EMBARCADOS EXPERIENCE

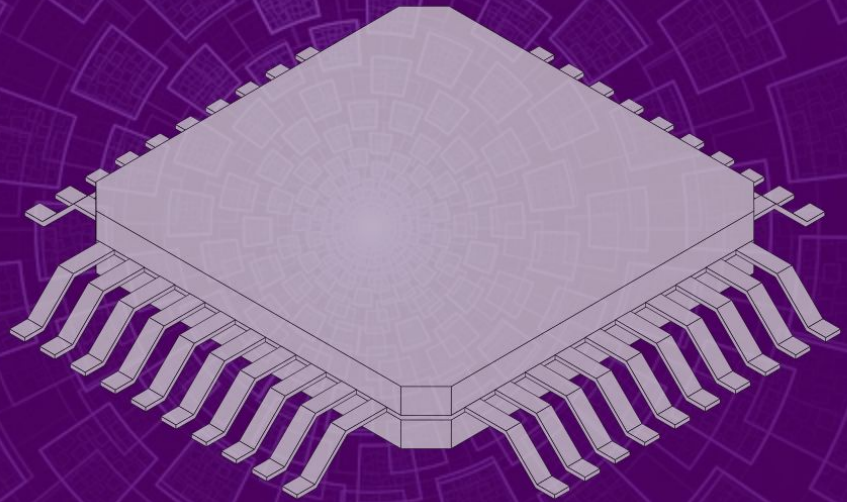
2020

Patrocínio Ouro

MOUSER ELECTRONICS

Realização

EMBARCADOS



28 de Novembro

Evento online com muito networking
e troca de conhecimento.

EMBARCADOS EXPERIENCE 2020

Patrocínio Ouro

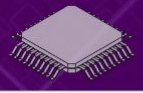
MOUSER ELECTRONICS

Realização

EMBARCADOS

Embarcando Deep Learning: Nano GPUs e TinyML

Marcelo Pias



Sobre o Marcelo...

PhD in Ciência da Computação, University College London

Post-doc estágios na INTEL Research (UK), University of Cambridge

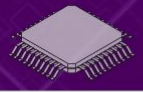
Cambridge University Spin-off: IoT Data Analytics

Professor Adjunto na Universidade Federal do Rio Grande (Rio Grande)

NVIDIA DEEP LEARNING INSTITUTE UNIVERSITY AMBASSADOR

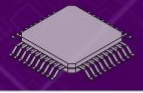
EMBRAPII UE - iTec/FURG | CENTRO DE ROBÓTICA E CIÊNCIA DE DADOS

Hub de Inovação em **Inteligência Artificial**

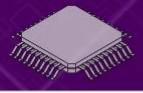


Tópicos hoje

- Aprendizado de máquina profundo (*Deep Learning*)
- Implantação de modelos (*deployment*)
- Modelo computacional CPU versus GPU
- Embarcando modelos em Nano GPUs (NVIDIA Jetson Nano)
- TinyML: modelos embarcados em CPUs



Aprendizado Profundo (*Deep Learning*)



