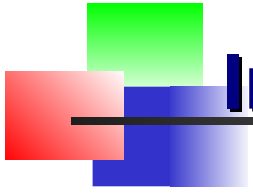


IA Embarquée

P. Bakowski



Intelligence Artificielle (IA) et IA Embarquée

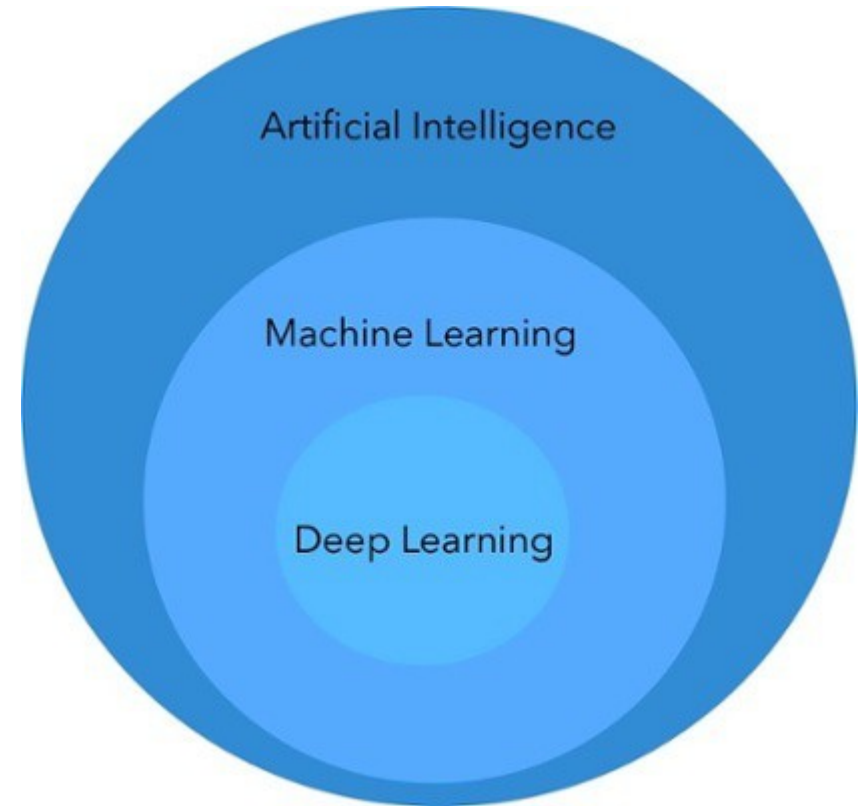
- Introduction
- Matériel pour l'IA Embarquée : CPU, GPU, TPU
- Logiciel pour l'IA Embarquée : TensorFlow Lite, TensorRT
- Lab 0 – Introduction au Réseaux Neuronaux
- Lab 1 – CPU-NEON , Inférence sur RPI 4
- Lab 2 – GPU – Jetson Nano, Inférence
- Lab 3 – GPU - Jetson-Nano, Reconnaissance faciale
- Lab 4 – GPU – Jetson-Nano, Voix
- Lab 5 – TPU/ASIC – Google CORAL, Inférence

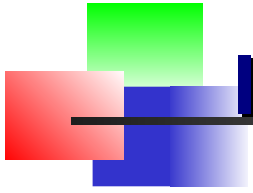


Introduction – IA, ML et DL

■ Le **Machine Learning** est une technologie d'intelligence artificielle permettant aux ordinateurs d'apprendre **sans avoir été programmés** explicitement à cet effet.

■ Pour apprendre et se développer, les ordinateurs ont toutefois **besoin de données** à analyser et sur lesquelles ils doivent **s'entraîner**.





Introduction – IA, ML et DL

- L'**apprentissage profond** est un ensemble de méthodes d'apprentissage automatique tentant de modéliser avec un haut niveau d'abstraction des données grâce à des architectures articulées de différentes **transformations non linéaires**.
- Ces techniques ont permis des progrès importants et rapides dans les domaines de l'analyse du **signal sonore ou visuel** et notamment de la **reconnaissance faciale**, de la **reconnaissance vocale**, de la **vision par ordinateur**, du traitement automatisé du langage.



Introduction – IA Embarquée

■ Grâce à notre capacité à construire des machines intelligentes qui simulent l'intelligence humaine, les implications pour le progrès technologique dans de nombreux secteurs sont infinies. Alors, **quoi de mieux que l'intelligence artificielle ?**

■ **Intelligence Artificielle Embarquée.**

■ L'intelligence artificielle (IA) embarquée est l'application de **ML** et **DL** au **niveau de l'appareil** (*embedded device*)

■ L'Appareil ou *embedded device* est un **SoC** ou un **FPGA** intégré sur une carte autonome – **SCB** (*Single Computer Board*)

■ Exemples : Nvidia -Jetson Nano, Xavier, Google - CORAL, ..



Introduction – Logiciel

- Les logiciels fonctionnant sur les équipements IA embarqués sont (presque) les mêmes que ceux sur les équipements standards (PC, serveurs,..)

- **Langages** : C/C++ et Python

- **Packages** : PyNum, Ski-Learn, ..

- **Couches applicatives** : Keras, TensorFlow, PyTorch, ..

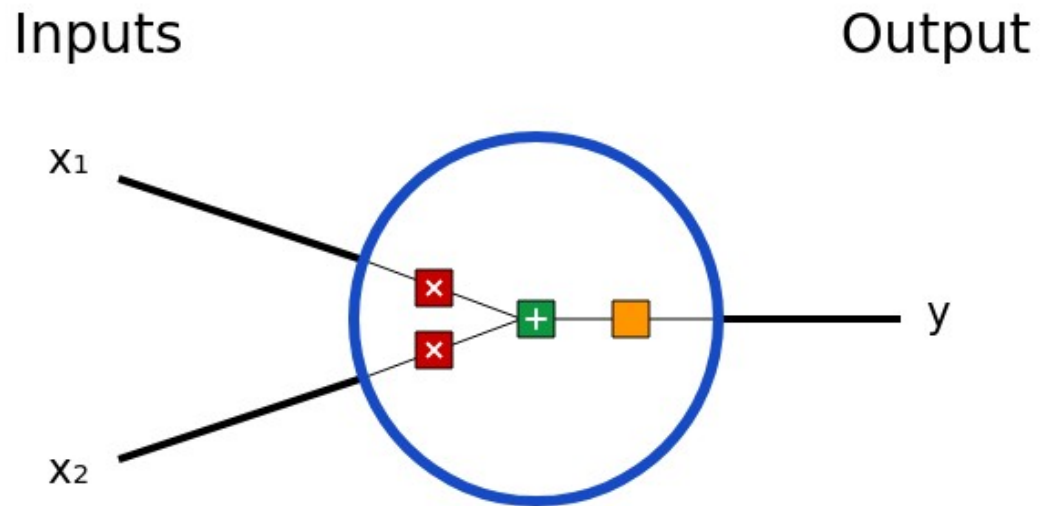
TensorFlow => TensorFlow Lite => TensorFlow Lite_EDGE

- **Couches d'exécution** : CUDA, cuDNN, ..

