



# THE DEVELOPER'S CONFERENCE

## Trilha Computação Cognitiva

Iago Elias e Alexandre Nakahara

Criando uma API de visão computacional  
para avaliar desempenho de plantações  
agrícolas com Oracle Cloud

# Break New Ground

Break New Ground

# Criando uma API de visão computacional para avaliar desempenho de plantações agrícolas com Oracle Cloud

Alexandre Nakahara, Oracle  
Iago Elias, bitHarvest / Visio.ai

Julho, 2019

# Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

ORACLE®

**TRANSFORMAR**

O MUNDO EMPODERANDO AS

**PESSOAS**

P O R M E I O D A

**INOVAÇÃO**

3



**Technovation –  
2017/2018/2019**



**Health AI Hackathon – School of AI  
03/2019**



**Maratona de Programação – Centro  
Paula Souza (ETEC) – 06/2019**



**PROA Coins – Instituto  
PROA – 05/2018**



**Sancathon – Future Farms  
04/2019**



**Recrutamento GenO – 05/2019**









20.000  
77%

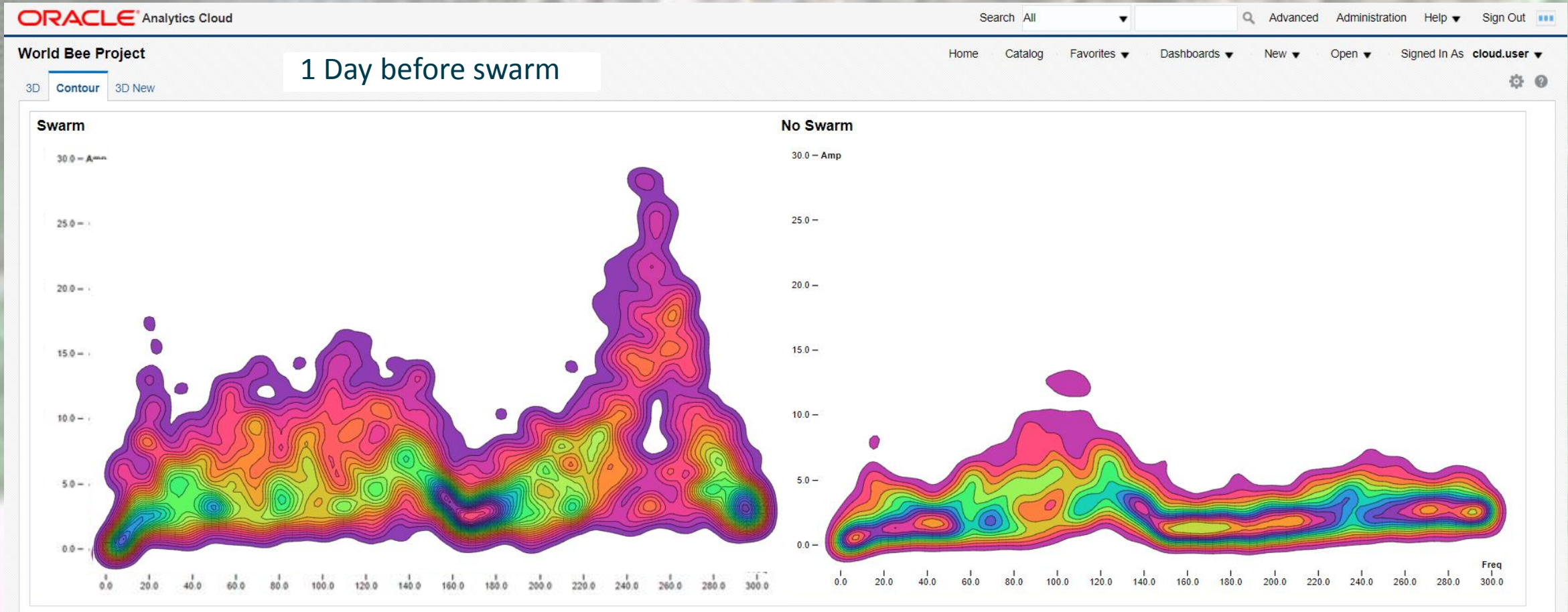
25% EU ↓  
40% US ↓  
45% UK ↓

The World Bee Project 

# Oracle & The World Bee project



# Analytics confirms low frequency “warble”



bees: 67%





Iago Elias  
bitHarvest / Visio.ai

ORACLE®

Copyright © 2019, Oracle and/or its affiliates. All rights reserved. |

Quem somos?

