TensorFlow

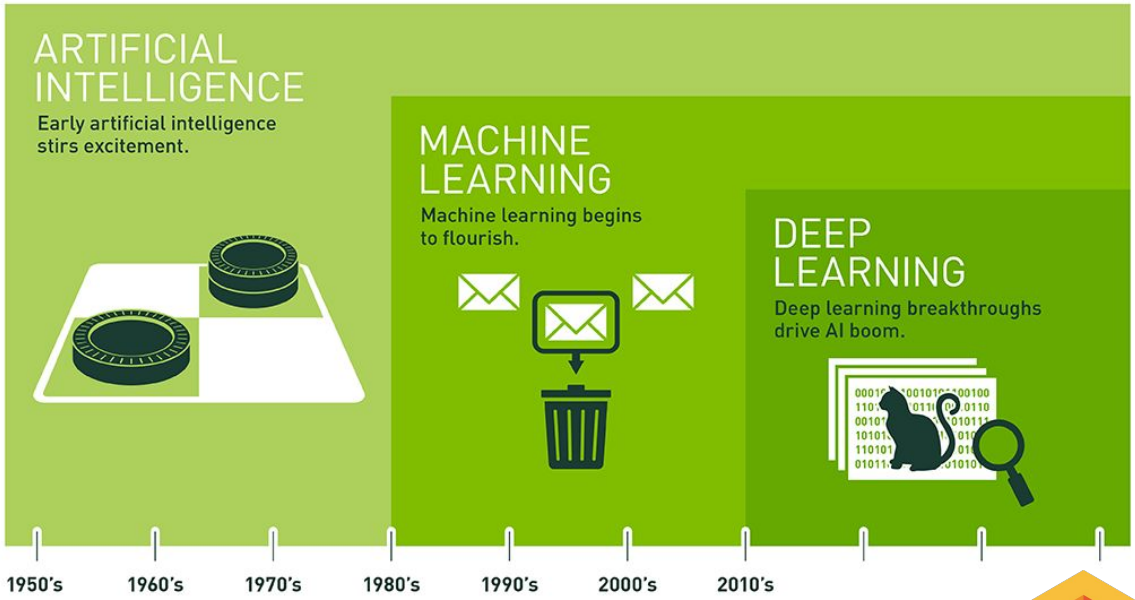
Gisela Miranda e Leonardo Felix

Machine Learning vs Deep Learning

Machine learning utiliza algoritmos para calcular dados, aprender com esses dados e fazer decisões baseadas no que aprendeu.

Deep learning estrutura os algoritmos em camadas para criar uma “rede neural” artificial que pode aprender e realizar decisões inteligentes por conta própria.

Deep learning neural networks é o termo usado para descrever como a inteligência artificial funciona de forma similar com a dos humanos.



Deep Learning

- **Google** - Reconhecimento de voz e imagens
- **Netflix** - Decidir o que você quer assistir na próxima vez
- **Amazon** - Decidir o que você quer comprar na próxima vez

Introdução

- A história do TensorFlow começa em 2011 com a 1ª geração do sistema de machine learning baseado em deep learning neural networks, o **DistBelief**.

- A 2ª geração, o **TensorFlow**, foi liberada em 2015, sendo este:
  + flexível
  + escalável
  + performance

A versão 1.0.0 foi lançada apenas em 2017.

O que é o TensorFlow?

"TensorFlow é uma interface para expressar algoritmos de deep learning e uma implementação para executar tais algoritmos ".

É uma interface que **permite a comunicação com o sistema operacional**, dando controle ao desenvolvedor, de forma facilitada, mantendo sob sua responsabilidade o gerenciamento de recursos, visando **aumentar a eficiência no processo de aprendizado de máquina**.

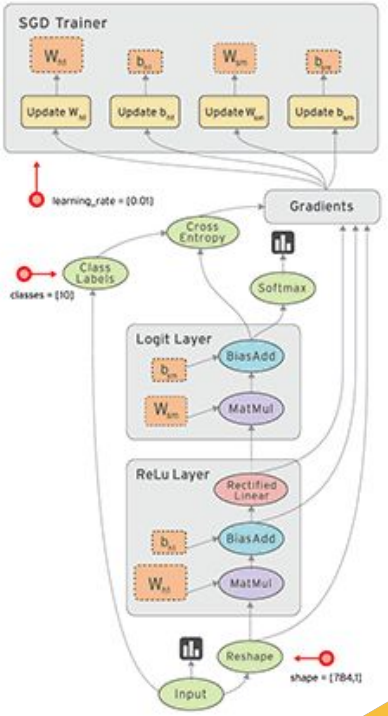
O que é o TensorFlow?

