0  70  1  "pontiac catalina"
15.0  8  390.0    190.0    3850.    8.5  70  1  "amc ambassador dpl"
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

40

41

Subir dados para o Google Colab

Carregar o dataset

```
df = pd.read_csv('/root/.keras/datasets/auto-mpg.data', header=None,
delim_whitespace=True)

df.head()
```

```
[32] df = pd.read_csv('/root/.keras/datasets/auto-mpg.data', header=None, delim_whitespace=True)
df.head()
```

	0	1	2	3	4	5	6	7	8
0	18.0	8	307.0	130.0	3504.0	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693.0	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150.0	3436.0	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150.0	3433.0	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140.0	3449.0	10.5	70	1	ford torino

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

41

42

Subir dados para o Google Colab

Upload de arquivo

```
from google.colab import files
uploaded = files.upload()
```

```
from google.colab import files
uploaded = files.upload()
```

... Escolher Arquivos nenhum arquivo selecionado Cancel upload

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

42

43

Subir dados para o Google Colab.

► Verificar os dados

```
df = pd.read_csv('acacia__todos__preproc.csv', error_bad_lines=False)
df.head()
```

► Também é possível fazer upload de Scripts Python e usar funções como bibliotecas

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

43

44

Acessar Dados do Google Drive

Carregar os dados

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

Deve-se entrar no link que aparece

Autorizar

Copiar a chave de autorização

Colar a chave de autorização no campo mostrado

```
[43] from google.colab import drive  
     drive.mount('/content/gdrive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?response_type=token&scope=https://www.googleapis.com/auth/drive.readonly  
Enter your authorization code:  
.....  
Mounted at /content/gdrive
```

Prof. Dr. Razer A. N. R. Montañó – UFPR/SEPT

Tensorflow

44

45

Acessar Dados do Google Drive..

➡ Pode-se navegar pelos diretórios

```
!ls -l
```

```
!ls -l gdrive
```

```
!ls -l 'gdrive/My Drive/'
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

45

1.3 Sintaxe Básica

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

46

47

Conteúdo

- Tensores
- Constantes
- Variáveis
- Operações básicas com tensores
- Strings
- Funções o TF

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

47

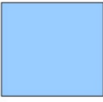
48

Tensores

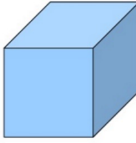
- Tensor
 - Array multidimensional
 - 0D – número escalar
 - 1D – vetor
 - 2D – matriz
 - etc




1d-tensor



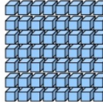
2d-tensor



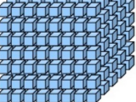
3d-tensor



4d-tensor



5d-tensor



6d-tensor

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

48

49

Constantes

- Um tensor que não muda de valores
- Sintaxe

```
x1 = tf.constant([1,2,3,4])
```

- Resultado

```
[7] x1 = tf.constant([1,2,3,4])  
x1  
↳ <tf.Tensor: id=6, shape=(4,), dtype=int32, numpy=array([1, 2, 3, 4], dtype=int32)>
```

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

49

50

Constantes

- Detalhes

```
[7] x1 = tf.constant([1,2,3,4])  
x1  
↳ <tf.Tensor: id=6, shape=(4,), dtype=int32, numpy=array([1, 2, 3, 4], dtype=int32)>
```

Forma do tensor
(dimensões)

Tipo dos dados

Array numpy

Tipo dos dados

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

50

51

Constantes

➤ Mais dimensões

```
x3 = tf.constant([[1,2,3,4] , [5,6,7,8]])
```

➤ Resultado

```
[8] x3 = tf.constant([[1,2,3,4] , [5,6,7,8]])
x3
[> <tf.Tensor: id=8, shape=(2, 4), dtype=int32, numpy=
array([[1, 2, 3, 4],
       [5, 6, 7, 8]], dtype=int32)>
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

51

52

Constantes.

