

# Why GPUs for Machine Learning?

Julien Nyambal

Capitec Bank

February 23, 2023

While a CPU is the brains of a computer, GPU is its soul ... Nvidia

# What will be covered ...

- ① What is Machine Learning?
- ② RAM, CPU, GPU, TPU
  - Description
- ③ RAM, CPU, GPU, TPU
  - TPU: Tensor Processing Unit
- ④ Hardware Comparison CPU vs GPU
  - Hardware Comparison CPU vs GPU
- ⑤ Tensor Operations on Hardware
  - Matrix Computation on GPU vs CPU
- ⑥ How does GPU works "faster" than the CPU?
- ⑦ Very short demo

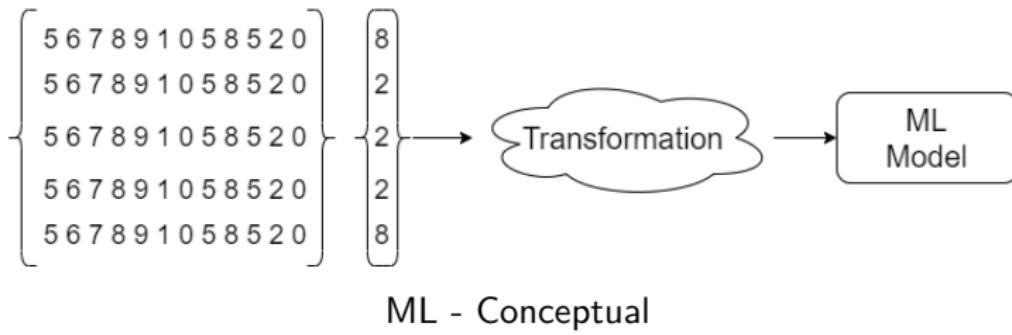
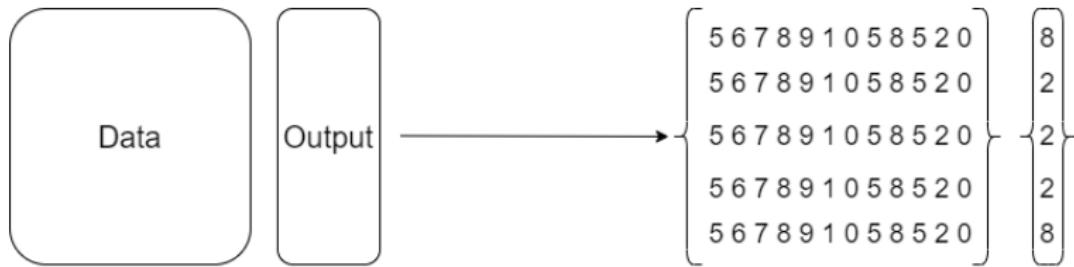


# What is Machine Learning?

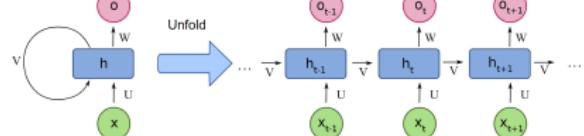
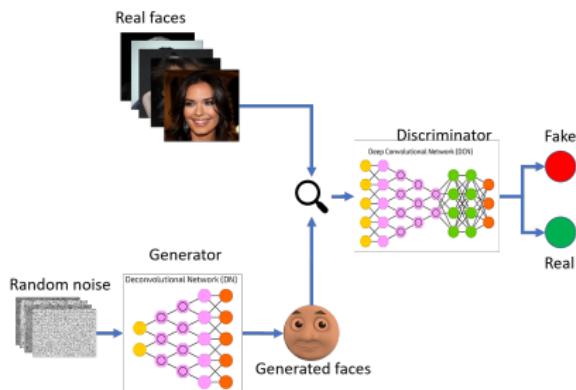
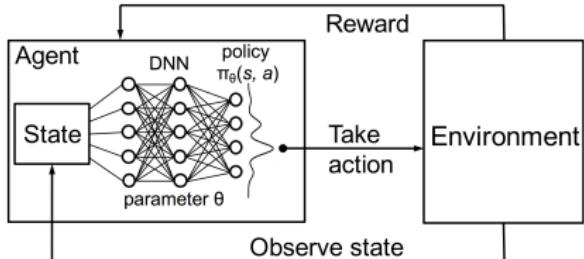
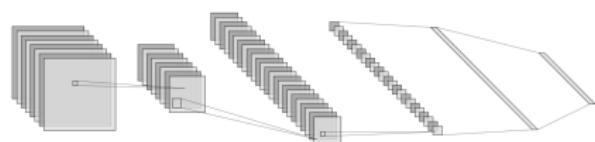