Baseado em gráfico de fluxo de dados (Dataflow graph)

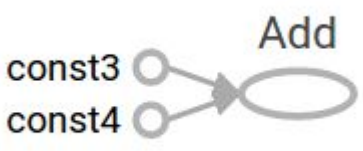
- As operações matemáticas (**ops**) são expressas como **nós**.
- Arrays de dados multidimensionais são chamados de **tensors.** Representados como **linhas**.

Por isso o nome TensorFlow.

Sua representação gráfica é feita pelo chamado **TensorBoard.**

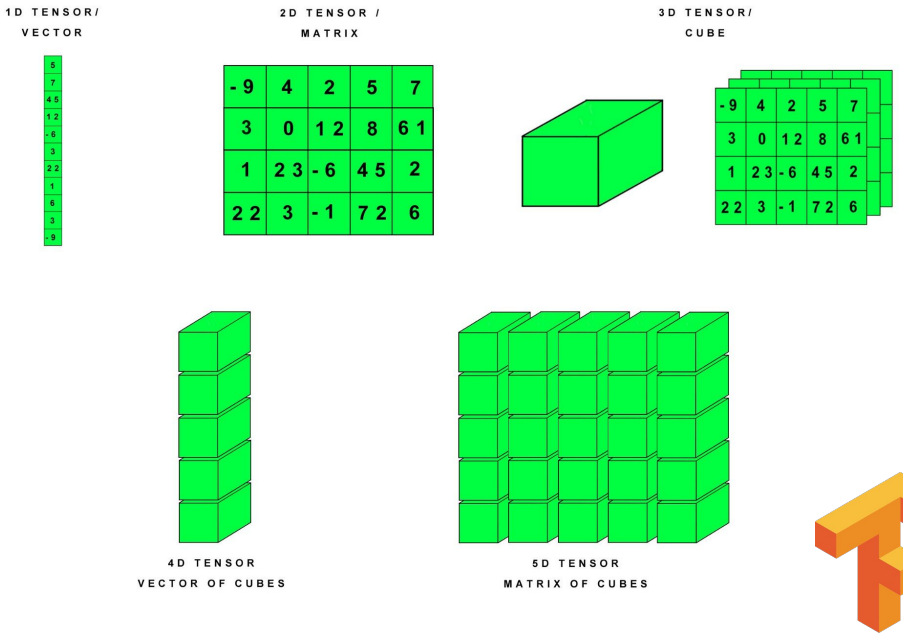


O que é o TensorFlow?



→ **Nodo**

Tensor ←



O que é o TensorFlow?

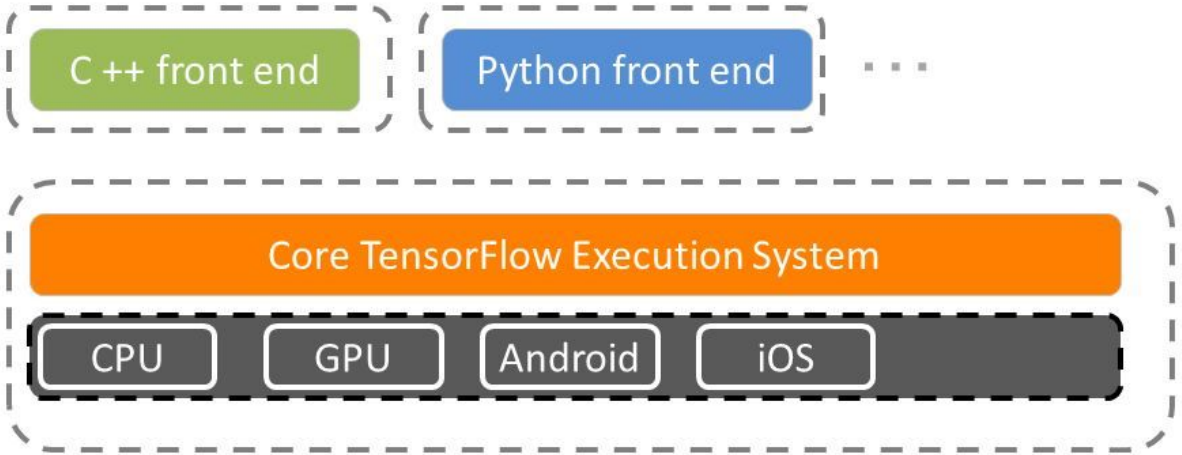
Pode ser utilizado em diversos sistemas uniformemente, podendo variar de dispositivos móveis a computadores pessoais ou até mesmo sistemas de larga-escala trabalhando de forma paralela.