- **OS** : Linux, FreeRTOS, ..

Introduction – Un Neurone

■ Un **Neurone** est un mécanisme essentiel dans le développement des applications d'IA. Voici sa structure fonctionnelle :



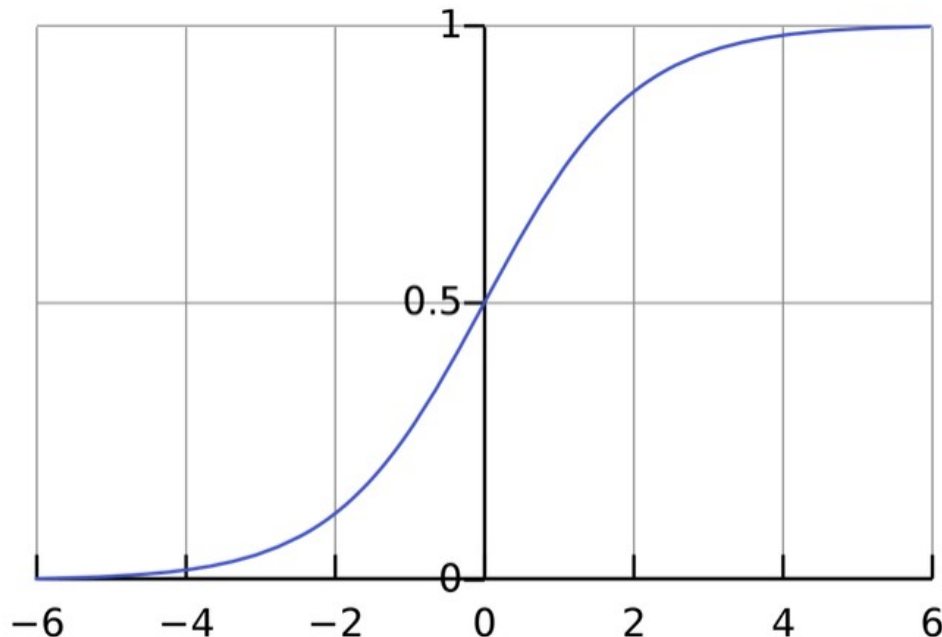
■
$$y = f(x_1 * w_1 + x_2 * w_2 + b)$$

■ où : **w_1** et **w_2** sont des **poids** (*weights*) et **b** est un **biais**

Introduction – Un Neurone

$$y = f(x_1 \cdot w_1 + x_2 \cdot w_2 + b)$$

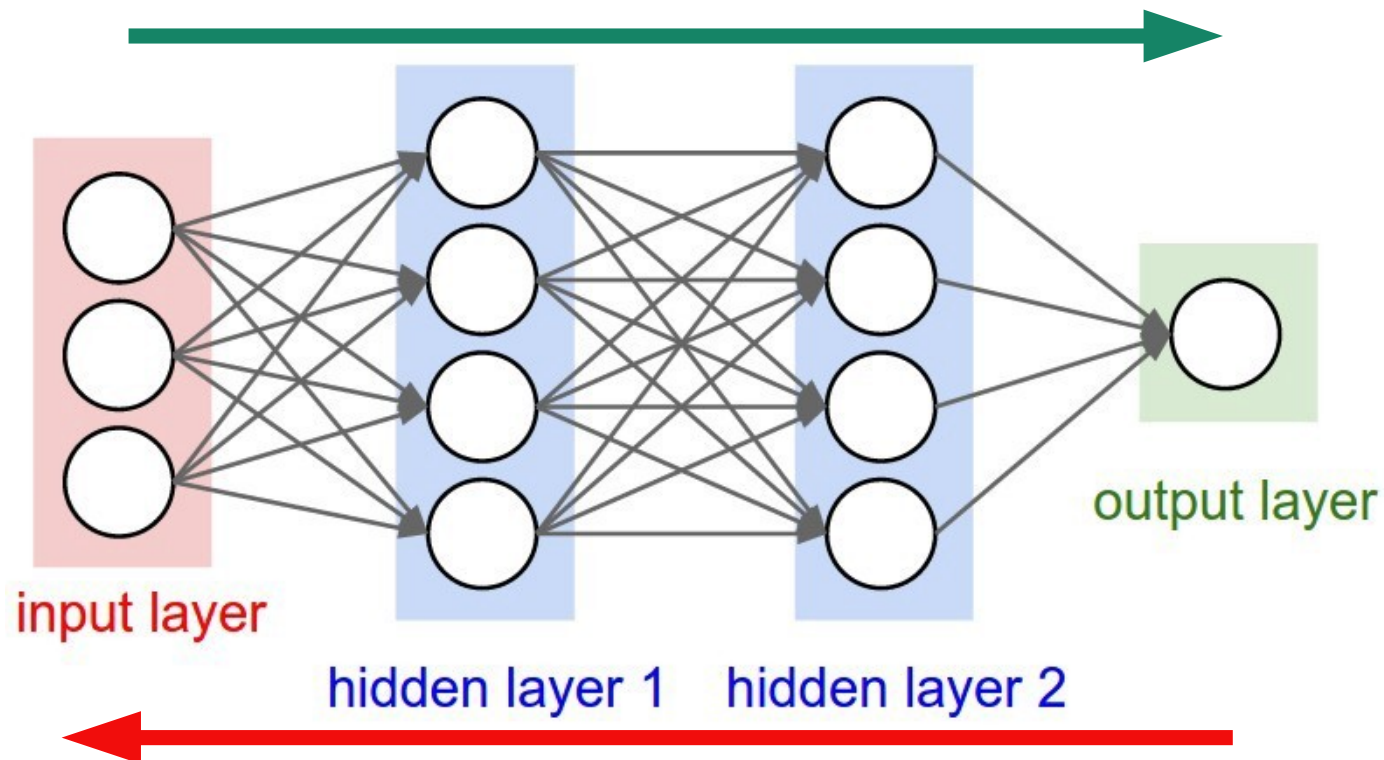
f - fonction sigmoïd représente la fonction de répartition de la loi logistique. Elle est **dérivable**, ce qui est une contrainte pour l'algorithme de **rétropropagation**



Exercice : écrivez l'équation de la fonction sigmoïd !

Introduction – Un Neurone

- **propagation** (*for-forwarding*)
- **rétropropagation** (*back-forwarding*)



Matériel pour IA embarquée

- Les **SoC** développés pour les applications de l'IA embarquée sont presque toujours basées sur les **CPU ARM**
- Ils intègrent les **GPUs** et/ou les **TPUs**
- Questions
 - Quelles sont les différences entre les **CPUs** et les **GPUs** ?
 - Quelles sont les différences entre les **GPUs** et les **TPUs** ?