A general definition of a ML

# What is Machine Learning?: Conceptual Overview



# What is Machine Learning?: Deep Learning



# RAM, GPU, CPU, TPU

Acronym	Central Processing Unit	Graphical Processing Unit	Tensor Processing Unit
Description	Performs basic logic, arithmetic, and I/O operations, and allocate commands to other components and sub-systems running in a computer. It is the orchestrator of the other hardware components of the computer system, including the GPU. Most systems have a <b>shared GPU resource for basic display</b> .	Dedicated device designed specifically to accelerate computer graphics workloads, particularly for 3D graphics. While they are still used for their original purpose of accelerating graphics rendering, GPU parallel computing is now used in a wide range of applications, including graphics and video rendering.	Intel
Processing style	Sequential (But multitask)	Distributed (Given a single heavy-duty load)	-
Graphic Rendering API	OpenCL	CUDA	-
Specialty	Orchestrating several tasks performed by the OS	<ul style="list-style-type: none"><li>Graphically intense video game Image Rendering</li><li>Distributed Complex matrix operation: <b>Machine Learning</b> and <b>HPC</b></li><li>Video Editing</li></ul>	-

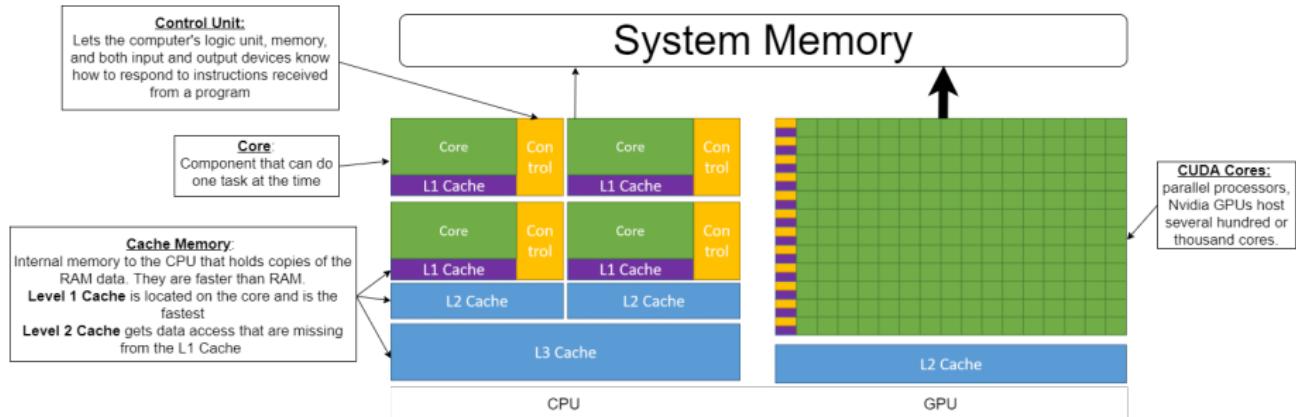


# RAM, GPU, CPU, TPU



PITEC

# Hardware Comparison CPU vs GPU



# Hardware Comparison CPU vs GPU

Processor Type	CPU	GPU
Processor Model	i9-10900K	Tesla V100
Manufacturer	Intel	Nvidia
Processor Speed (Max)	5.30 GHz	1.380 GHz
Memory (up to)	128 GB (RAM)	32 GB (GRAM)
Number of Cores	10	5120 (CUDA cores)
Memory Bandwidth	45.8 GB/s	900 GB/s
Price	~R 16 000	~R 200 000