Sistemas Operacionais:

- Linux
- Mac OS X
- Windows
- Android
- iOS

Arquitetura do TensorFlow

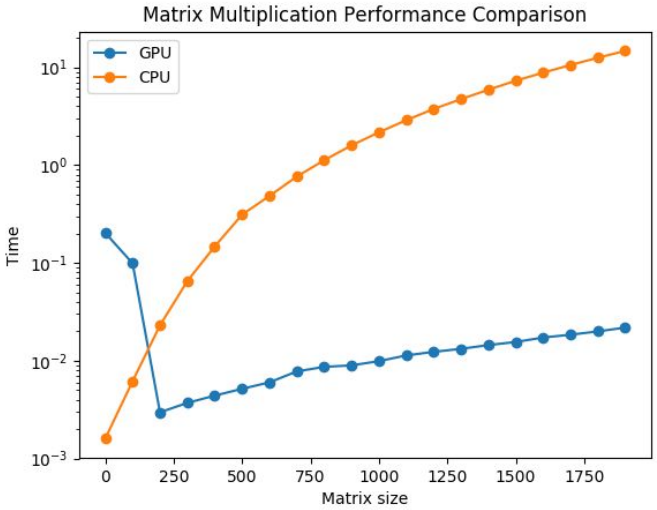
- API em **C++** e em **Python**
- A linguagem de baixo nível utilizada é o **C**



Arquitetura do TensorFlow

O TensorFlow permite que os dados ou instruções sejam passados entre **dispositivos com diferentes sistemas operacionais** ou funcionalidades, como de uma CPU de um dispositivo móvel, para a TPU ligada em um servidor, por exemplo.

O programa pode segmentar facilmente GPUs, TPUs (*Tensor Processing Unit*) ou CPUs móveis conforme necessário.