Paramètres du notebook

Accélérateur matériel

None ▼ ⓘ

☐ Omettre l'élément de sortie des cellules de code lors de l'enregistrement de ce notebook

Annuler Enregistrer

Paramètres du notebook

Accélérateur matériel

GPU ▼ ⓘ

Pour tirer le meilleur parti de Colab, évitez d'utiliser un GPU si vous n'en avez pas besoin. [En savoir plus](#)

☐ Omettre l'élément de sortie des cellules de code lors de l'enregistrement de ce notebook

Annuler Enregistrer

Paramètres du notebook

Accélérateur matériel

TPU ▼ ⓘ

Pour tirer le meilleur parti de Colab, évitez d'utiliser un TPU si vous n'en avez pas besoin. [En savoir plus](#)

☐ Omettre l'élément de sortie des cellules de code lors de l'enregistrement de ce notebook

Annuler Enregistrer



Matériel pour IA embarquée : CPUs

- Modern **CPU Features Summary:**

- Has Several Cores

- **Specialized in Serial Processing**

- Capable of executing a **handful** of operations at once

- Have high speed FLOPS utilization

- Supports large models thanks to its large memory capacity

- **Acceleration: Multi-Core & SIMD (ARM-NEON)**

Paramètres du notebook

Accélérateur matériel

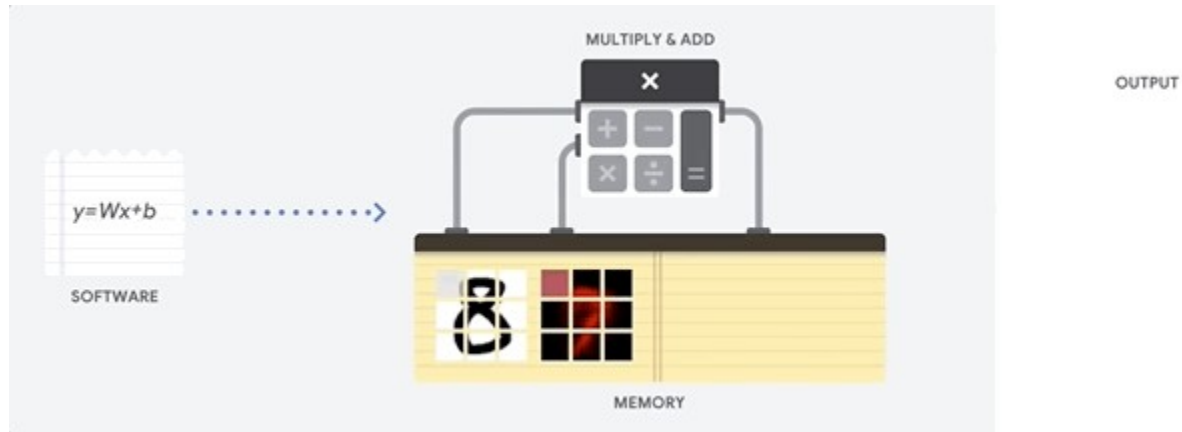
None ▾ ?

☐ Omettre l'élément de sortie des cellules de code lors de l'enregistrement de ce notebook

Annuler

Enregistrer

Matériel pour IA embarquée : CPUs





Matériel pour IA embarquée : GPUs

■ GPU Features Summary:

- Has **hundreds/thousands of cores**
- High memory throughput
- Specialized for parallel processing
- Capable of executing **thousands of operations** at once
- Taking into account integer and floating point operations

Paramètres du notebook

Accélérateur matériel

GPU  

Pour tirer le meilleur parti de Colab, évitez d'utiliser un GPU si vous n'en avez pas besoin. [En savoir plus](#)

☐ Omettre l'élément de sortie des cellules de code lors de l'enregistrement de ce notebook

Annuler [Enregistrer](#)

Matériel pour IA embarquée : GPUs





Matériel pour IA embarquée : TPUs

- **TPUs Features Summary:**
 - Special Hardware for **Matrix Processing**
 - High Latency (compared to CPU) ; long start
 - Very High Throughput : very large buses
 - Compute with Extreme Parallelism on smaller units (FP-16 bit)
 - Highly-optimized for large batches and CNNs (convolutional neural network)

Paramètres du notebook

Accélérateur matériel

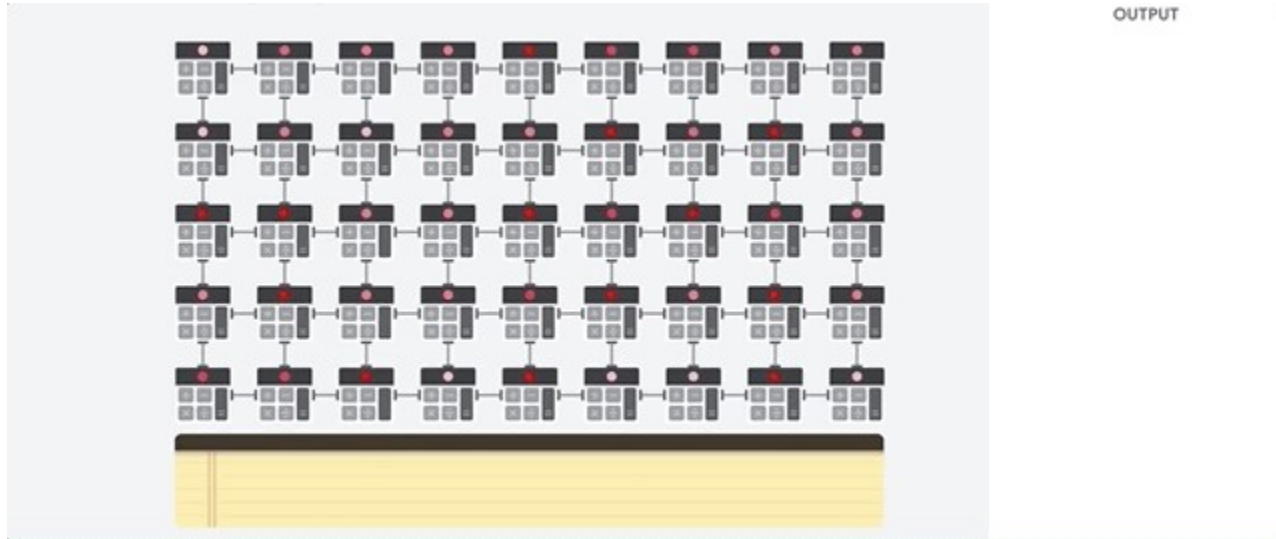
TPU  

Pour tirer le meilleur parti de Colab, évitez d'utiliser un TPU si vous n'en avez pas besoin. [En savoir plus](#)

☐ Omettre l'élément de sortie des cellules de code lors de l'enregistrement de ce notebook

Annuler [Enregistrer](#)

Matériel pour IA embarquée : TPUs



Dans un TPU, les ALUs sont **directement connectés** les uns aux autres sans utiliser la mémoire. Ils peuvent donner directement des informations de passe qui réduiront considérablement la latence.