➤ Mais dimensões

```
x4 = tf.constant([[[1,2,3,4] , [5,6,7,8]], [[9,10,11,12] ,
[13,14,15,16]]])
```

➤ Resultado

```
[9] x4 = tf.constant([[[1,2,3,4] , [5,6,7,8]], [[9,10,11,12] , [13,14,15,16]]])
x4
[> <tf.Tensor: id=10, shape=(2, 2, 4), dtype=int32, numpy=
array([[[ 1,  2,  3,  4],
         [ 5,  6,  7,  8]],
       [[ 9, 10, 11, 12],
         [13, 14, 15, 16]]], dtype=int32)>
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

52

53

Variáveis

- Um tensor que pode mudar de valores
- Sintaxe

```
v1 = tf.Variable([[1., 2., 3.], [4., 5., 6.]])
```

➤ Resultado

```
v1 = tf.Variable([[1., 2., 3.], [4., 5., 6.]])
v1
```

```
<tf.Variable 'Variable:0' shape=(2, 3) dtype=float32, numpy=
array([[1., 2., 3.],
       [4., 5., 6.]]) dtype=float32>
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

53

54

Variáveis.

➤ Detalhes

```
v1 = tf.Variable([[1., 2., 3.], [4., 5., 6.]])
v1
|
<tf.Variable 'Variable:0' shape=(2, 3) dtype=float32, numpy=
array([[1., 2., 3.],
       [4., 5., 6.]]) dtype=float32>
```

Forma do tensor
(dimensões)

Tipo dos dados

Array
numpy

Tipo dos dados

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

54

55

Diferença Variable e Constant.

```
[11] x = tf.Variable([1, 2, 3])
     y = tf.constant([10, 20, 30])
     print(x)
     print(y)

<tf.Variable 'Variable:0' shape=(3,) dtype=int32, numpy=array([1, 2, 3], dtype=int32)>
tf.Tensor([10 20 30], shape=(3,), dtype=int32)

[12] x.assign([4, 5, 6])
     print(x)

<tf.Variable 'Variable:0' shape=(3,) dtype=int32, numpy=array([4, 5, 6], dtype=int32)>

y.assign([4, 5, 6])
print(y)

-----
AttributeError                                Traceback (most recent call last)
<ipython-input-13-96bfe3de2228> in <module>()
----> 1 y.assign([4, 5, 6])
      2 print(y)

AttributeError: 'tensorflow.python.framework.ops.EagerTensor' object has no attribute 'assign'
```

Prof. Dr. Razer A. N. R. Montañó - UFPR/SEPT

Tensorflow

55

56

Operações Básicas com Tensores

➡ Adição (+), Subtração (-), Multiplicação (*), Divisão (/)

```
[15] c1 = tf.constant([1.0, 2.0, 3.0])
     c2 = tf.constant([4.0, 5.0, 6.0])
     c1 + c2

<tf.Tensor: id=29, shape=(3,), dtype=float32, numpy=array([5., 7., 9.], dtype=float32)>

[16] c1 - c2

<tf.Tensor: id=31, shape=(3,), dtype=float32, numpy=array([-3., -3., -3.], dtype=float32)>

[17] c1 * c2

<tf.Tensor: id=33, shape=(3,), dtype=float32, numpy=array([ 4., 10., 18.], dtype=float32)>

[18] c1 / c2

<tf.Tensor: id=35, shape=(3,), dtype=float32, numpy=array([0.25, 0.4 , 0.5 ], dtype=float32)>
```

Resultados

Prof. Dr. Razer A. N. R. Montañó - UFPR/SEPT

Tensorflow

56

57

Operações Básicas com Tensores

➤ Possível também com constantes

```
[29] c1 + 3
↳ <tf.Tensor: id=73, shape=(3,), dtype=float32, numpy=array([4., 5., 6.], dtype=float32)>

[30] c1 - 3
↳ <tf.Tensor: id=76, shape=(3,), dtype=float32, numpy=array([-2., -1., 0.], dtype=float32)>

[31] c1 * 3
↳ <tf.Tensor: id=79, shape=(3,), dtype=float32, numpy=array([3., 6., 9.], dtype=float32)>

[32] c1 / 3
↳ <tf.Tensor: id=82, shape=(3,), dtype=float32, numpy=array([0.33333334, 0.66666667, 1.], dtype=float32)>
```

