

Why GPUs for Machine Learning?

Julien Nyambal

Entelect

March 11, 2021

The CPU is the heart of the computer, and the GPU his soul ...

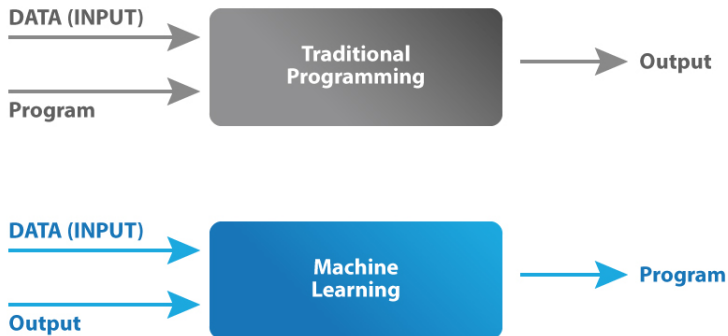


What will be covered ...

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



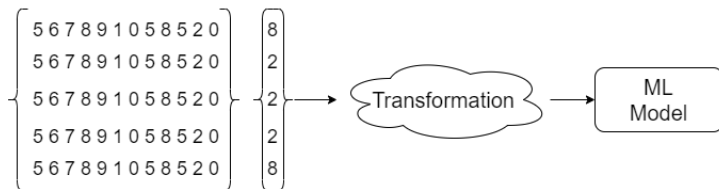
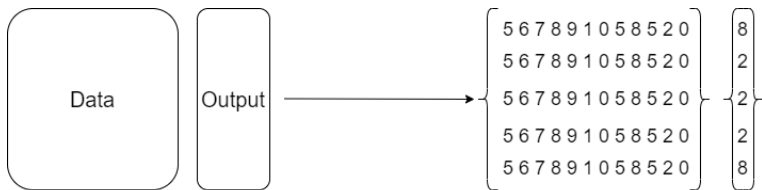
What is Machine Learning?



A general definition of a ML



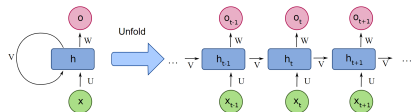
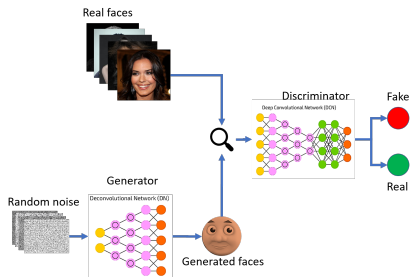
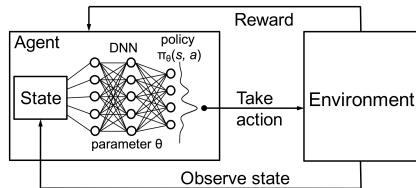
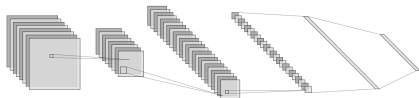
What is Machine Learning?: Conceptual Overview



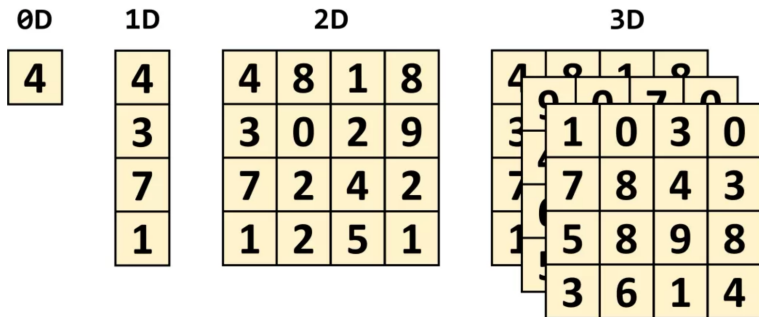
ML - Conceptual



What is Machine Learning?: Deep Learning



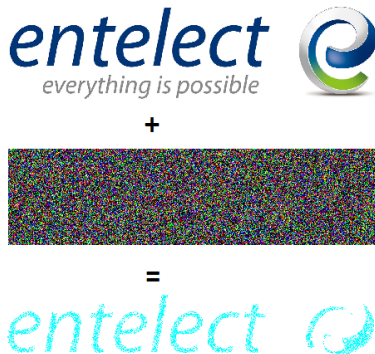
Tensors: A naive description



A general definition of a Tensor



Tensors: Operations - Addition



Tensors: Operations - Matrix Multiplication

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



Tensors: Operations - Matrix Multiplication

Diagram illustrating Brute Force Matrix Multiplication. The first column of the first matrix (A) is highlighted with a red box, and the first row of the second matrix (B) is highlighted with a blue box. A red arrow points from the first element of the first column (5) to the first element of the first row (10), and a blue arrow points from the first element of the first row (10) to the first element of the first column (5). The resulting matrix (C) is shown to the right, with the first element (10) highlighted in a red box.