Aplicações de *Deep Learning*

Reconhecimento de Fala

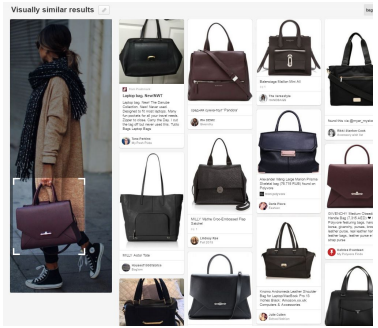
Sistemas de Recomendação

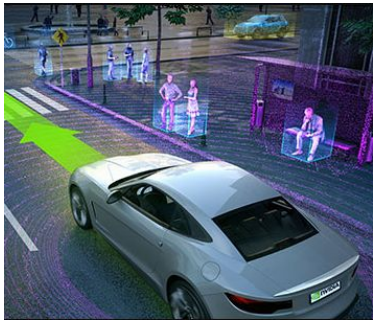
Veículos Autônomos

Reconhecimento de objeto em tempo-real

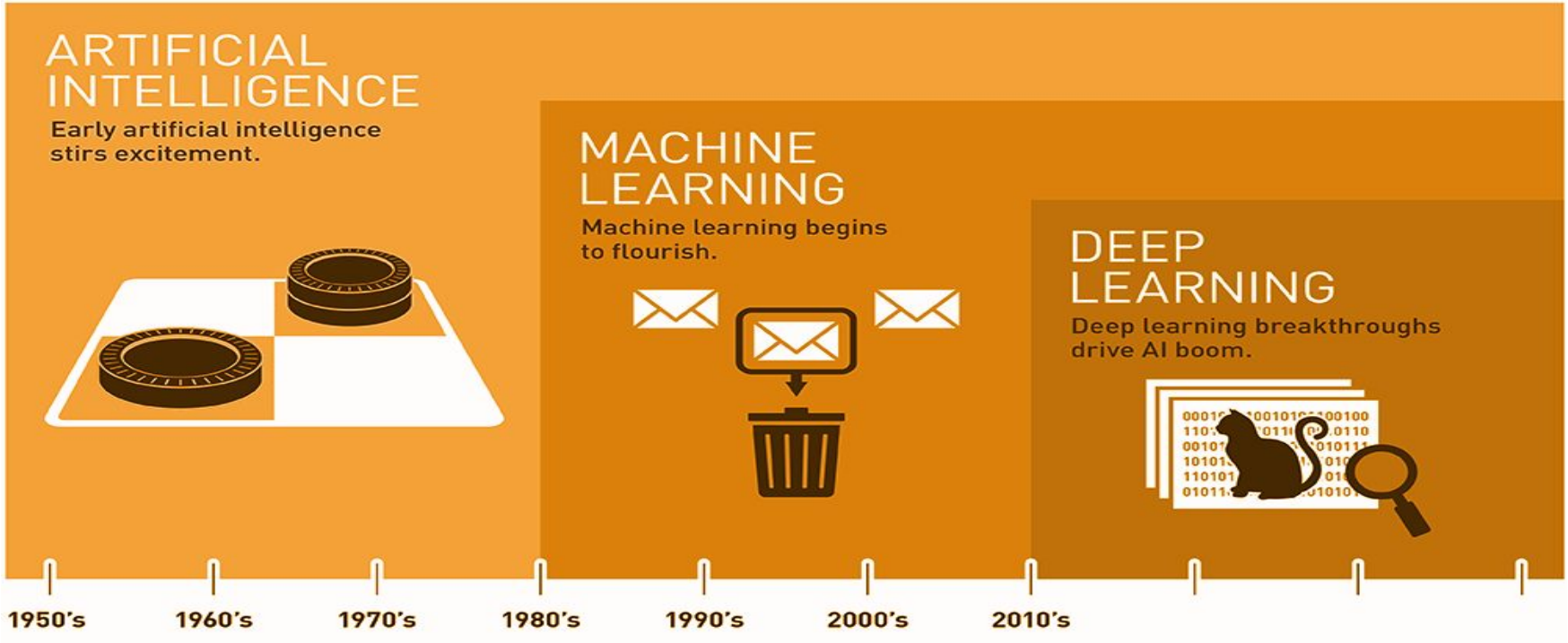
Tradução de línguas em tempo-real

e muito mais...

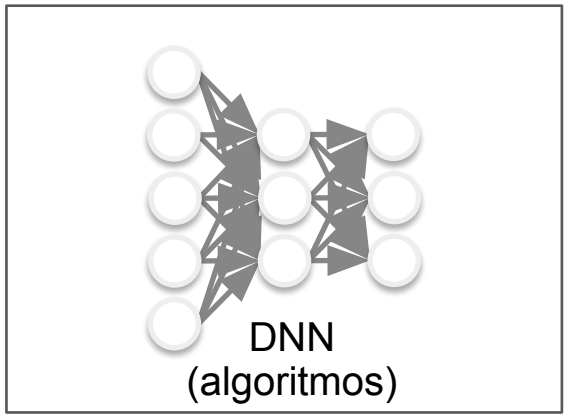




Atingindo objetivos complexos

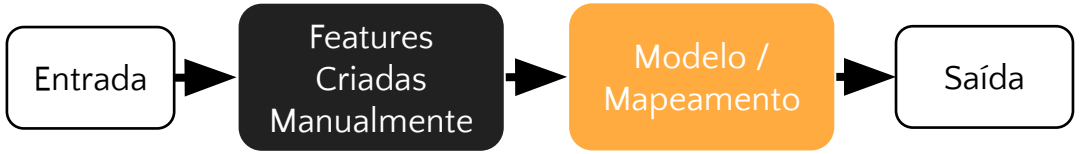


“Tempestade Perfeita”

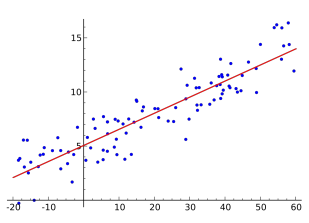


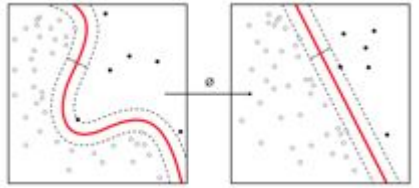
Fluxo de trabalho (diferenças)

Aprendizado de Máquina “Clássico” [1990 : hoje]



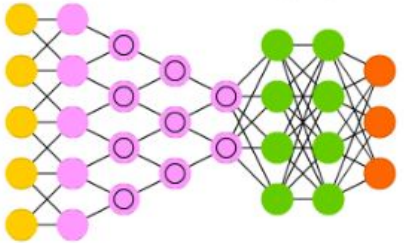
Exemplos [Regressão e SVMs]



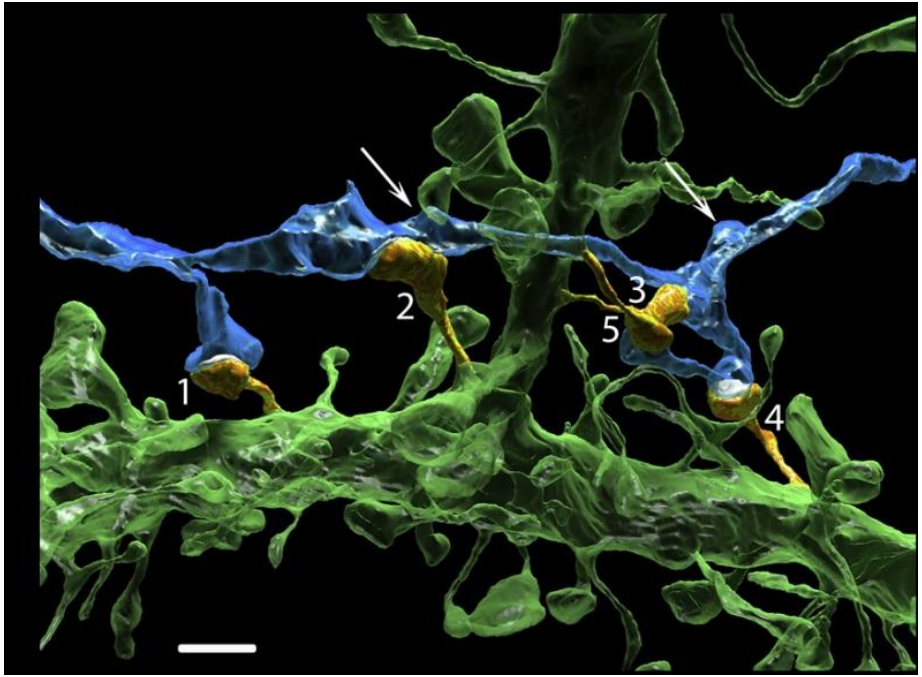


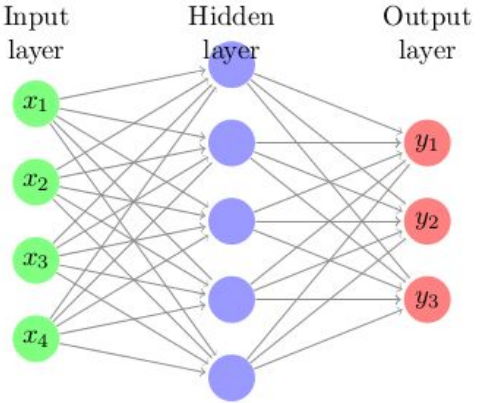
Aprendizado Profundo / Fim-a-fim[2012 : hoje]