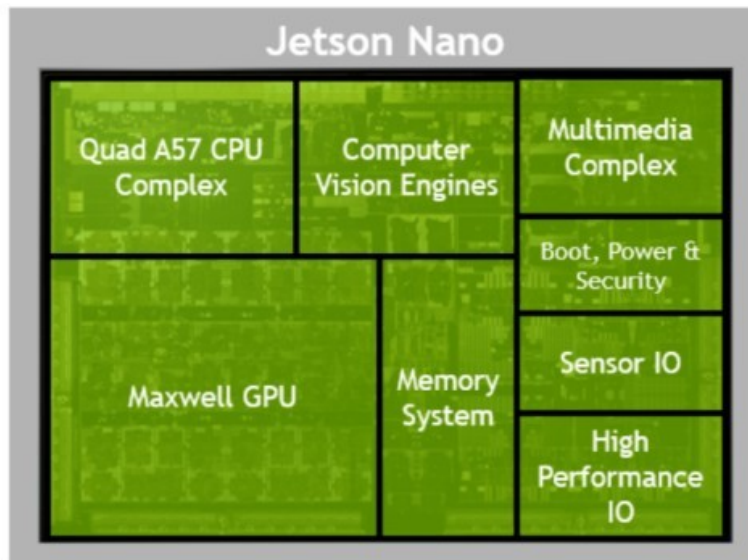
Matériel : Nvidia Jetson Nano

JETSON NANO Low Cost AI Computer Module

Heterogeneous CPU Complex
Quad Cores A57 with 2MB L2
for multi-threaded operation
1.43Ghz

Maxwell Tensor Core GPU
128 CUDA Tensor Cores
512 CUDA GFLOPS (FP16)

Memory
4GB 64-Bit LPDDR4
Bandwidth 25.6 GB/s
16GB eMMC



Computer Vision I
ISP
Video Image Com

Multimedia Engine
Encode 4k, 4x1080
Decode 4k60, 2x4K
JPEG encode & de
H.264, H.265, VP9
HDMI, DP and eDP

Boot, Power & Security
Boot and power management
ARM TrustZone Secure

Industry Standard IO
GPIO, I2C, I3S, SDIO, SPI, UART
Support up to 12 CSI @1.5Gbps

Industry Standard High-Speed IO
PCIe Gen2 rootport x1 | x2 | x4
RGWII Ethernet
USB 3.0 and 2.0
USB 3.0 Gen2 Host and Device

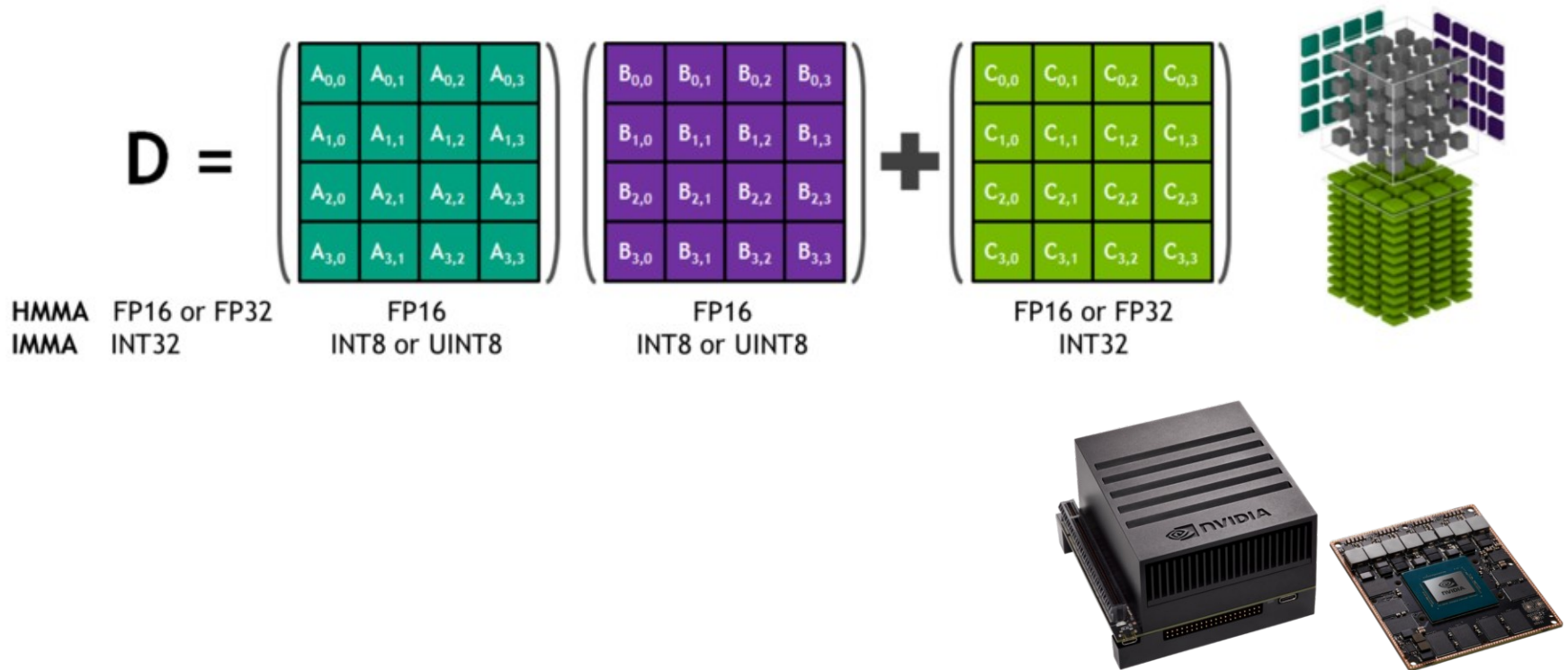


GPU Maxwell à 128 cores à 2 opérateurs – 32 bits (921 MHz)
performance = 0,5 TOPS (INT8)