# BitHarvest

## BitHarvest

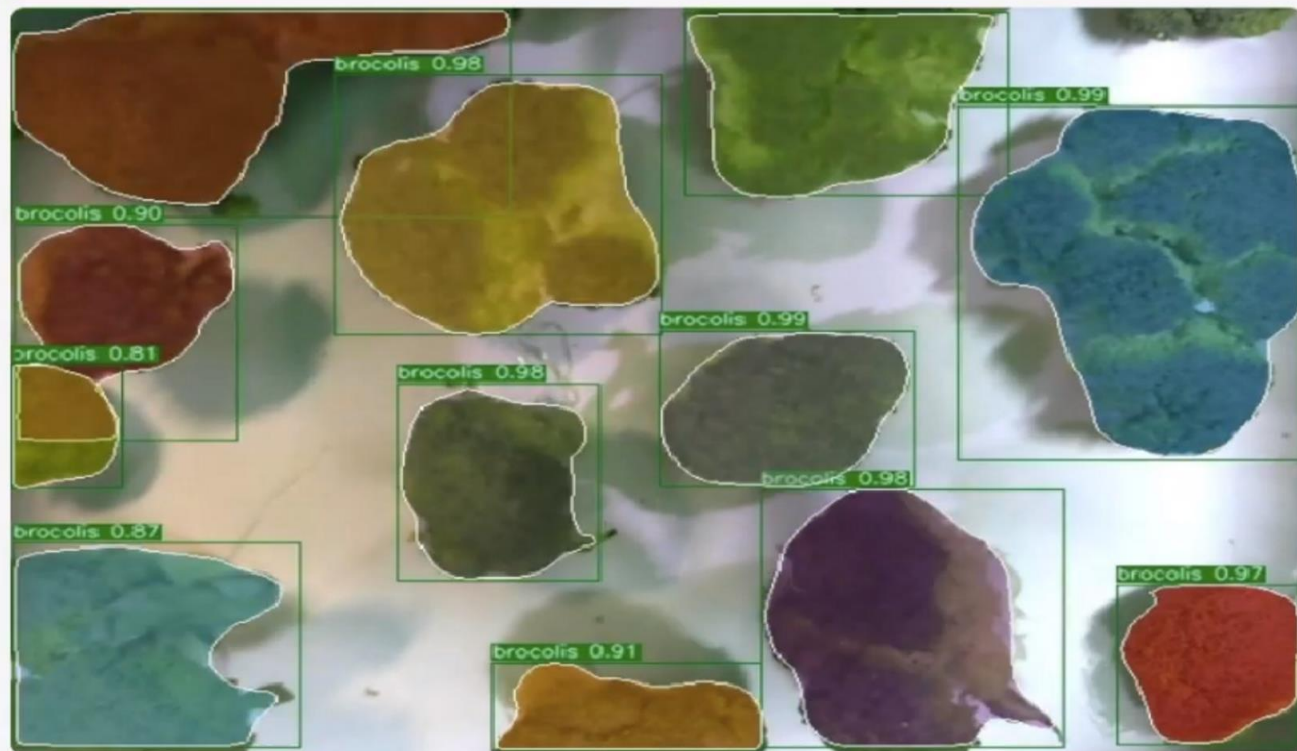
🔍 Busca Avançada

📄 Exportar Dados

🗑️ Ver Gavetas

⚙️ Configurações

## Stream



API de  
Inferência?



# API

(Interface de Programação de Aplicações)

# REST API

NodeJS (Express, Hapi)  
Python (Django, Flask,  
Starlette+Uvicorn)

# Onde utilizar

- Aplicações em tempo real
- Quando se tem embarcados no sistema
- Arquitetura de microsserviços

# Onde (talvez) não utilizar

- Aplicações cujo processamento pode ser atrasado

# Inferência

# Soluções

 TensorFlow

 mxnet



Caffe

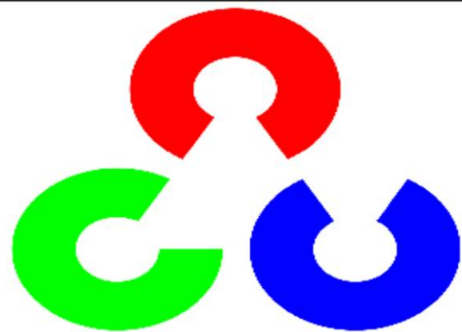
 torch

PYTORCH

theano

 Caffe2

versus

  
OpenCV

OpenVINO™

**Por que o OpenCV?**

## Vantagens:

- Desempenho melhor (Hardware da Intel)
- Agnóstico de Framework e Hardware (FPGA, CPU, GPU, MOVIDIUS VPU)
- Atualizado com frequência
- OpenVINO

The logo for OpenVINO, featuring the word "OpenVINO" in a bold, black, sans-serif font. The "O" at the beginning and the "O" at the end are stylized, each containing a purple circular icon with a white crescent shape inside. A small "TM" trademark symbol is located to the upper right of the final "O".