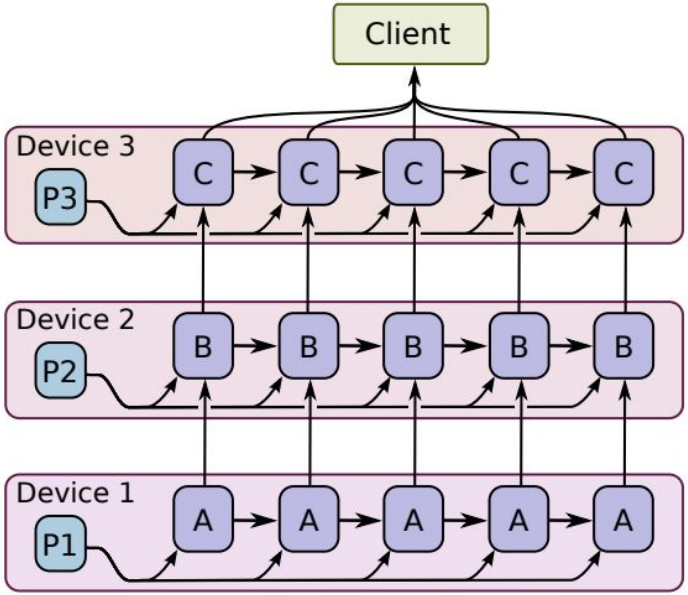


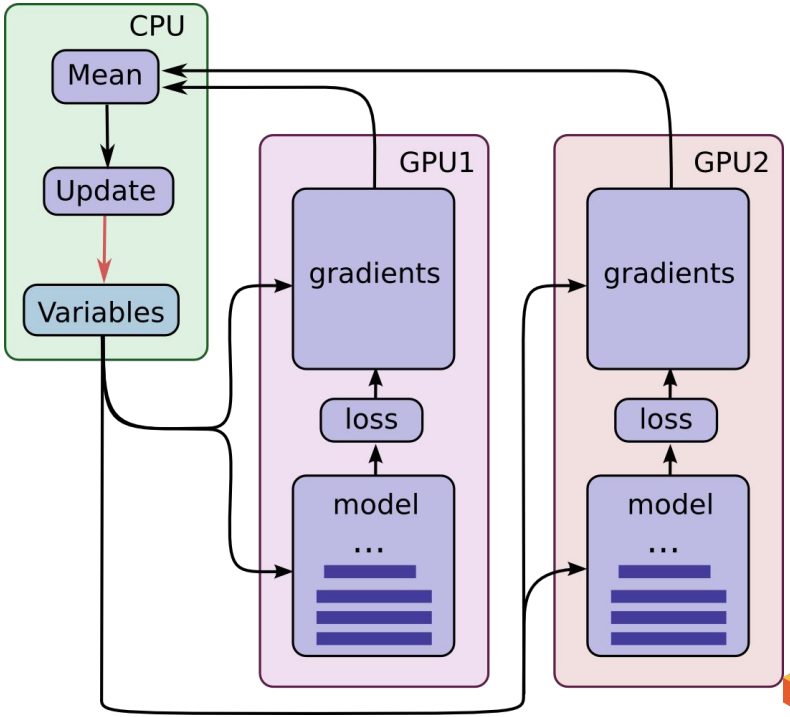
Comunicação com o Sistema Operacional

Possui interface simples para abstração com outros dispositivos ou redes:

- **with tf.device(‘gpu:0’)** - Define o uso da GPU como processador principal e aloca a memória necessária
- **tf.GPUOptions(per_process_gpu_memory_fraction = 0.5)** - Limita o uso da memória da GPU em 50%
- **tf.Session()** executa os comandos definidos no escopo de cada dispositivo, em paralelo, se possível.
- Também é possível apresentar os logs com **log_device_placement = True**

Comunicação com o Sistema Operacional





Comunicação com o Sistema Operacional

- Não é necessário controlar threads e/ou processos
- Não é necessário controlar o retorno de uma das funções.
  A biblioteca criará o grafo de dependências e fará o gerenciamento.
- Em sistemas distribuídos, os resultados serão enviados ao dispositivo que calcula os resultados de forma automática.

Sincronização de Processos

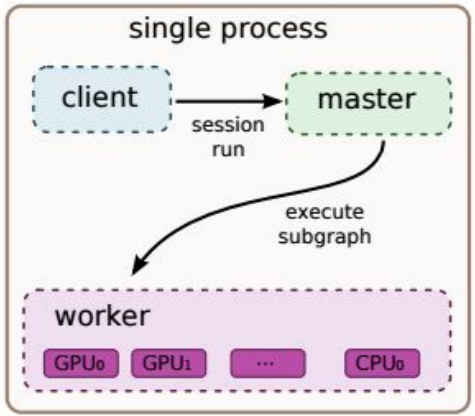
O TensorFlow utiliza um **mutex** para realizar exclusões mútuas, de forma a eliminar a preocupação com segurança de threads. Exemplos:

- Acesso concorrente a um **Tensor**
- Leitura e escrita em arquivo

Implementações

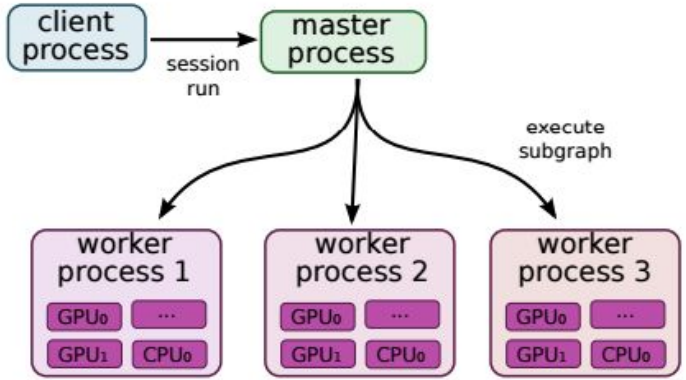
Local

- Client, master e worker rodam em uma única máquina (processo com um sistema operacional)



Distribuído

- Client, master, e workers rodam em processos diferentes e em máquinas diferentes.