Exemplo [Conv Net]



Inspiração Biológica





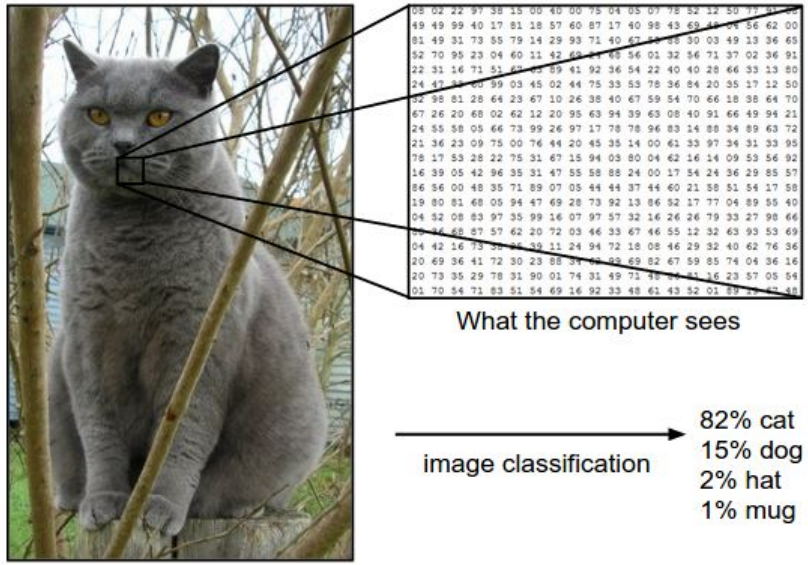
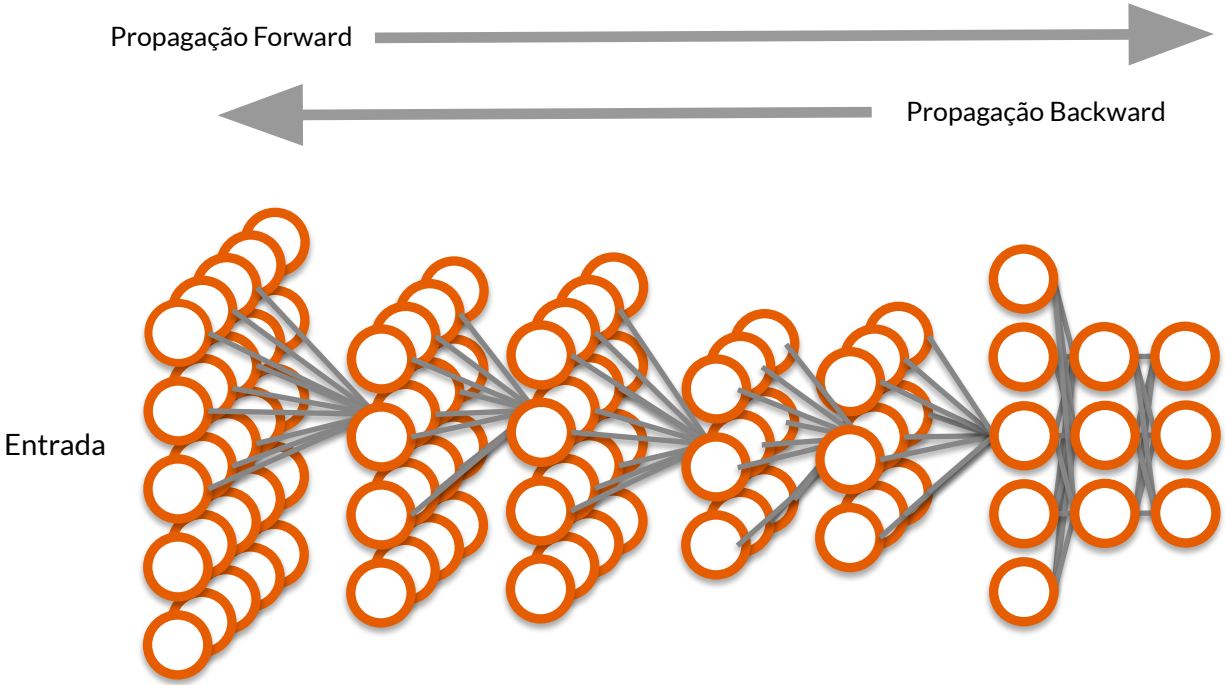


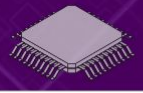
Image from the Stanford CS231 Course

Deep Learning - Treinamento



Processo

- Propagação “Forward” resulta em um rótulo estimado/previsto para cada imagem de treinamento
- Função de perda (erro) usada para calcular a diferença entre um rótulo conhecido e o rótulo previsto para cada imagem
- Pesos são ajustados durante a propagação “backward”
- Repetir o processo



Embarcando Modelos (Deployment)

Deep Learning - Abordagem

Treinamento:

Erros

DNN

Cachorro
Gato
Rinoceronte

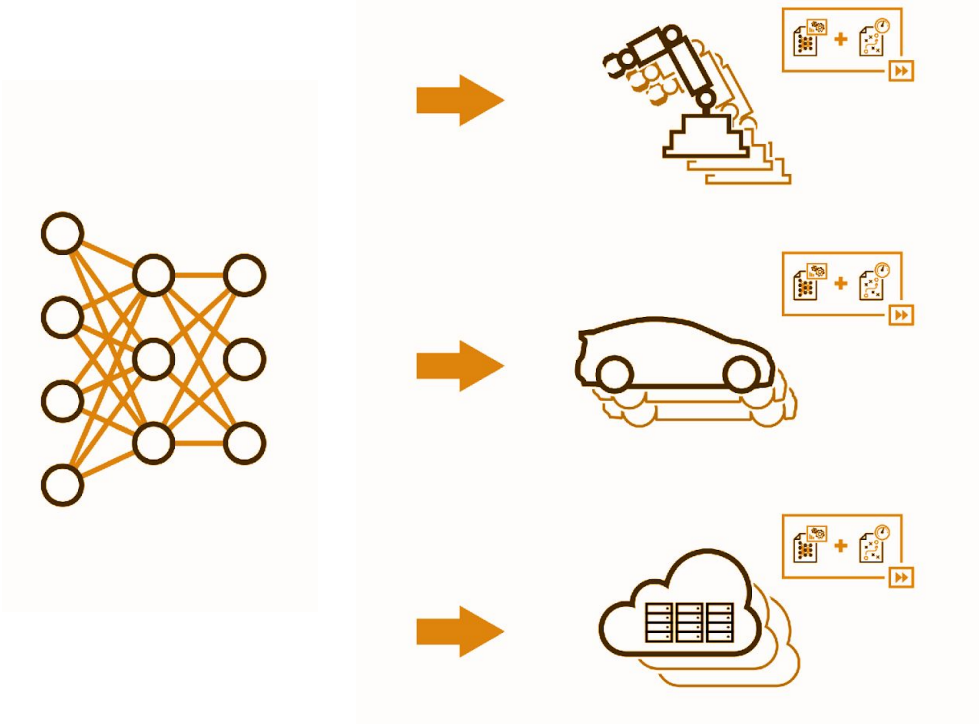
Deploy (implantação/embarcado):

DNN

Cachorro

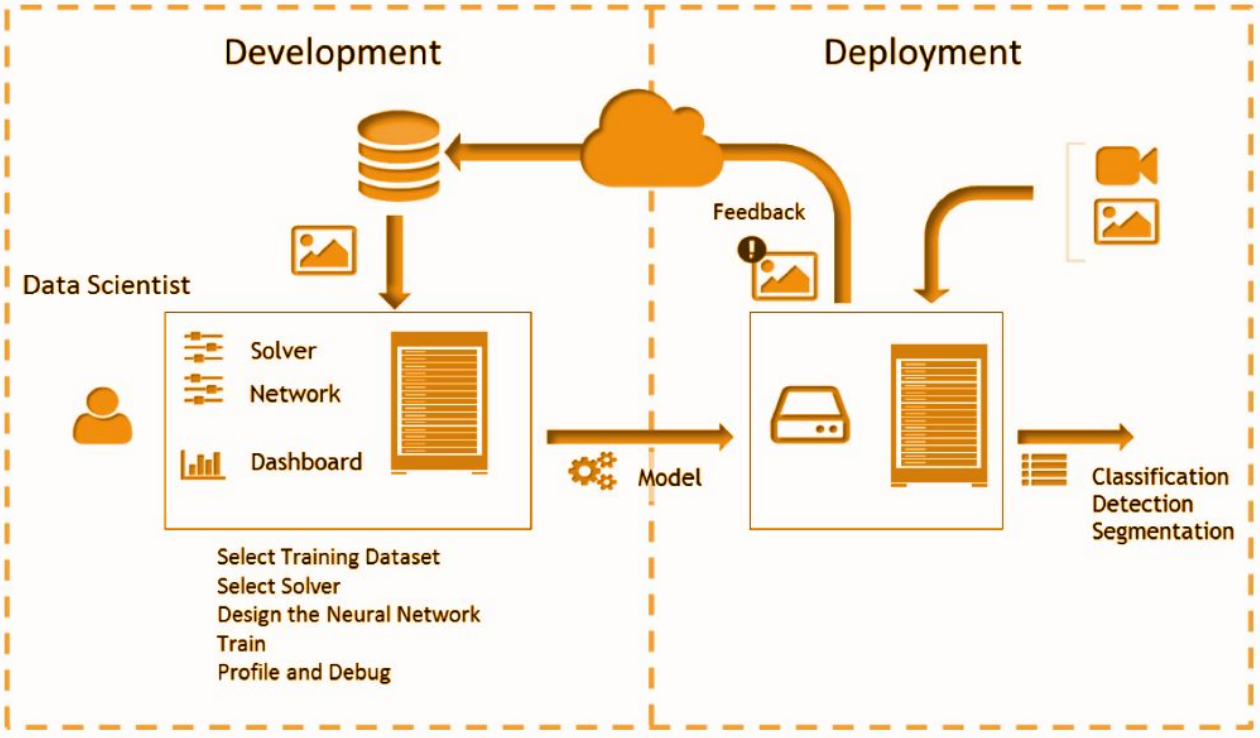
Deployment / Implantação (Embarcados)

Como podemos utilizar a rede neural treinada como parte de uma solução?