OpenVINO™

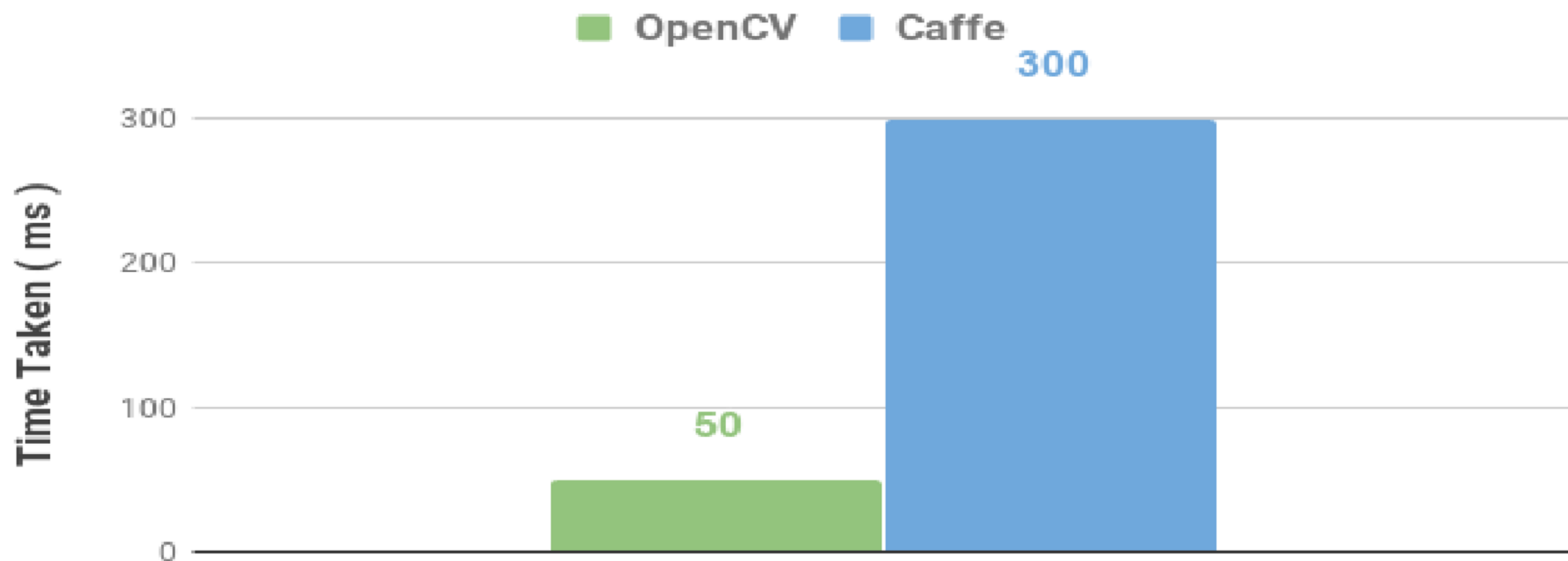
## Desvantagens

- GPU's da NVIDIA rodando em **OpenCL**
- Não efetua treinamento
- A aplicação de redes pré-treinadas demanda mais conhecimento



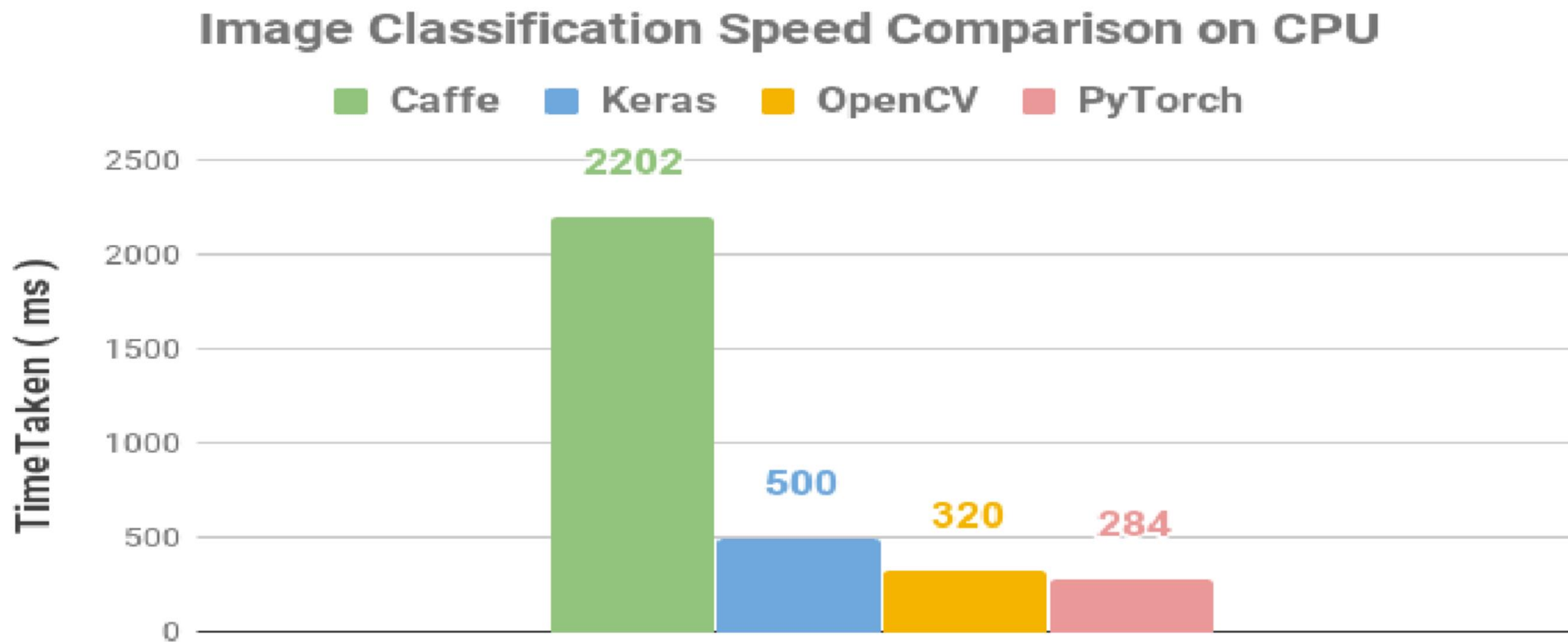
# Por que o OpenCV?