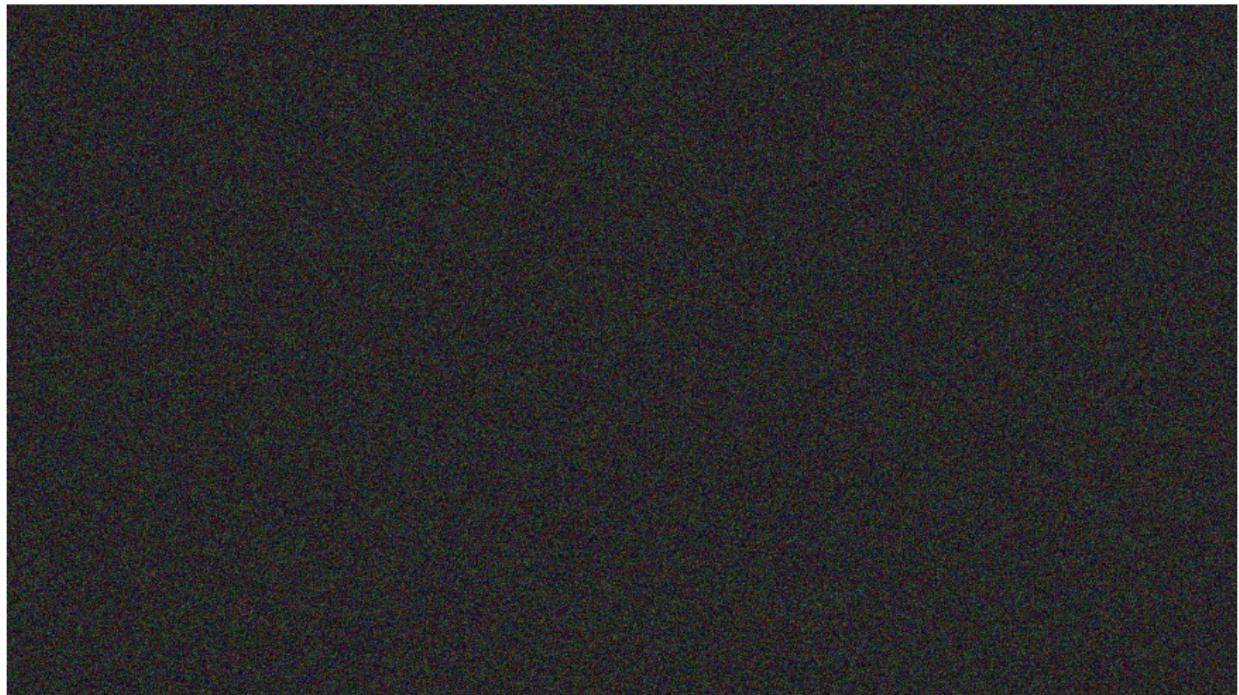
# Tensors: Operations - Addition



Original Image



# Tensors: Operations - Addition



Noise



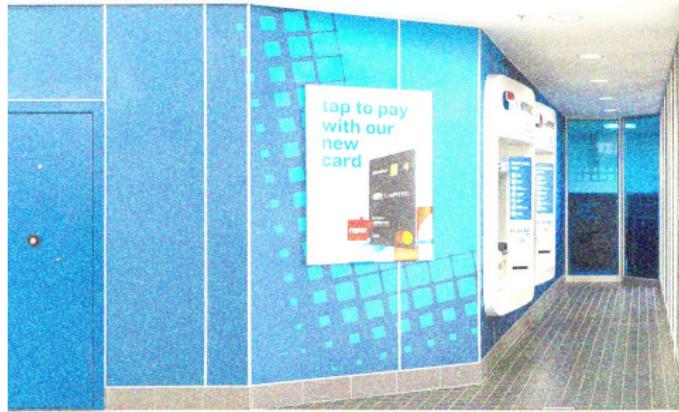
# Tensors: Operations - Addition



Image + Noise



# Tensors: Operations - Addition



(Image + Noise) + Image

# Tensors: Operations - Matrix Multiplication

- The most used operation in Machine Learning/Deep Learning
- There are many types of Matrix Multiplication of **MatMul**. There are 3 common matrix multiplication:
  - ▶ Brute Force Multiplication
  - ▶ Column-Wise Multiplication
  - ▶ Block Multiplication

# Tensors: Operations - Matrix Multiplication



Image



# Tensors: Operations - Matrix Multiplication



Convolution 1



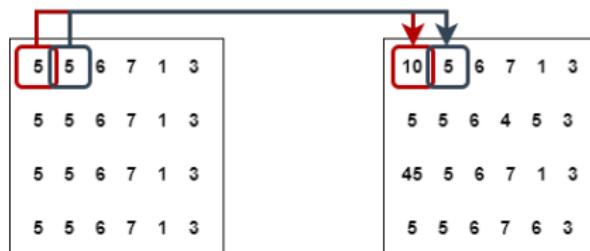
# Tensors: Operations - Matrix Multiplication