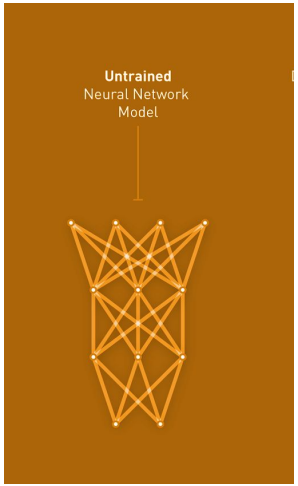


Deployment / Implantação (Embarcados)

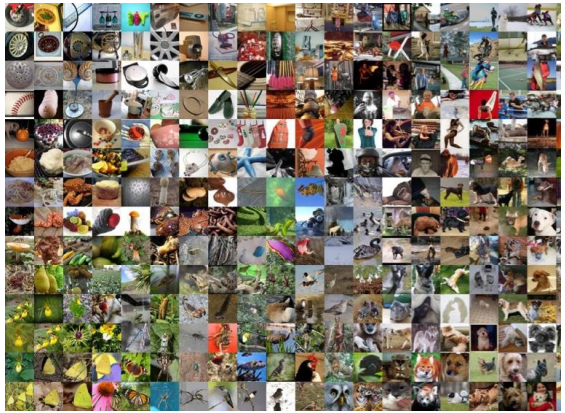
Como podemos utilizar a rede neural treinada como parte de uma solução?



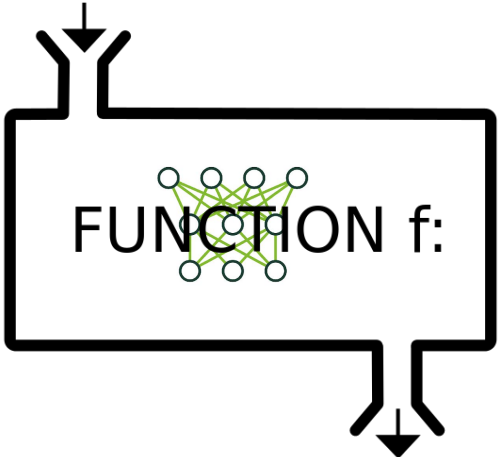
Arquitetura Modelo = deploy.prototxt

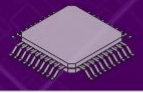


Pesos calculados = ***.caffemodel



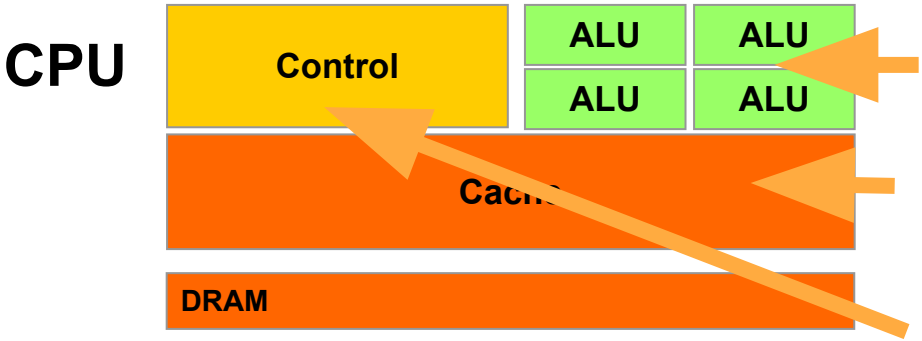
Modelo





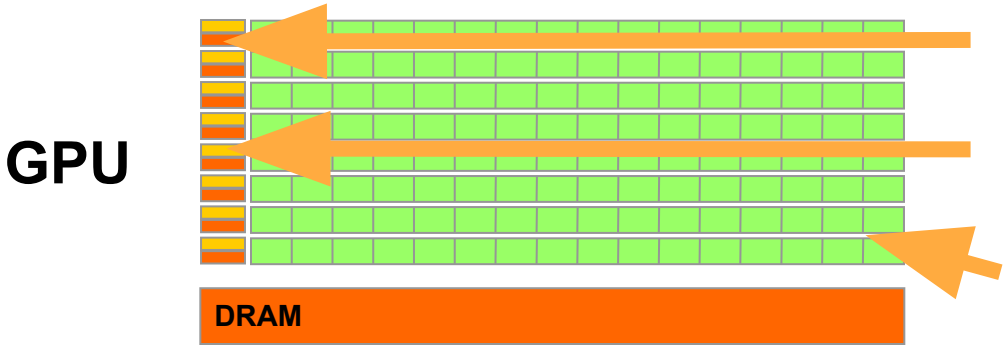
Modelo Computacional CPU versus GPU

CPUs: Projeto voltado à Latência

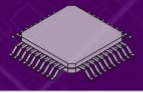


- ULA Potente
  - Reduzir latência de cada operação
- Caches de tamanho considerável
  - Converter grande latência de acesso à memória em pequena latência com acesso à cache
- Controle sofisticado
  - Previsão de *branches* para reduzir latência de *branches*