Object Tracking Speed Comparison on CPU



**OpenCV is 6X Faster**

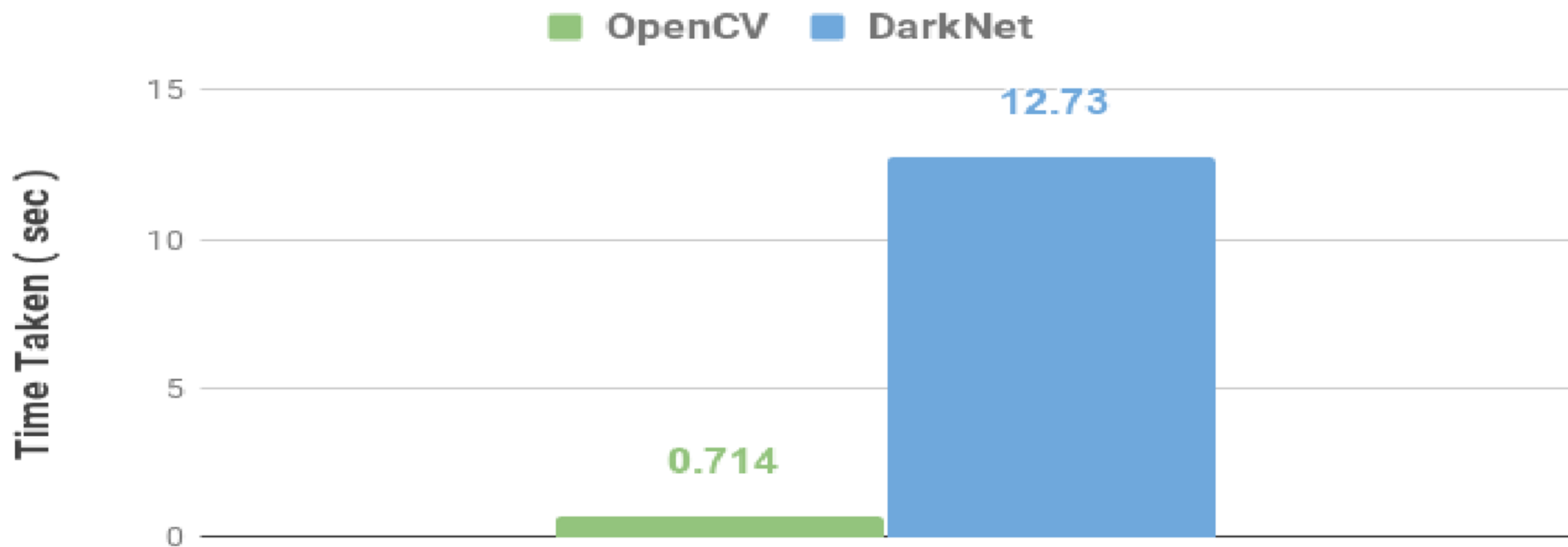
# Por que o OpenCV?



**OpenCV is faster by 7X ( Caffe ) and 1.5X ( Keras )**

# Por que o OpenCV?

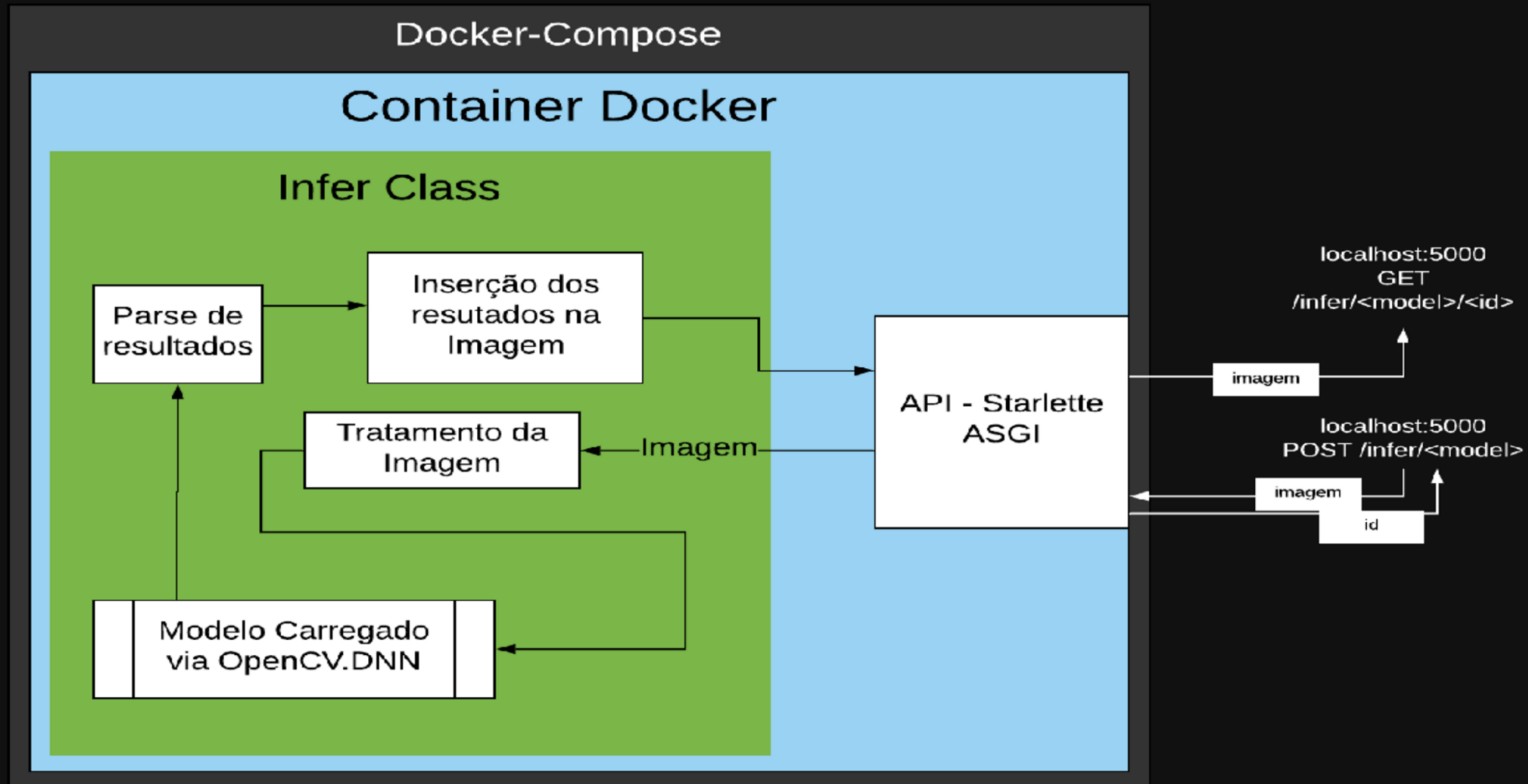
Object Detection Speed Comparison on CPU