Experimento

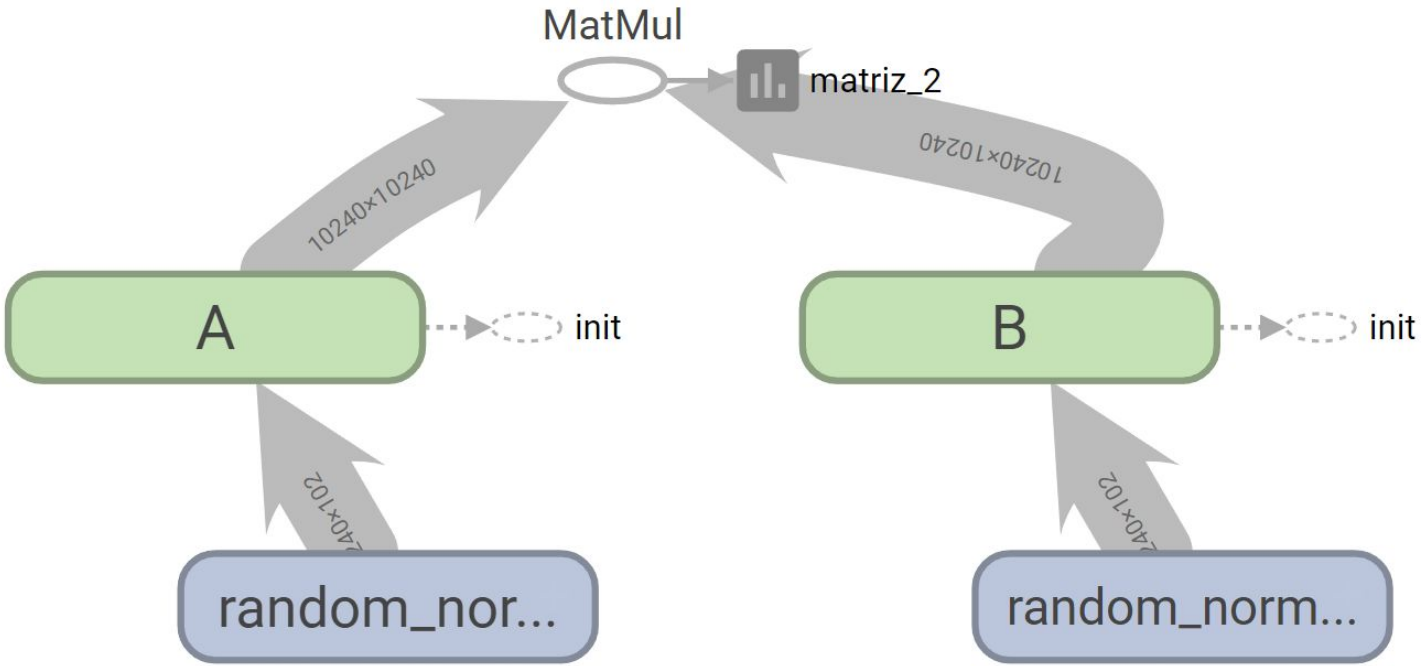
Objetivo: Comparar a execução **CPU vs GPU**

Foi utilizado uma multiplicação de matrizes com números aleatórios de tamanho N*1024 x N*1024, sendo o N=10.
Implementação utilizando a linguagem **Python**

```
with tf.device('/cpu:0'):
    A = tf.Variable(tf.random_normal((N*1024,N*1024)), name="A")
    B = tf.Variable(tf.random_normal((N*1024,N*1024)), name="B")
    C = tf.matmul(A, B)
```

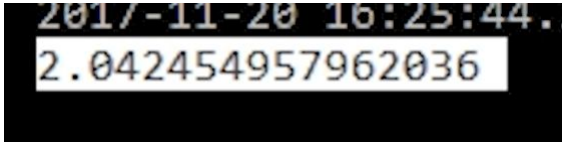
```
with tf.device('/gpu:0'):
    A = tf.Variable(tf.random_normal((N*1024,N*1024)), name="A")
    B = tf.Variable(tf.random_normal((N*1024,N*1024)), name="B")
    C = tf.matmul(A, B)
```

Experimento

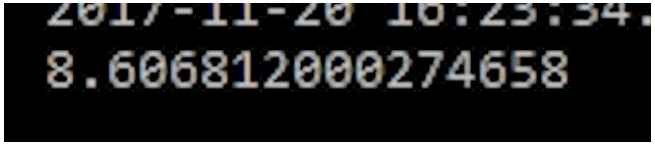


Conclusão

Execução GPU(s):



Execução CPU(s):



- **Facilidade de definir o dispositivo** necessário para processar cada tipo de dado
- Grande parte das **responsabilidades** de gerenciamento de recursos **fica com a biblioteca**
- **Paralelização** pode ser feita com poucas linhas de código

Experimento - link

https://drive.google.com/open?id=1q21o9ZHkXQ9-jn7sURY-jBLjrgkG4uXe