Resultados

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

57

58

Operações Básicas com Tensores

➤ Usando a biblioteca numpy

```
import numpy as np
```

➤ Algumas funções

```
[37] import numpy as np
     np.sqrt(c1)
↳ array([1., 1.4142135, 1.7320508], dtype=float32)

[38] np.square(c1)
↳ array([1., 4., 9.], dtype=float32)

[40] np.log(c1)
↳ array([0., 0.6931472, 1.0986123], dtype=float32)
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

58

59

Operações Básicas com Tensores.

Função	Função	Função
<code>sin(x)</code>	<code>log(x)</code>	<code>exp(x)</code>
<code>cos(x)</code>	<code>log2(x)</code>	<code>exp2(x)</code>
<code>tan(x)</code>	<code>log10(x)</code>	<code>sign(x)</code>
<code>deg2rad(x)</code>	<code>add(x1, x2)</code>	<code>remainder(x1, x2)</code>
<code>rad2deg(x)</code>	<code>multiply(x1, x2)</code>	<code>mod(x1, x2)</code>
<code>around(x)</code>	<code>divide(x1, x2)</code>	<code>minimum(x)</code>
<code>sum(x)</code>	<code>subtract(x1, x2)</code>	<code>maximum(x)</code>
<code>prod(x)</code>	<code>power(x1, x2)</code>	<code>fabs(x)</code>

Fonte: <https://docs.scipy.org/doc/numpy-1.13.0/reference/routines.math.html>

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

59

60

Criar tensores com zeros e uns.

➤ Tensores contendo zeros

```
z = tf.zeros([3, 4])
```

➤ Tensores contendo uns

```
u = tf.ones([2, 3])
```

➤ Resultado

```
[21] z = tf.zeros([3, 4])
z
<tf.Tensor: id=42, shape=(3, 4), dtype=float32, numpy=
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]], dtype=float32)>

[22] u = tf.ones([2, 3])
u
<tf.Tensor: id=46, shape=(2, 3), dtype=float32, numpy=
array([[1., 1., 1.],
       [1., 1., 1.]], dtype=float32)>
```

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

60

61

Strings

► Declaradas como Constantes ou Variáveis

```
str = tf.constant("TensorFlow")
str
```

```
[43] str = tf.constant("TensorFlow")
      str
↳ <tf.Tensor: id=92, shape=(), dtype=string, numpy=b'TensorFlow'>

[44] tf.strings.length(str)
↳ <tf.Tensor: id=94, shape=(), dtype=int32, numpy=10>

[45] tf.strings.unicode_decode(str, "UTF8")
↳ <tf.Tensor: id=99, shape=(10,), dtype=int32, numpy=array([ 84, 101, 110, 115, 111, 1
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

61

62

Strings..

► Tensor de Strings

```
[49] str_vec = tf.constant(["String 1", "String 2", "String 3"])
      str_vec
↳ <tf.Tensor: id=111, shape=(3,), dtype=string, numpy=array([b'String 1', b'String 2',
```

► Laço pelas Strings

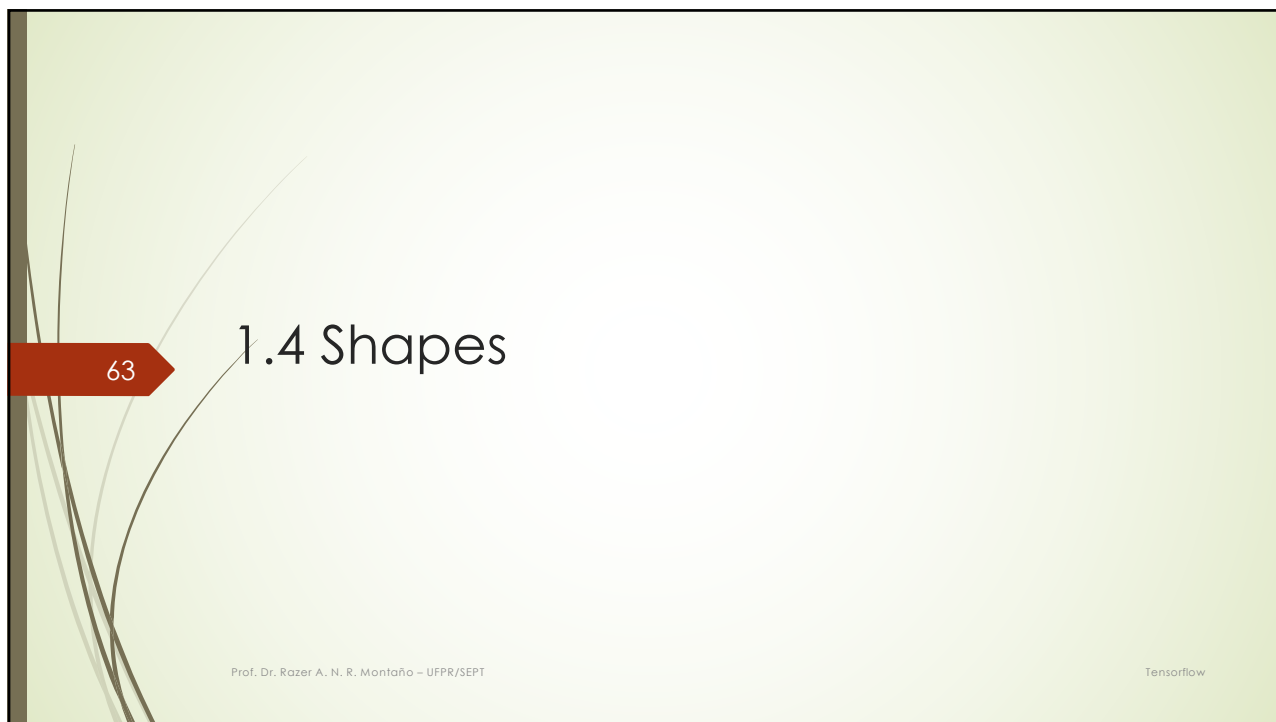
```
[50] for it in str_vec:
      print(it)

↳ tf.Tensor(b'String 1', shape=(), dtype=string)
↳ tf.Tensor(b'String 2', shape=(), dtype=string)
↳ tf.Tensor(b'String 3', shape=(), dtype=string)
```