**OpenCV is 18X Faster**

**Implementação**

# Arquitetura da API



## Compute - GPU Instances

[Pricing calculator](#) →

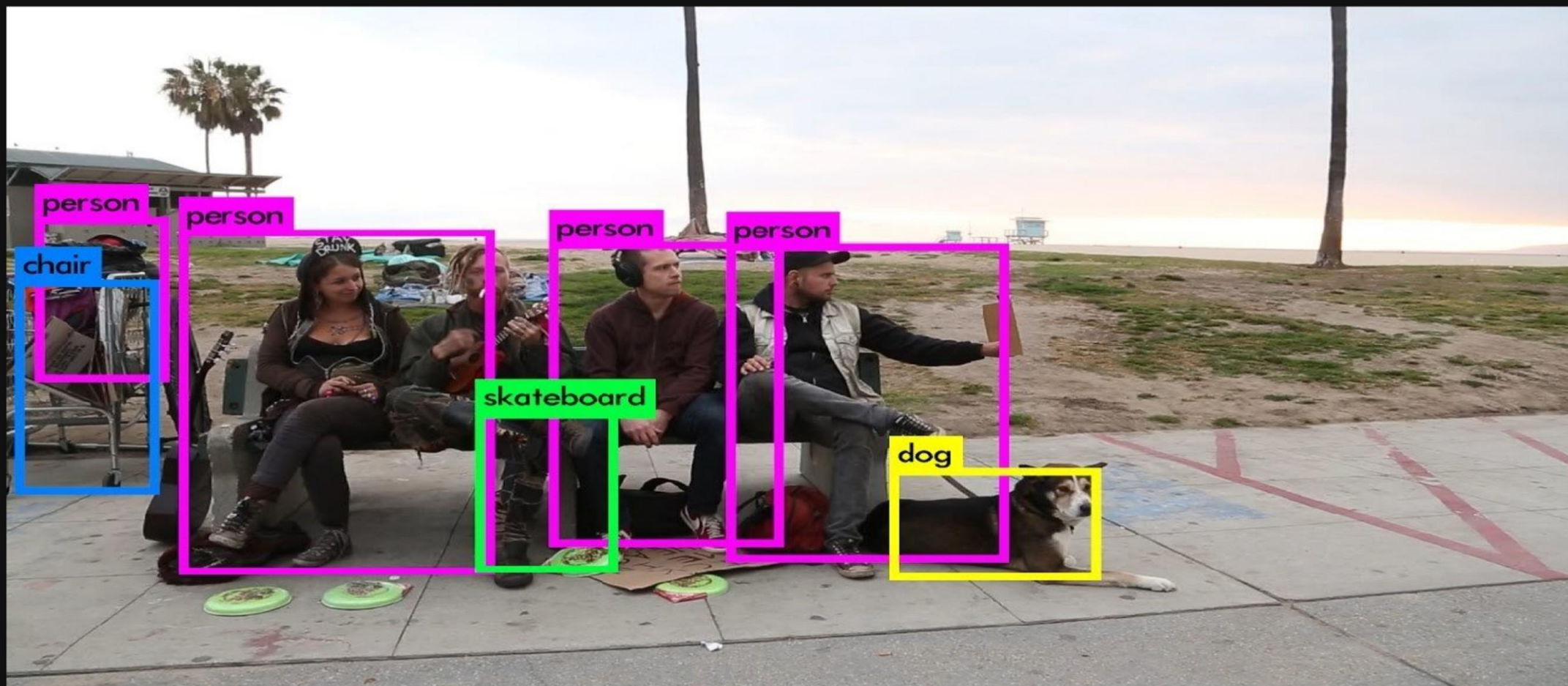
Instance Type	Shape	Pay as You Go (GPU Per Hour)	Service Includes per Month
Pascal Virtual Machine GPU	VM.GPU2.1	R\$5.0746	<ul style="list-style-type: none"><li>• GPU: 1x P100</li><li>• OCPU: 12</li><li>• Memory: 72 GB</li><li>• Network: 8 Gbps</li><li>• Storage: Up to 1 PB of remote Block Volumes</li></ul>
Pascal Bare Metal GPU	BM.GPU2.2	R\$5.0746	<ul style="list-style-type: none"><li>• GPU: 2x P100</li><li>• OCPU: 28</li><li>• Memory: 192 GB</li><li>• Network: 2 x 25 Gbps</li><li>• Storage: Up to 1 PB of remote Block Volumes</li></ul>
Volta Virtual Machine GPU	VM.GPU3.1	R\$11.7413	<ul style="list-style-type: none"><li>• GPU: 1x V100</li><li>• OCPU: 6</li><li>• Memory: 90 GB</li><li>• Network: 4 Gbps</li></ul>