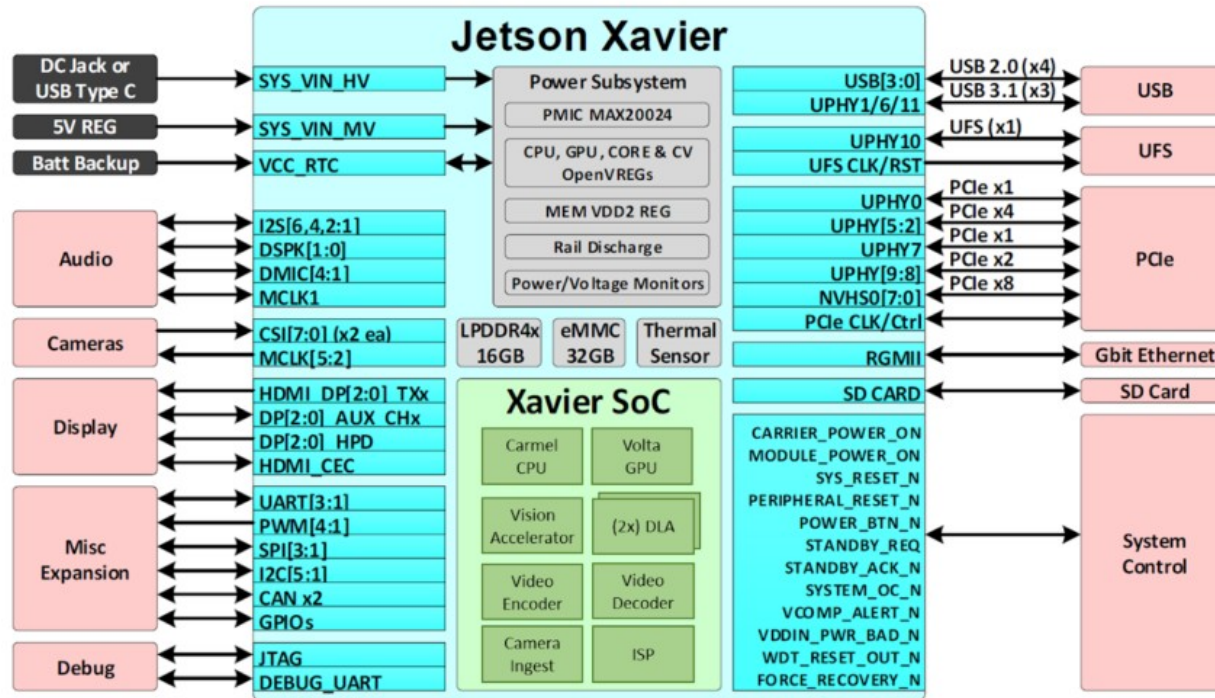
Matériel : Nvidia Jetson Xavier

Jetson Xavier – GPU Volta à 512 cores à 4 opérateurs 64 bits (1377 MHz) – TPU avec 64 Tensor cores

performance = 22 TOPS (INT8)



Matériel : Nvidia Xavier SoC



Attention : Jetson Xavier est un **SBC**, qui intègre un **SoC** Xavier de **11 mld** transistors (**7 nm**) .

Le coût de développement **2 mld de dollars**.

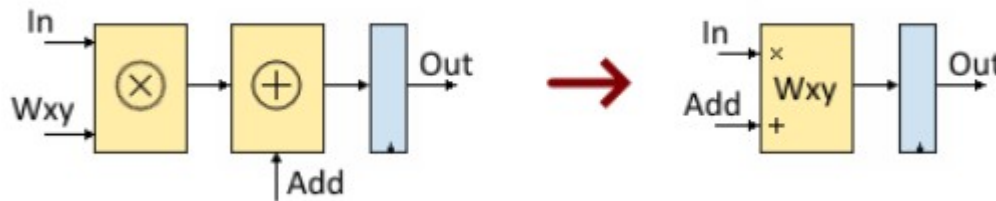
Matériel : Google Coral



Le **SoC Google Coral** utilise un **ASIC** fabriqué par l'équipe Google appelé **Edge TPU**.

Il utilise une **matrice systolique** pour effectuer les opérations type **tenseur**.

Les éléments de la matrice sont les cellules **mul-add**.



Matériel : Google Coral SBC

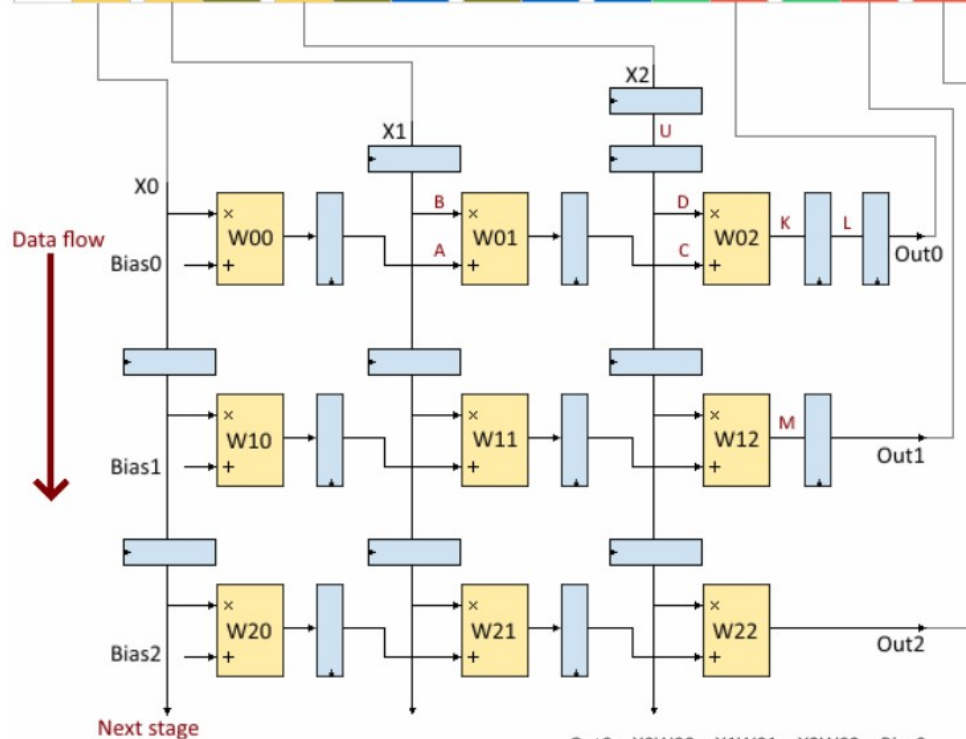


Le Google Coral mini SBC

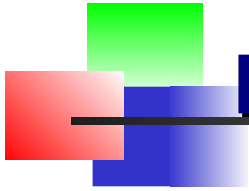
CPU	MediaTek 8167s SoC (Quad-core Arm Cortex-A35)
GPU	IMG PowerVR GE8300 (integrated in SoC)
ML accelerator	Google Edge TPU coprocessor: 4 TOPS (int8); 2 TOPS per watt
RAM	2 GB LPDDR3
Flash memory	8 GB eMMC
Wireless	Wi-Fi 5 (802.11a/b/g/n/ac); Bluetooth 5.0
Audio/video	3.5mm audio jack; digital PDM microphone; 2.54mm 2-pin speaker terminal; micro HDMI (1.4); 24-pin FFC connector for MIPI-CSI2 camera (4-lane); 24-pin FFC connector for MIPI-DSI display (4-lane)
Input/output	40-pin GPIO header; 2x USB Type-C (USB 2.0)