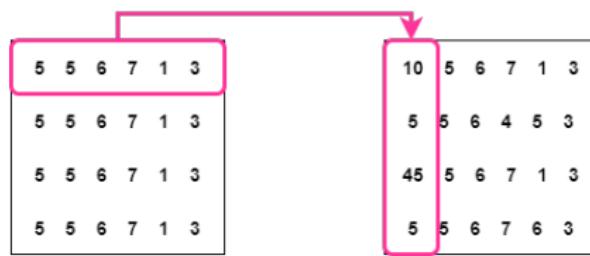
Convolution 2



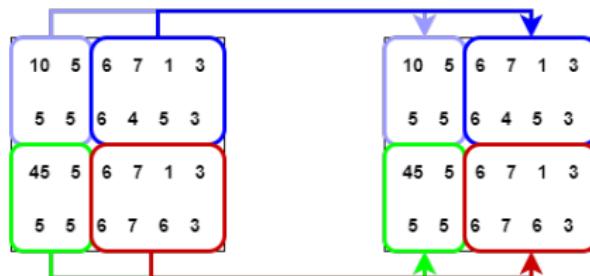
# Tensors: Operations - Matrix Multiplication



Brute Force Matrix Multiplication



Column-Wise Matrix Multiplication



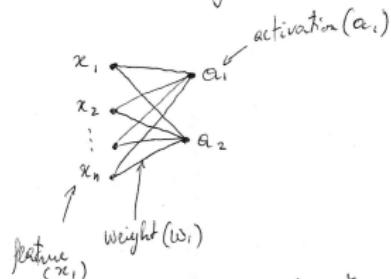
Block Matrix Multiplication

# Tensors: Operations - Matrix Multiplication

Forward propagation

$$f(w_{ij}, x_i) = W \cdot X + b \quad (1)$$

Assuming:



To calculate the activations

$$a_i = w_{i,1} \times x_1 + w_{i,2} \times x_2 + \dots + w_{i,n} \times x_n + b$$

Consequently

The above applies for all activations

By hand, and that way  
this is tedious!

Using Matrix notation:

$$\begin{bmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,n} \\ w_{2,1} & w_{2,2} & \dots & w_{2,n} \\ \vdots & & & \\ w_{n,1} & w_{n,2} & \dots & w_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$$

weight  
of size  
 $2 \times 4$

data  
 $n \times 1$

bias  
 $2 \times 1$

activation  
 $2 \times 1$

In the CPU: Individual operations are done at a time

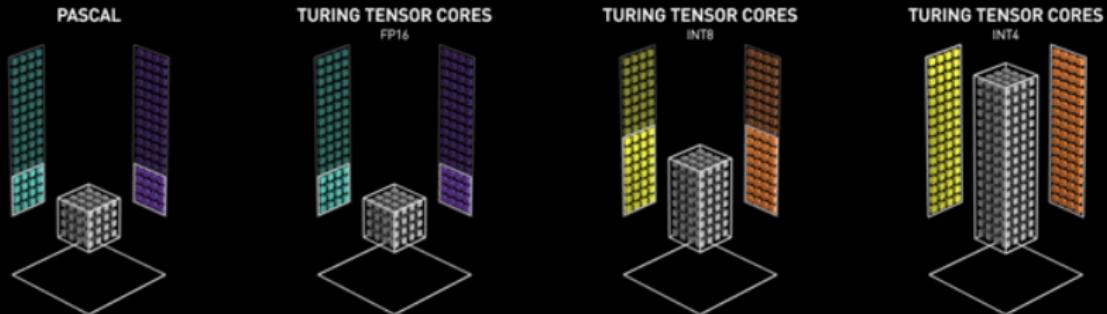
$$= w_{1,1} \times x_1 + w_{1,2} \times x_2 + \dots$$

The CPU is pretty fast but if there are many rows and columns it will take time.

In the GPU Batch operations:

Instead of a single value computed, the GPU takes in a whole row multiplied by a column.

# Matrix Computation on GPU vs CPU



# How does GPU works "faster" than the CPU?

- **Larger memory bandwidth:** More data can get in and out the component at a time,
- **Parallelization:** Those many CUDA cores work well in parallel for one given task
- **Fast Memory access:** Multiple L1 and L2 Cache memory,
- CUDA API allows you not to worry about memory allocation or deallocation, type of Matrix Multiplication to use or how to orchestrate the parallelism of the cores. Some popular frameworks using CUDA:



Keras



PyTorch



TensorFlow



CAPITEC

# Demo

We will run short 2 experiments on both the CPU and the GPU:

- Multiplication of 2 scalar:  $8.8 \times 8.9$
- Multiplication of 2 relatively big matrices

Demo Link: Colab Experiment.