GPUs: Projeto voltado ao Throughput



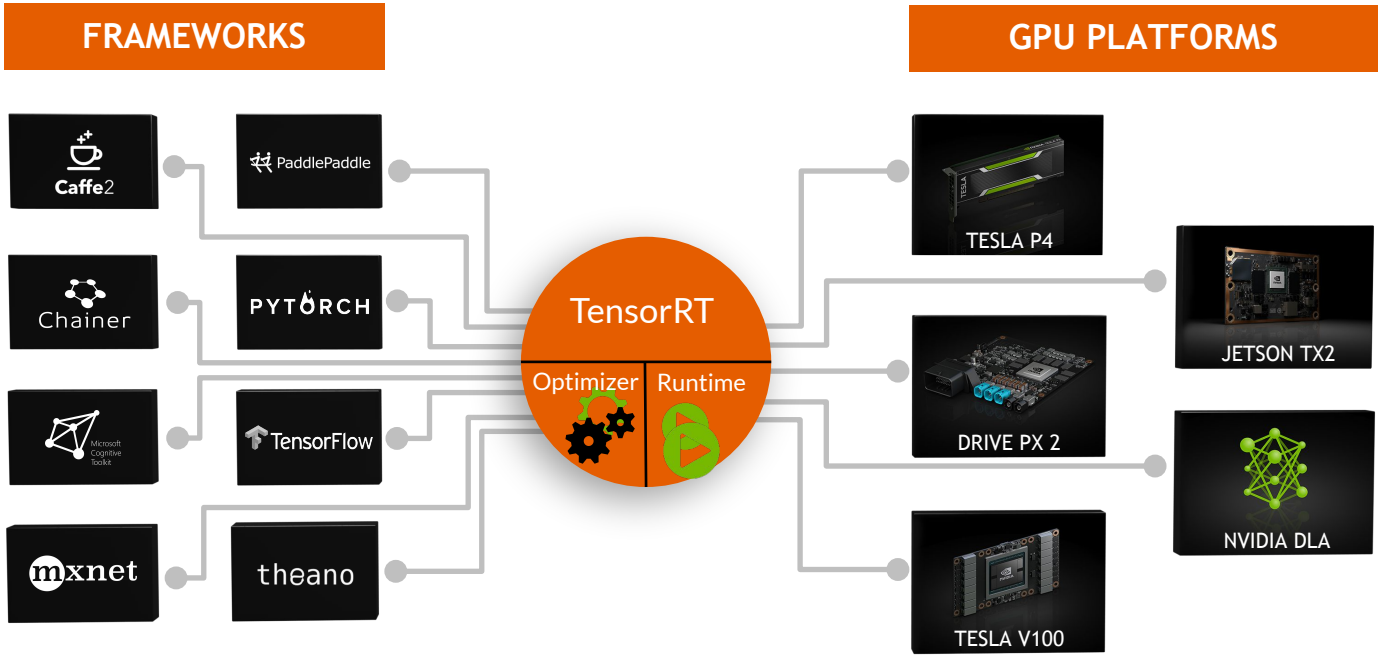
- Pequenos caches
  - Para melhorar throughput de memória
- Controle Simplificado
  - Sem previsão de *branch*
- ULAs com eficiência energética
  - Muitas, grande latência but extremamente otimizada (pipelines) para alto número de operações por unidade de tempo (throughput)
- Exige número grande de threads para tolerar latências
  - Lógica para threading
  - Estado de threads



Aplicações que utilizam ambos CPU e GPU

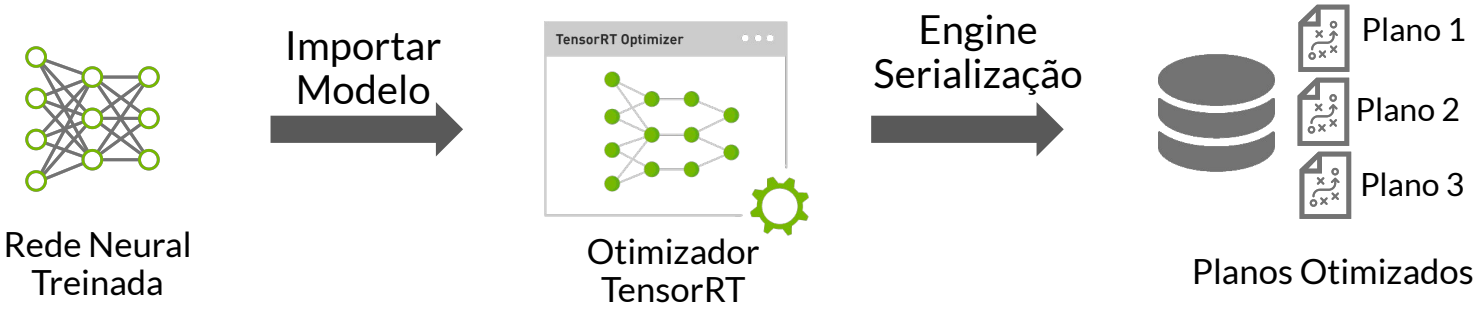
- CPUs p/ partes sequenciais onde latência importa
  - CPUs pode ser 10X+ mais rápidas que GPUs para código sequencial
- GPUs p/ partes paralelas onde throughput é importante
  - GPUs chegam a ser 10X+ mais rápidas que CPUs para código paralelo

NVIDIA Tensor-RT

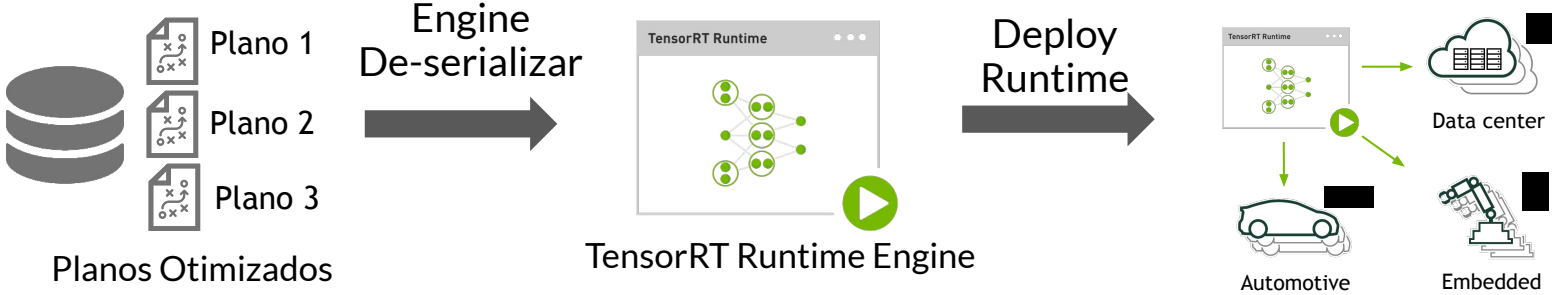


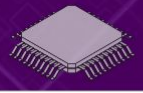
TensorRT: WORKFLOW para Embarcar

Passo 1: Otimizar modelo treinado



Passo 2: Implantar Planos Otimizados

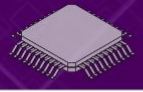




Embarcando modelos em Nano GPUs

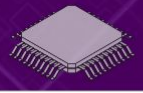
JETSON NANO SPECIFICATIONS

GPU	128 Core Maxwell 472 GFLOPs (FP16)
CPU	4 core ARM A57 @ 1.43 GHz
Memory	4 GB 64 bit LPDDR4 25.6 GB/s
Storage	16 GB eMMC
Video Encode	4K @ 30 | 4x 1080p @ 30 | 8x 720p @ 30 (H.264/H.265)
Video Decode	4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 16x 720p @ 30 | (H.264/H.265)
Camera	12 (3x4 or 4x2) MIPI CSI-2 DPHY 1.1 lanes (1.5 Gbps)
Display	HDMI 2.0 or DP1.2 | eDP 1.4 | DSI (1 x2) 2 simultaneous
UPHY	1 x1/2/4 PCIE 1 USB 3.0
SDIO/SPI/SysIOs/GPIOs/I2C	1x SDIO / 2x SPI / 5x SysIO / 13x GPIOs / 6x I2C