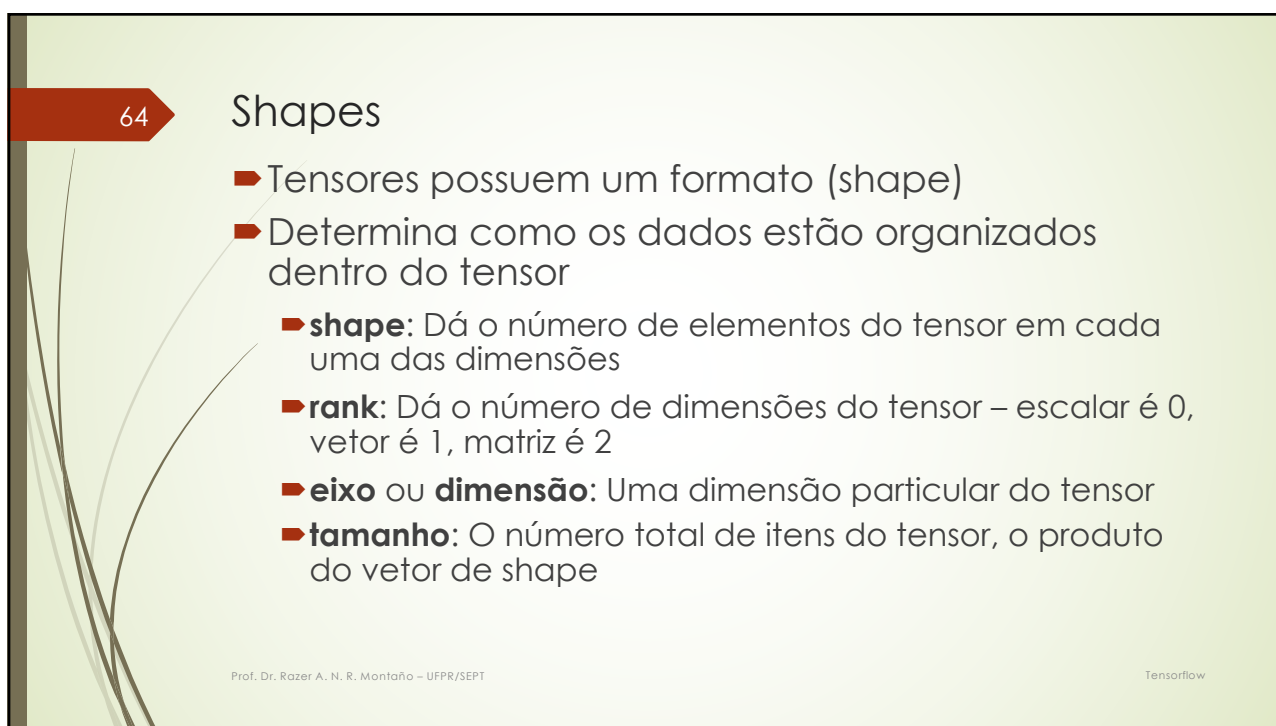
Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