R\$5 reais a hora



**Classe de Detecção**

# YOLO





# Carregar e Configurar

```
1 net = cv2.dnn.readNetFromDarknet(cfg, weights)
2 net.setPreferableBackend(cv2.dnn.DNN_BACKEND_DEFAULT)
3 net.setPreferableTarget(cv2.dnn.DNN_TARGET_OPENCL)
```

## Configuração (cfg)

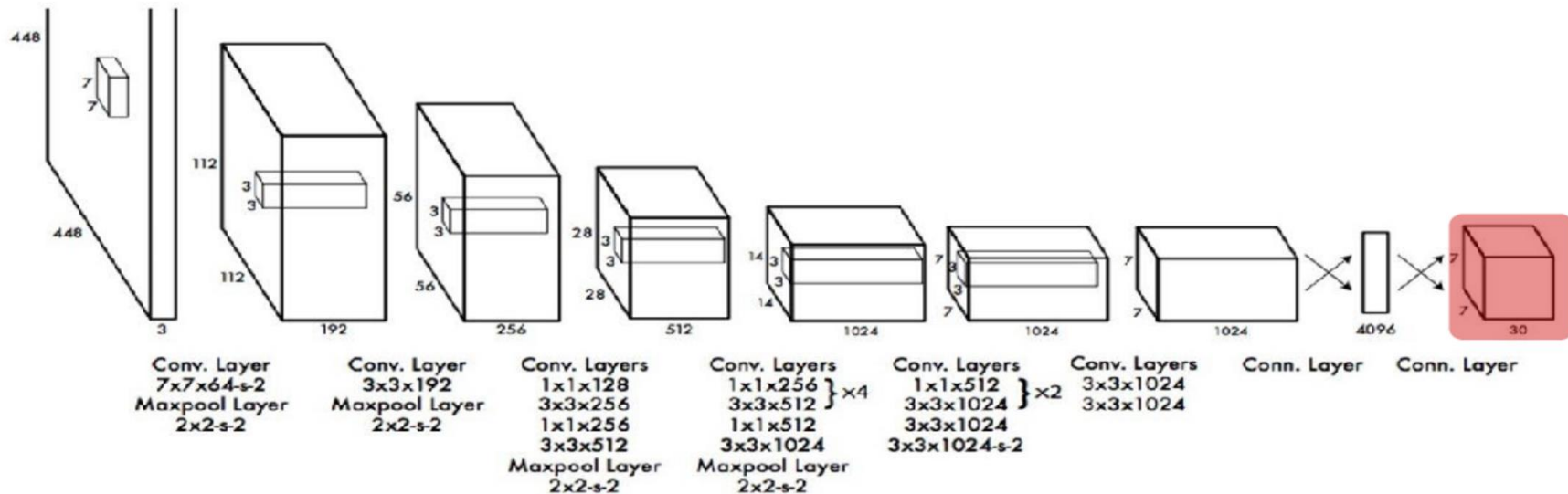
```
1 [convolutional]
2 filters=64
3 size=7
4 ...
5
6 [maxpool]
7 size=2
8 stride=2
9
10 [convolutional]
11 filters=192
12 ...
13
14 [maxpool]
15 size=2
16 stride=2
17
18 [convolutional]
19 filters=128
20 ...
```

## Pesos



# Capturar a Referência das Camadas de Output

```
1 ln = net.getLayerNames()
2 ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]
```



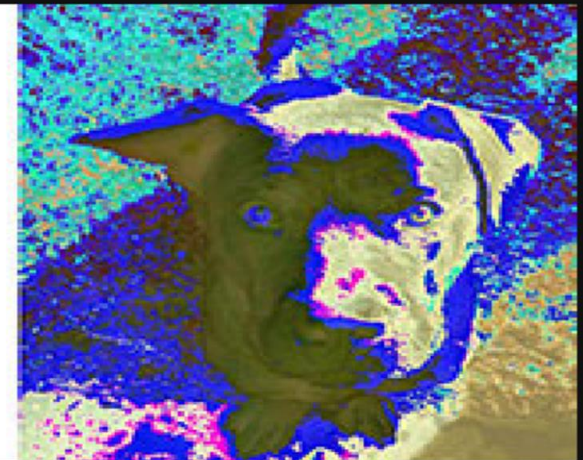
**Figure 3: The Architecture.** Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating  $1 \times 1$  convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution ( $224 \times 224$  input image) and then double the resolution for detection.