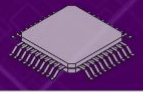


Developer Kit Technical Specifications	
GPU	128-core NVIDIA Maxwell™
CPU	Quad-core ARM® A57 @ 1.43 GHz
Memory	2 GB 64-bit LPDDR4 25.6 GB/s
Storage	microSD (Card not included)
Video Encoder	4Kp30 | 4x 1080p30 | 9x 720p30 (H.264/H.265)
Video Decoder	4Kp60 | 2x 4Kp30 | 8x 1080p30 | 18x 720p30 (H.264/H.265)
Connectivity	Gigabit Ethernet, 802.11ac wireless*
Camera	1x MIPI CSI-2 connector
Display	HDMI
USB	1x USB 3.0 Type A, 2x USB 2.0 Type A, 1x USB 2.0 Micro-B
Others	40-pin header (GPIO, I2C, I2S, SPI, UART) 12-pin header (Power and related signals, UART) 4-pin fan header*
Mechanical	100 mm x 80 mm x 29 mm

Equipe Hydrone
Competição Brasileira de Robótica
Desafio Petrobras, 12-14 Novembro



TinyML: modelos embarcados em CPUs



O que é TinyML?

- Rodar “aprendizado de máquina” em sistemas embarcados com menos de 1 milliwatt de potência. Definição pode abranger MCUs que usam algumas dezenas de milliwatts (disponíveis facilmente no mercado).
- Consumo de Energia é fator importante
  - Permite dispositivos funcionem sem intervenção humana em baterias ou através de captação de energia externa.

https://www.tinyml.org/home/

Slides baseados em : https://www.tinyml.org/wp-content/uploads/2020/03/tinyML_Talks_200331_Pete_Warden.pdf

Hardware

Arduino Nano BLE Sense 33

- $30

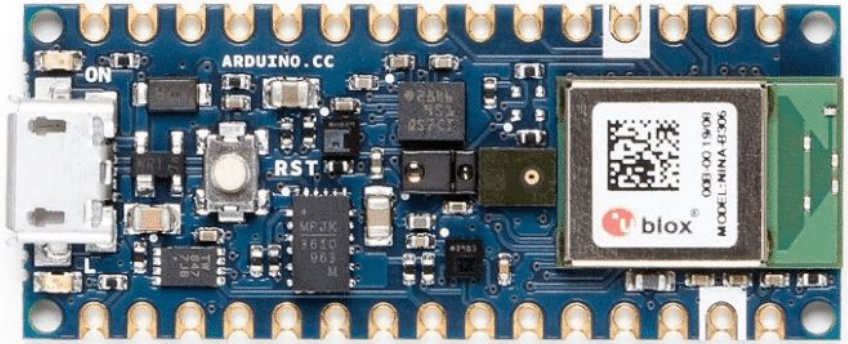
- Fácil de programar

- Bluetooth Low Energy (BLE)

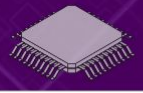
- 10’s milliwatts

- Não muito fácil de alimentar c/ bateria

- Pequeno em tamanho

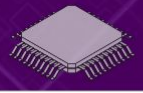


https://www.embarcados.com.br/novas-super-placas-da-familia-arduino-nano/



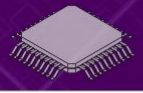
Software

- TensorFlow Lite Micro
- tensorflow.org/lite/microcontrollers
- github.com/uTensor/uTensor
- edgeimpulse.com
- cartesiam.ai



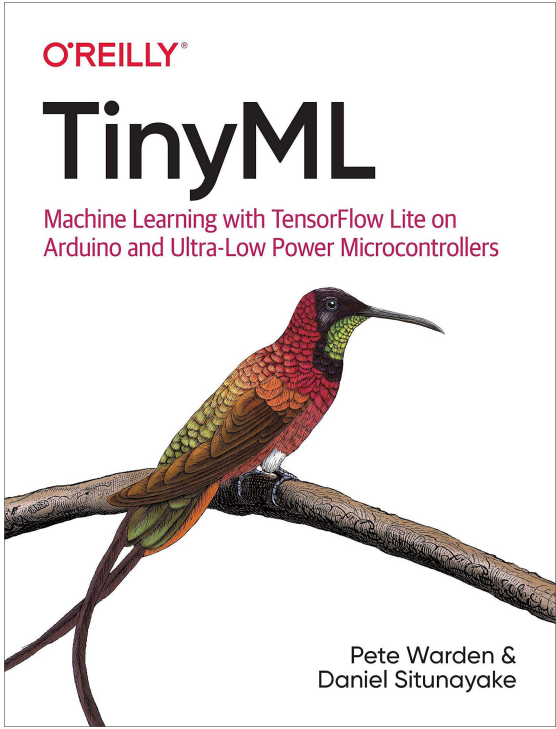
TensorFlow Lite Micro

- Biblioteca do Google
  - *Machine learning* em microcontroladores e DSPs
  - tensorflow/lite/micro utiliza < 20KB (tamanho do binário), não possui sistema operacional
  - Modelos são criados no TensorFlow
  - TensorFlow Lite converte modelos em representações otimizadas p/ pequenos dispositivos.
- Aplicações do tipo “always-on”
- Tem atraído o interesse de parceiros da indústria, com kernels otimizados para Arm Cortex M, Cadence Tensilica Hifi, Synopsys ARC chips, e outros.