62



63



64

65

Shapes

```
import tensorflow as tf

rank_2_tensor = tf.constant([[2, 2], [3, 3], [4, 4]],
                             dtype=tf.float16)
print(rank_2_tensor.shape)
print(rank_2_tensor.ndim)
print(rank_2_tensor.dtype)
```

```
↳ (3, 2)
   2
   <dtype: 'float16'>
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

65

66

Shapes: Obter dados

```
import tensorflow as tf

rank_2_tensor = tf.constant([[2, 2], [3, 3],
                              [4, 4]], dtype=tf.float16)

print(rank_2_tensor[1, 1])
print(rank_2_tensor[1])
print(rank_2_tensor[:, 1])
print(rank_2_tensor[-1])
```

```
↳ tf.Tensor(3.0, shape=(), dtype=float16)
   tf.Tensor([3. 3.], shape=(2,), dtype=float16)
   tf.Tensor([2. 3. 4.], shape=(3,), dtype=float16)
   tf.Tensor([4. 4.], shape=(2,), dtype=float16)
```

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPT

Tensorflow

66

67

Shapes: Uso de reshape

```
import tensorflow as tf

var_x = tf.Variable(tf.constant([[1], [2], [3]]))
print(var_x.shape)

↳ (3, 1)

print(var_x)

↳ <tf.Variable 'Variable:0' shape=(3, 1) dtype=int32, numpy=
  array([[1],
        [2],
        [3]], dtype=int32)>

reshaped = tf.reshape(var_x, [1, 3])
print(reshaped.shape)

↳ (1, 3)

print(reshaped)

↳ tf.Tensor([[1 2 3]], shape=(1, 3), dtype=int32)
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

67

68

Shapes: Uso de reshape

- Pode-se alterar o shape de um tensor
- Usa-se o comando:

```
reshaped = tf.reshape(tensor, [<novo shape>])
```

- Onde:

- **tensor**: é o tensor sendo remodelado
- **[<novo shape>]**: são as novas dimensões. Se alguma delas for -1, o TF automaticamente calcula quantas são necessárias

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

68

69

Shapes: Uso de reshape

```
import tensorflow as tf

rank_2_tensor = tf.constant([[2, 2], [3, 3], [4, 4]],
                             dtype=tf.float16)
print(rank_2_tensor)

reshaped = tf.reshape(rank_2_tensor, [1, 6])
print(reshaped)
```

```
tf.Tensor(
[[2. 2.]
 [3. 3.]
 [4. 4.]], shape=(3, 2), dtype=float16)
tf.Tensor([[2. 2. 3. 3. 4. 4.]], shape=(1, 6), dtype=float16)
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

69

70

Shapes: Uso de reshape

```
import tensorflow as tf

rank_3_tensor = tf.constant(
    [ [ [2, 1], [2, 1] ],
      [ [3, 1], [3, 1] ],
      [ [4, 1], [4, 1] ] ] )
print(rank_3_tensor)
```

```
tf.Tensor(
[[[2 1]
  [2 1]]

 [[3 1]
  [3 1]]

 [[4 1]
  [4 1]]], shape=(3, 2, 2), dtype=int32)
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

70

71

Shapes: Uso de reshape

```
reshaped = tf.reshape(rank_3_tensor, [2, 3, -1])
print(reshaped)
```

```
tf.Tensor(
[[[2 1]
  [2 1]
  [3 1]]

 [[3 1]
  [4 1]
  [4 1]]], shape=(2, 3, 2), dtype=int32)
```

```
reshaped = tf.reshape(rank_3_tensor, [2, 1, -1])
print(reshaped)
```

```
tf.Tensor(
[[[2 1 2 1 3 1]
  [3 1 4 1 4 1]]], shape=(2, 1, 6), dtype=int32)
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

71

72

Shapes: Uso de reshape

```
reshaped = tf.reshape(rank_3_tensor, [2, -1])
print(reshaped)
```

```
tf.Tensor(
[[2 1 2 1 3 1]
 [3 1 4 1 4 1]], shape=(2, 6), dtype=int32)
```

```
reshaped = tf.reshape(rank_3_tensor, [-1])
print(reshaped)
```

```
tf.Tensor([2 1 2 1 3 1 3 1 4 1 4 1], shape=(12,), dtype=int32)
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