Matériel : Google Coral

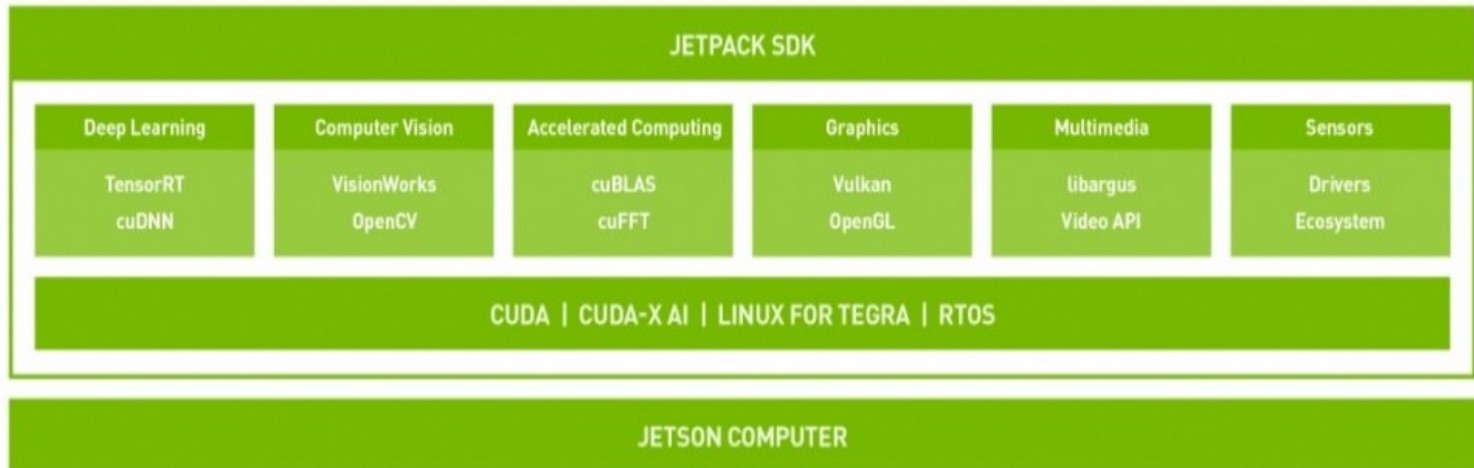
Clk	X0	X1	B	X2	U	D	A	C	K	L	Out0	M	Out1	Out2
0			?		?	?	?	?	?	?	?	?	?	?
1						?		?	?	?	?	?	?	?
2									?	?	?	?	?	?
3										?	?	?	?	?
4														
5														
6														



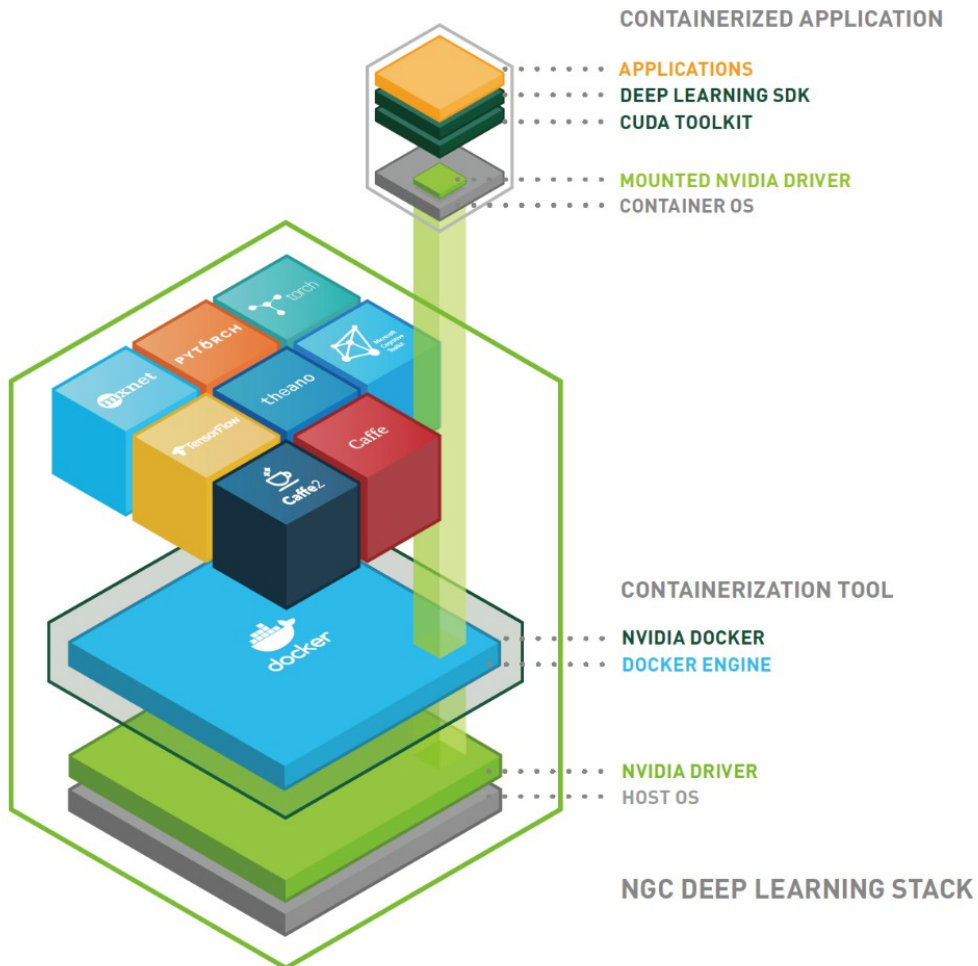
$$\begin{aligned} \text{Out0} &= X0W00 + X1W01 + X2W02 + \text{Bias0} \\ \text{Out1} &= X0W10 + X1W11 + X2W12 + \text{Bias1} \\ \text{Out2} &= X0W20 + X1W21 + X2W22 + \text{Bias2} \end{aligned}$$