Onde iniciar? Como aprender mais?

- Livro “TinyML”, Pete Warden & Daniel Situnayake
- tinymlbook.com
- Capítulos com exemplos práticos, passo-a-passo para embarcar modelos em vários tipos de placas de desenvolvimento e CPUs.
- Os primeiros capítulos estão disponíveis gratuitamente no website http://tinymlbook.com



EMBARCADOS EXPERIENCE 2020

Patrocínio Ouro

MOUSER ELECTRONICS

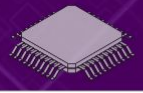
Realização

EMBARCADOS

28 de Novembro

OBRIGADO!

Evento online com muito networking
e troca de conhecimento.

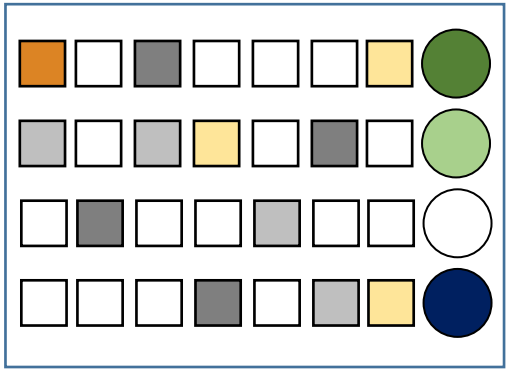


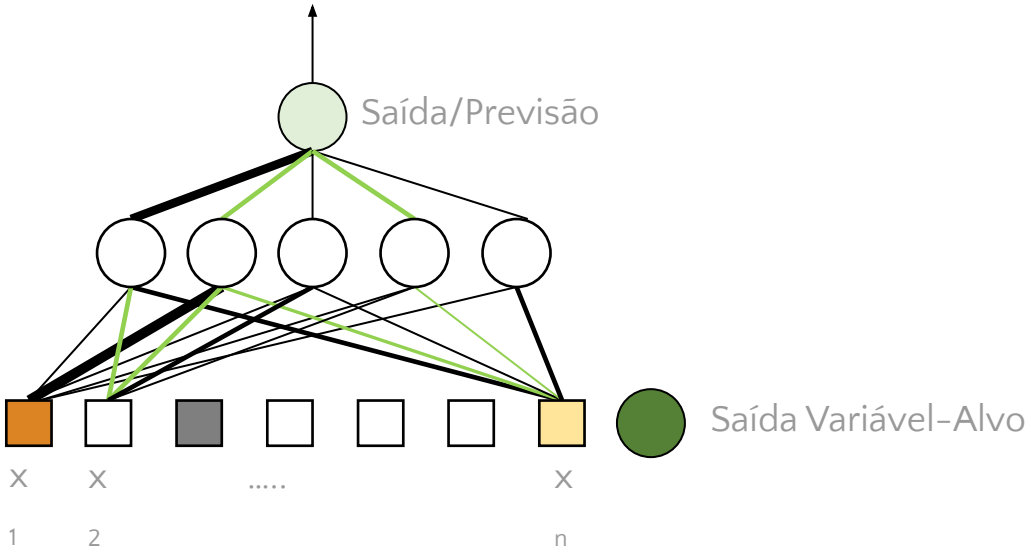
Deep Learning - Medindo Desempenho

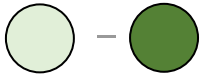
Requisito	Desafios
High Throughput	**Dificuldade de processar grande volume, grande velocidade geração de dados** Impacto: aumento de custo ($, time) por inferência
Low Response Time	**Aplicações não entregam resultados em tempo-real** Impacto: afeta negativamente a experiência do usuário (reconhecimento de voz, recomendações personalizadas, detecção tempo-real)
Power and Memory Efficiency	**Aplicações ineficientes** Impacto: aumento de custo (processamento e resfriamento), torna implantação impraticável

Princípio do Aprendizado

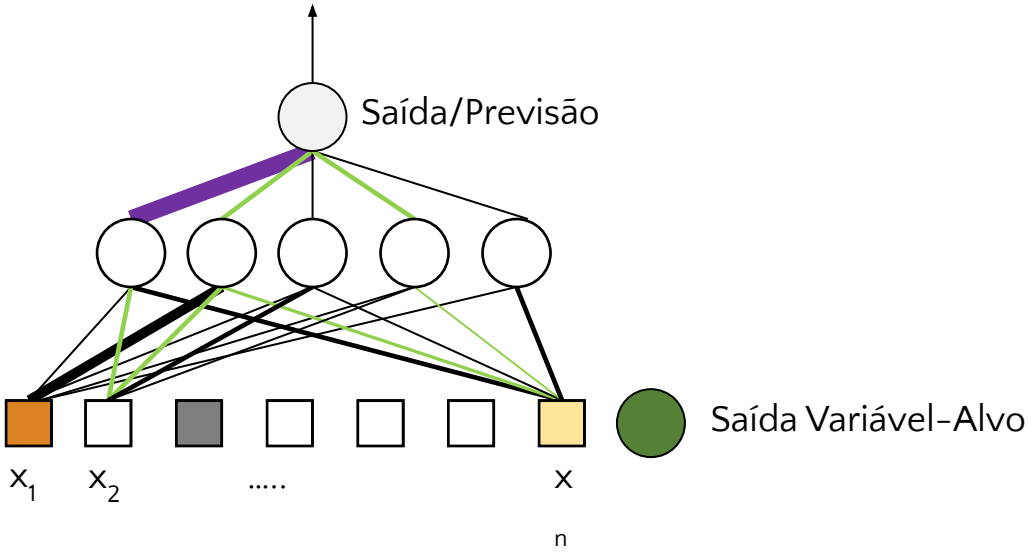
Dataset





Erro:  = 5

Princípio do Aprendizado



Erro: – = 15

Princípio do Aprendizado