72

73

Shapes: Uso de reshape..

```
reshaped = tf.reshape(rank_3_tensor, [1, -1, 2])  
print(reshaped)
```

```
tf.Tensor(  
[[[2 1]  
 [2 1]  
 [3 1]  
 [3 1]  
 [4 1]  
 [4 1]]], shape=(1, 6, 2), dtype=int32)
```

```
reshaped = tf.reshape(rank_3_tensor, [2, 1, -1, 2])  
print(reshaped)
```

```
tf.Tensor(  
[[[[2 1]  
   [2 1]  
   [3 1]]]  
  
 [[3 1]  
  [4 1]  
  [4 1]]], shape=(2, 1, 3, 2), dtype=int32)
```

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

73

1.5 Lazy Execution x Eager Execution

Prof. Dr. Razer A. N. R. Montaña - UFPR/SEPT

Tensorflow

74

75

Lazy Execution x Eager Execution

► Eager Execution

► Comportamento esperado

```
x = [[ 2. ]]
m = tf.matmul(x, x)
print("Resultado: {}".format(m))
```

► Execução é imediata

► **m** contém um tensor com o resultado

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

75

76

Lazy Execution x Eager Execution

► Lazy Execution

► Modo Grafo (*graph mode*)

► Cria um grafo de cálculo (**tf.Graph**)

```
x = [[ 2. ]]
m = tf.matmul(x, x)
print("Resultado: {}".format(m))
```

► Execução é adiada

► **tf.matmul** adiciona uma operação ao grafo (**tf.Operation**)

► **m** contém uma referência para um nodo de computação

► São feitas otimizações no grafo de cálculo

► Execução do cálculo é adiada

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

76

77

Lazy Execution x Eager Execution

- Versão 1.x do TF
 - Default era LAZY EXECUTION
 - Versões 1.3 e 1.4 possuíam a opção de EAGER EXECUTION (não por default)
- Versão 2.x do TF
 - Por default faz EAGER EXECUTION
 - Pode-se executar qualquer conjunto de operações sem construir o grafo
 - Perde em desempenho (sem otimizações)
- Para se construir o grafo para posterior execução usa-se a anotação
 - **@tf.function**
 - Faz LAZY EXECUTION

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

77

78

Lazy Execution x Eager Execution

- Funções anotadas com **@tf.function**
 - Pode ser aplicado a funções e a métodos de classes
 - Cria um grafo do Tensorflow a partir da função
 - Os nós são as operações dentro da função
 - Pode ser executada na GPU e TPU (Tensor Processing Unit – Google)

Prof. Dr. Razer A. N. R. Montão – UFPR/SEPT

Tensorflow

78

79

Lazy Execution x Eager Execution

```
@tf.function
def square(x):
    return (x * x)

a = tf.constant([10, 20])

res = square(a)

print(res)
```

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

79

80

Lazy Execution x Eager Execution

- Usa grafos para controlar o fluxo de dados
- Grafo de operações por onde os dados (tensores) fluem
 - Dependências entre operações
- Todas as computações podem ser representadas por um grafo
 - No caso de execução de funções **@tf.function**
 - Objetos **tf.Operation** : representam unidades de operações
 - Objetos **tf.Tensor** : representam os dados

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

80

81

Lazy Execution x Eager Execution

➤ Eager Execution

- Permite uso de código Python
- **if, else, while**, etc

➤ Escrever código em modo grafo (Lazy Execution) não é simples

- Código Python precisa ser traduzido

for/while -> tf.while_loop

if -> tf.cond

for _ in dataset -> dataset.reduce

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

81

82

Lazy Execution x Eager Execution

➤ Módulo AutoGraph

- Faz a transformação de Python para TF
- Automaticamente aplicado em função **@tf.function**
- Permite programar diretamente em Python

Prof. Dr. Razer A. N. R. Montão - UFPR/SEPT

Tensorflow

82

83

Lazy Execution x Eager Execution..

- Quando usar LAZY EXECUTION
 - Aumentar o desempenho do cálculo: otimizações
 - Operações que não são naturalmente feitas em GPU/TPU
 - Muitas operações simples: bom ganho de desempenho
- Quando não usar
 - Não colocar em todas as funções
 - Quando se quer debbugar
 - Poucas operações custosas : pouco ganho de desempenho

Prof. Dr. Razer A. N. R. Montaña – UFPR/SEPTTensorflow

83