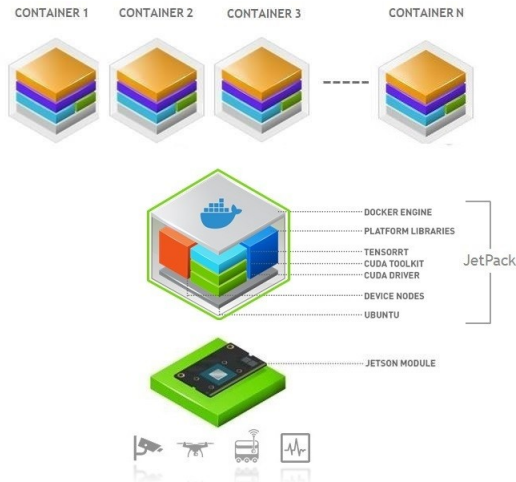
Logiciel : JetPack



Logiciel : ML Docker



Logiciel : ML Docker



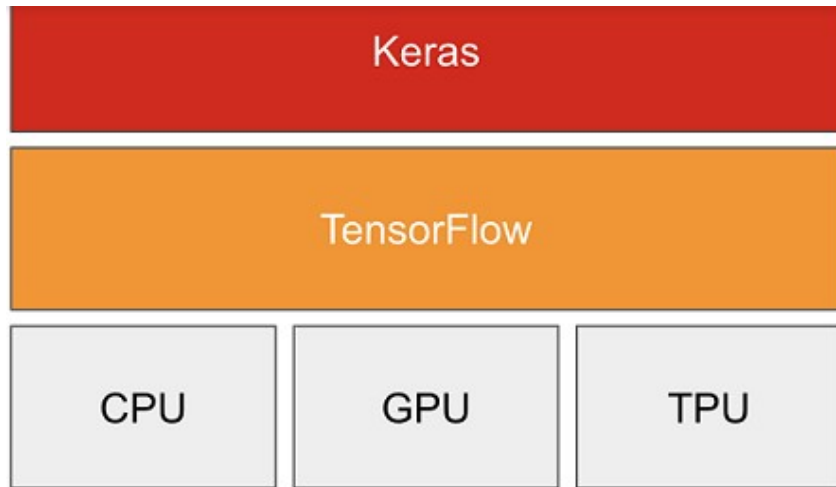
Machine Learning container contient : **TensorFlow**, **PyTorch**, **JupyterLab** et d'autres frameworks de ML et de science des données populaires tels que **scikit-learn**, **scipy** et **Pandas** préinstallés dans un environnement **Python 3.6**.

Jetson Inference container contient :

TensorRT et l'apprentissage par transfert avec **PyTorch** et les application de **classification d'images** pour **C++** ou **Python**, la **détection d'objets** et des **démonstrations de caméra en direct**



Logiciel : TensorFlow et Keras



Deep Learning ...


Layers, models, optimizers, losses, metrix, ..

Tensor Manipulation ...

Tensors, variables, differentiations, distribution, ..

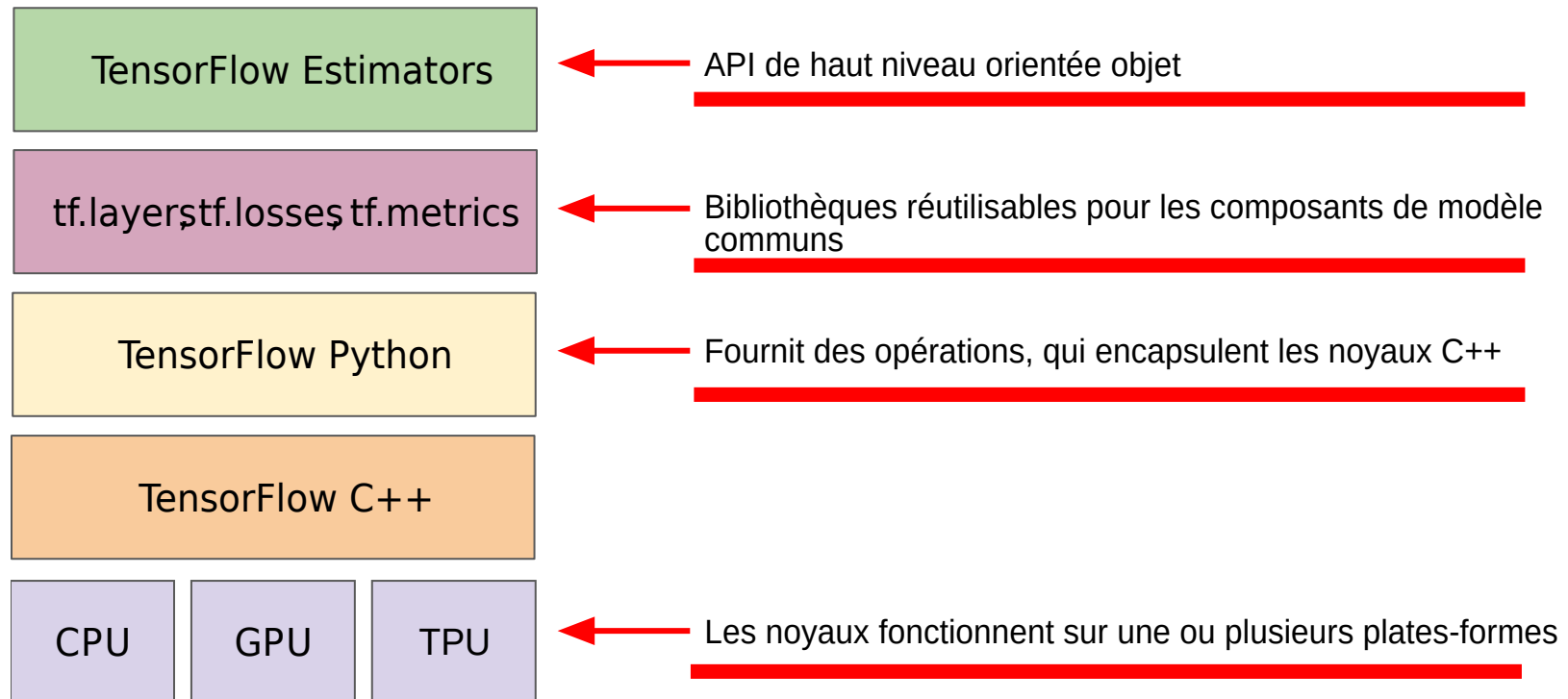
Coding ...

```
Import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
```



TensorFlow

Pour la développement nous allons utiliser les bibliothèques de **Python** les plus avancées pour le **ML** (Machine Learning) et **DL** (Deep Learning) , telles que **TensorFlow** et **Keras**.





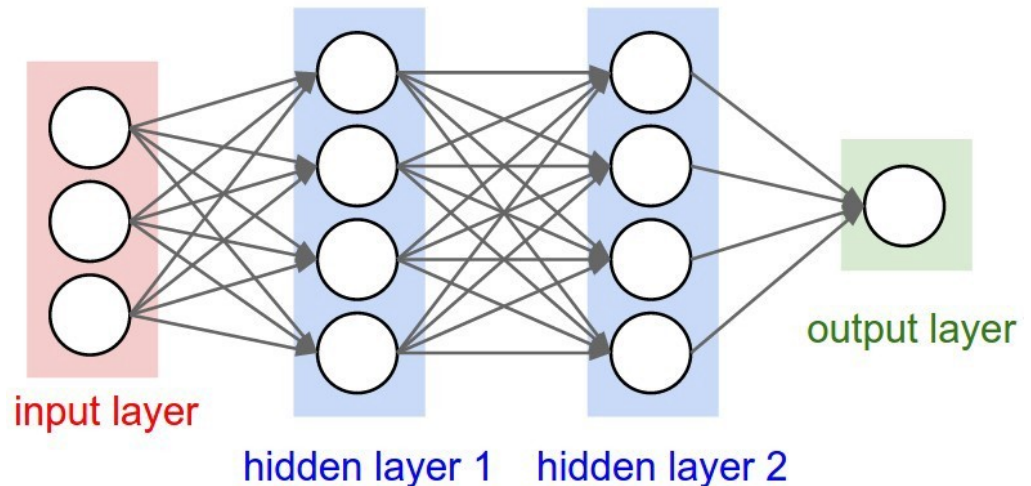
Keras fournit les composants pour créer les couches (layers) d'un réseau de neurones.