# Tratar a Imagem e gerar o output

```
1 blob = cv2.dnn.blobFromImage(img, 1.0/255, (416, 416),  
2                                     swapRB=True, crop=False)  
3 net.setInput(blob)  
4 outs = net.forward(getOutputLayers(net))
```



R=124.96  
- G=115.97 =  
B=106.13



# Parse dos resultados:

detecção

scores/classe

```
1 [ [0.04615184 0.05549661 0.46020883 ... 0. 0. 0. ]
2 [0.04845504 0.03119141 0.3057728 ... 0. 0. 0. ]
3 [0.05163545 0.04100422 0.81520915 ... 0. 0. 0. ]
4 ...
5 [0.95453477 0.95826906 0.42958248 ... 0. 0. 0. ]
6 [0.9574697 0.9671171 0.29094929 ... 0. 0. 0. ]
7 [0.963808 0.9633634 0.86105084 ... 0. 0. 0. ] ]
8 [ [0.0196179 0.02369402 0.04342464 ... 0. 0. 0. ]
9 [0.01811114 0.01925643 0.39157847 ... 0. 0. 0. ]
10 [0.02052197 0.01649838 0.06510343 ... 0. 0. 0. ]
11 ...
12 [0.9750362 0.9769141 0.04074052 ... 0. 0. 0. ]
13 [0.9818321 0.9775298 0.3982786 ... 0. 0. 0. ]
14 [0.98003006 0.9840621 0.07373534 ... 0. 0. 0. ] ]
15 [ [0.00570609 0.00743651 0.00960583 ... 0. 0. 0. ]
16 [0.00791775 0.01174264 0.01763124 ... 0. 0. 0. ]
17 [0.00960346 0.00898112 0.17557846 ... 0. 0. 0. ]
18 ...
19 [0.98920095 0.9892078 0.01725708 ... 0. 0. 0. ]
20 [0.9899234 0.987255 0.01848863 ... 0. 0. 0. ]
21 [0.9878559 0.99048084 0.15287374 ... 0. 0. 0. ] ]
```

# Parse dos resultados:

índice 0 = classe pessoa



detecção

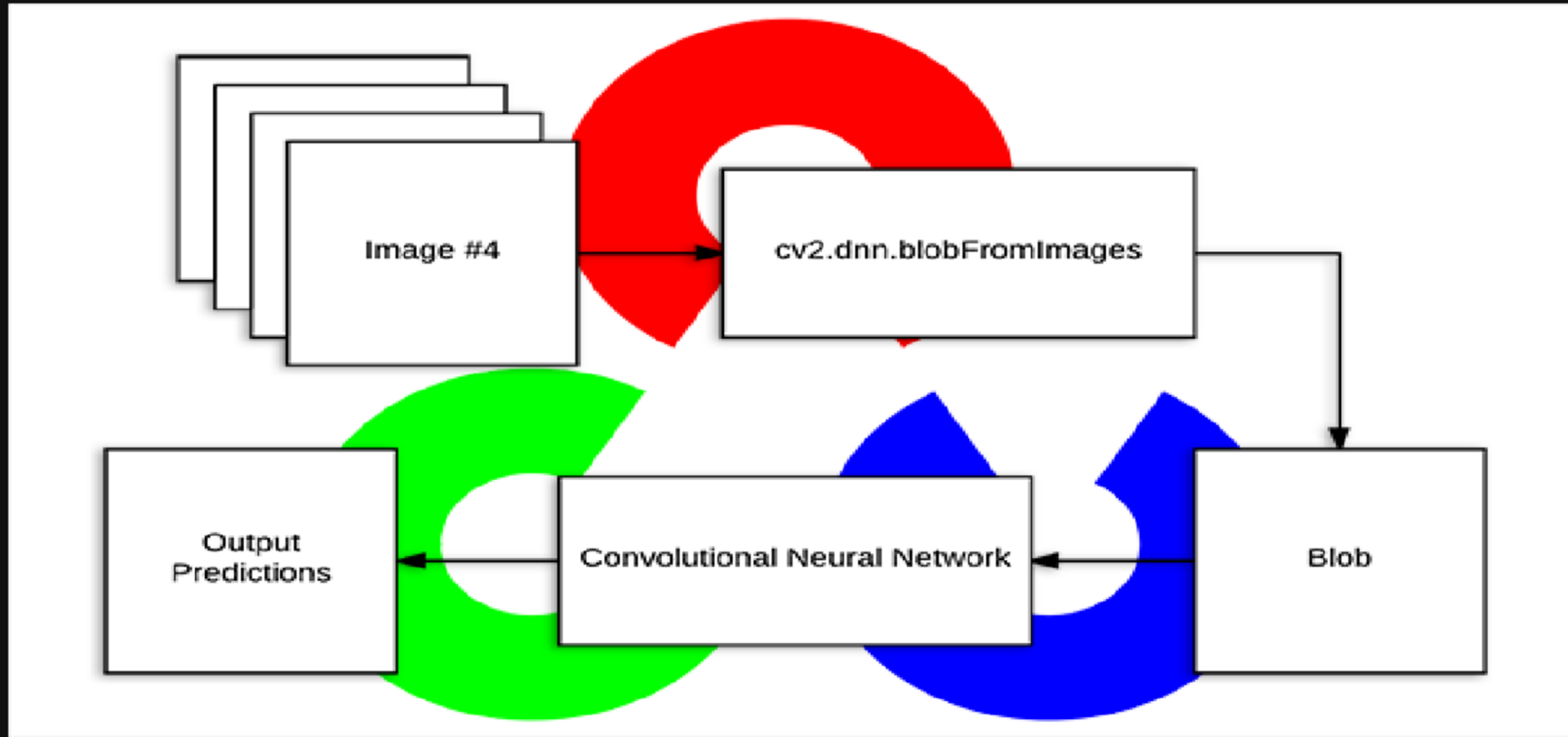
1	[	4.6151835e-02	5.5496614e-02	4.6020883e-01	1.2107838e-01	3.5152695e-07	
2		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
3		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
4		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
5		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
6		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
7		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
8		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
9		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
10		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
11		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
12		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
13		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
14		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
15		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
16		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	
17		0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	]