5	5	6	7	1	3
5	5	6	7	1	3
5	5	6	7	1	3
5	5	6	7	1	3

10	5	6	7	1	3
5	5	6	4	5	3
45	5	6	7	1	3
5	5	6	7	6	3

Brute Force Matrix Multiplication

Diagram illustrating Column-Wise Matrix Multiplication. The first column of the first matrix (A) is highlighted with a pink box, and the first column of the second matrix (B) is highlighted with a pink box. A pink arrow points from the first element of the first column (5) to the first element of the first column (10). The resulting matrix (C) is shown to the right, with the first column (10, 5, 45, 5) highlighted in a pink box.

5	5	6	7	1	3
5	5	6	7	1	3
5	5	6	7	1	3
5	5	6	7	1	3

10	5	6	7	1	3
5	5	6	4	5	3
45	5	6	7	1	3
5	5	6	7	6	3

Column-Wise Matrix Multiplication

Diagram illustrating Block Matrix Multiplication. The first two columns of the first matrix (A) are highlighted with a blue box, and the first two rows of the second matrix (B) are highlighted with a blue box. A blue arrow points from the first element of the first column (5) to the first element of the first row (10). The resulting matrix (C) is shown to the right, with the first two columns (10, 5, 45, 5) highlighted in a blue box.

10	5	6	7	1	3
5	5	6	4	5	3
45	5	6	7	1	3
5	5	6	7	6	3

10	5	6	7	1	3
5	5	6	4	5	3
45	5	6	7	1	3
5	5	6	7	6	3

Block Matrix Multiplication

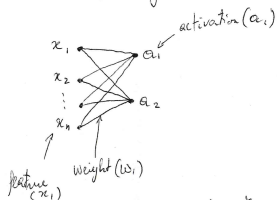


Tensors: Operations - Matrix Multiplication

Forward propagation

$$f(w_i, x_i) = W \cdot X + b \quad (1)$$

Assuming:



To calculate the activations

$$a_1 = w_{1,1} \times x_1 + w_{1,2} \times x_2 + \dots + w_{1,n} \times x_n + b$$

~~However~~

The above applies for all activations

By hand, and that way
this is tedious!

Using Matrix notation:

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

weight of size 2×4 data $n \times 1$ bias 2×1 activation 2×1

In the CPU: Individual operations are done at a time

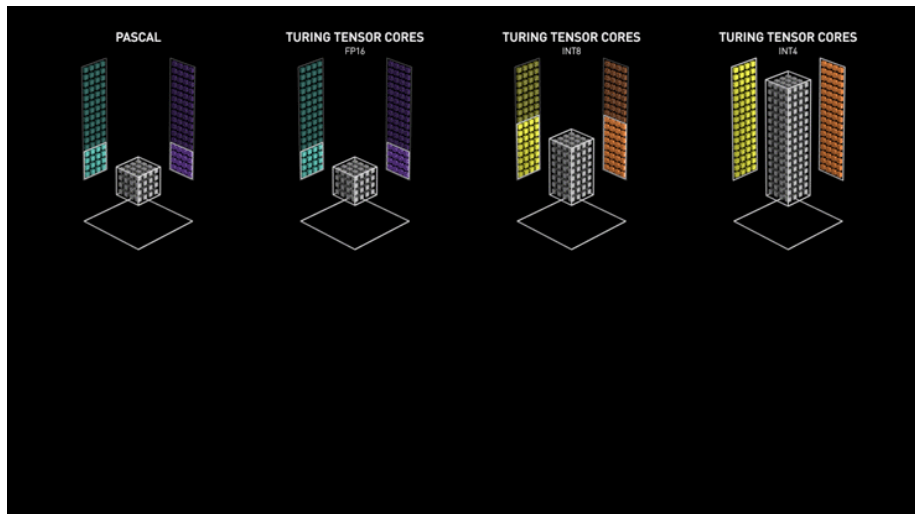
$$= w_{1,1} \times x_1 + w_{1,2} \times x_2 + \dots$$

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

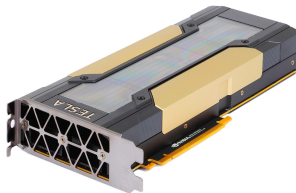
In the GPU Batch operations:

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

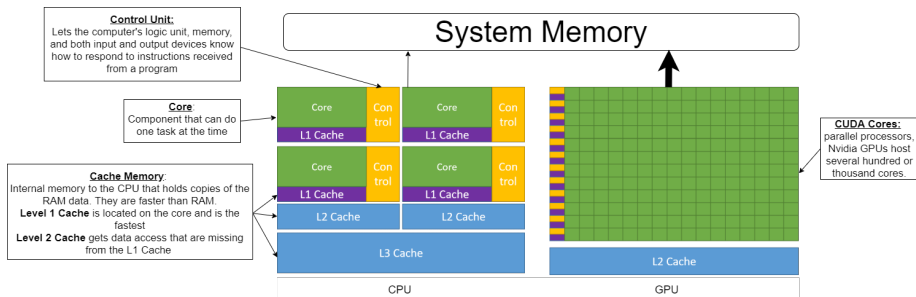
Matrix Computation on GPU vs CPU



RAM, GPU, CPU, TPU



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



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



Demo

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

- Multiplication of 2 scalar: 8.8×8.9
- Multiplication of 2 relatively big matrices

Demo Link: Colab Experiment.