Couche d'entrée est conceptuellement différente des autres couches

Couche cachée 1: 4 unités (**4 neurones**)

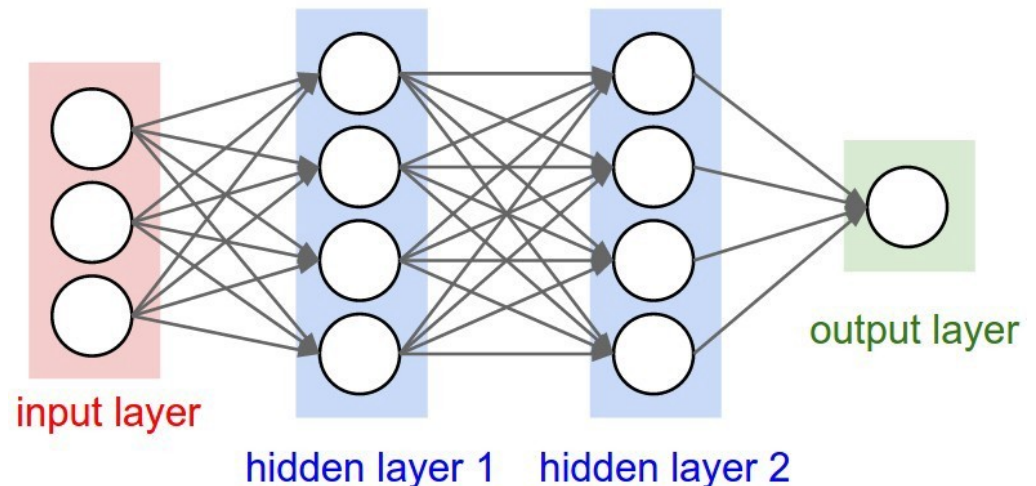
Couche cachée 2: 4 unités

Couche de sortie: 1 unité

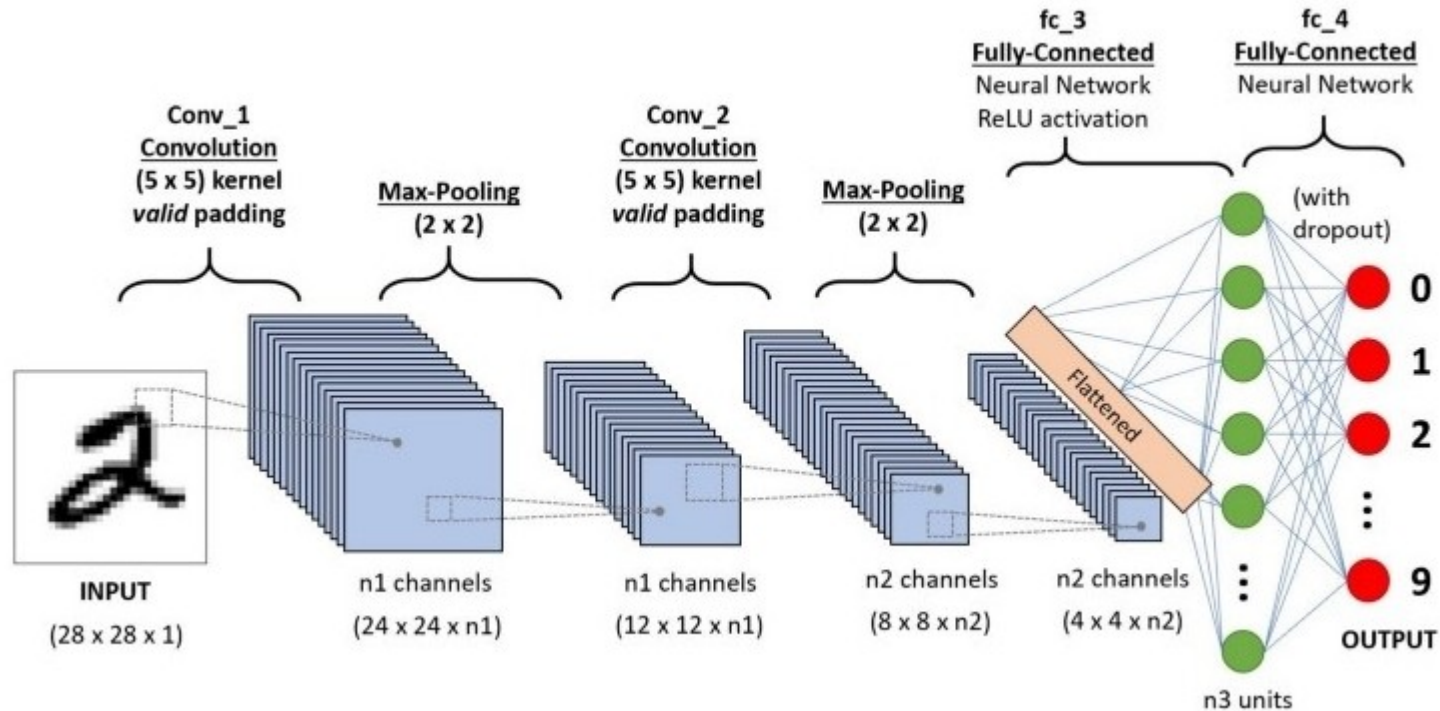


Keras : un simple modèle Dense

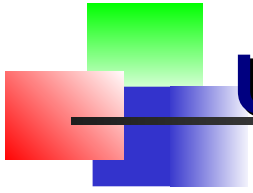
```
from keras.models import Sequential
from keras.layers import *
model = Sequential()
# input with shape=(3,)
model.add(Dense(units=4,input_shape=(3,))) #hidden layer 1
model.add(Dense(units=4)) #hidden layer 2
model.add(Dense(units=1)) #output layer
```



Un exemple de réseau neuronal



Un réseau de neurones convolutifs (**ConvNet/CNN**) qui peut prendre une image d'entrée, attribuer de l'importance (poids et biais apprenables) à divers aspects / objets de l'image et **être capable de différencier l'un de l'autre**.



Un résumé

Dans cette présentation nous avons parlé de:

- Le **domaine** de l'intelligence artificielle (IA)
- Les **caractéristiques** d'intelligence artificielle embarquée (IAE)
- Le **matériel** disponible pour l'implémentation des modèles d'IAE
- Le **logiciel** utilisé pour le développement/entraînement des modèles
- Nous avons donné quelques **exemples de modèles**