scores/classe

# Parse dos resultados:

```
1 for out in outs:
2
3     for detection in out:
4         scores = detection[5:]
5         max_score = np.argmax(scores)
6         score = scores[max_score]
7         if score >= threshold:
8             centerx, centery, width, height = tuple(int(val*multiplier) for
9             x, y = int(centerx - width/2), int(centery - height/2)
10            objects.append((x,y), (x+width, y+height), names[max_score]))
```

# Processo



# API

## POST

```
1 @api.route('/infer/{key}', methods=['POST'])
2 async def inf(request):
3     key      = request.path_params['key']
4     payload = await request.body()
5
6     image    = loaded_models[key].decode(payload)
7     bboxes   = loaded_models[key].detect(image)
8     print (bboxes, file=sys.stderr)
9     image    = loaded_models[key].insert_bboxes(image, bboxes)
10    # res      = loaded_models[key].encode_image(image)
11    unique_filename = str(uuid.uuid4())
12    cv2.imwrite(f'files/{unique_filename}.jpg', image)
13
14    return Response(f'{unique_filename}', status_code=200)
```



# API

## GET

```
1 @api.route('/file/{id}', methods=['GET'])
2 async def get_image(request):
3     key = request.path_params['id']
4     return FileResponse(f'files/{key}.jpg', status_code=200)
```

Demonstração

[bit.do/tdc\\_api](https://bit.do/tdc_api)

Repositório Github:

[dinthea/infer\\_service](https://github.com/dinthea/infer_service)

Testes no Jupyter

[bit.do/jupyter\\_tdc](https://bit.do/jupyter_tdc)

Contato

[iagoelifa@gmail.com](mailto:iagoelifa@gmail.com)

[iago@visio.ai](mailto:iago@visio.ai)

# Agenda

ORACLE®

TDC

## Autonomous Database para Dev

Não se preocupe mais com a infraestrutura que sustenta seu banco de dados, e sem downtime!

9h30 11h30 13h30 15h30 17h30

## K8s em Oracle Cloud

Use Kubernetes e Docker em um ambiente de alta disponibilidade na nuvem!

10h30 12h30 14h30 16h30

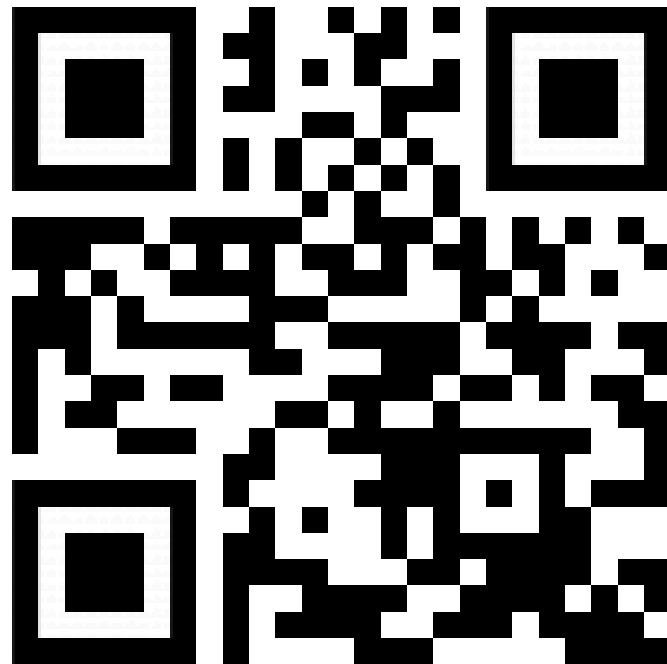


[oracle.com/goto/tdcsp](https://oracle.com/goto/tdcsp)

Vagas limitadas a 6 participantes por ordem de chegada

Solicite um ambiente de trial antes de participar do evento

**Solicite um ambiente de trial antes de participar de nossos laboratórios práticos!**



**<http://oracle.com/goto/tdcsp>**

# Break New Ground