Tema 3.2
Hardware para Deep Learning

Miguel Ángel Martínez del Amor
Deep Learning
Departamento Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

Contenido

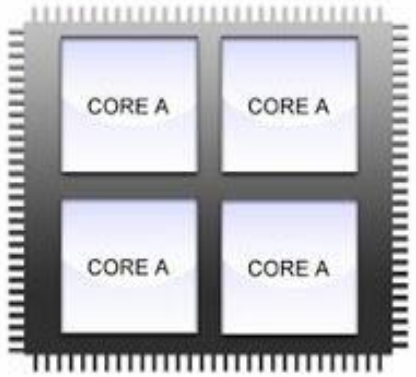
- Necesidad de cómputo paralelo
- Hardware paralelo:
  - CPUs multinúcleo
  - GPUs
  - TPUs
- Plataformas distribuidas:
  - Clusters
  - Cloud

Necesidad de cómputo paralelo

- Hacer **inferencia** (propagación hacia adelante) “no es demasiado” costoso.
- El **entrenamiento** mediante backpropagation es muy **costoso.**
- Una **red pequeña** puede ejecutarse sin problema en una **CPU**.
- Una **red profunda** puede usarse para **inferir** en una **CPU**.
- Sin embargo, cuando necesitamos **entrenar** una **red profunda** en un tiempo razonable (de meses a días), o hacer **inferencia** en **tiempo real** (cámaras de tráfico), necesitamos hardware **paralelo**.

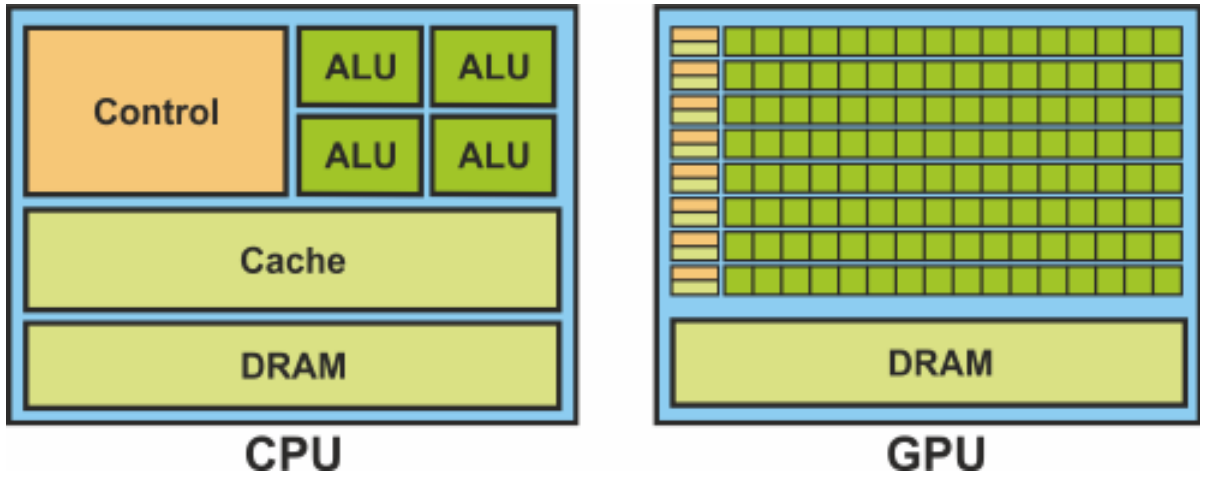
Hardware paralelo: CPUs multinúcleo

- Las CPUs actuales son multiprocesadores con varios núcleos (del orden de 4 a varias decenas).
- La librería por excelencia para programar procesadores **multinúcleo** (multicore) es **OpenMP**.
- Sin embargo, rendimiento **limitado**. No todos los frameworks de DL soportan OpenMP.
- Vendedores: Intel, AMD, ARM (móvil).



Hardware paralelo: GPUs

- **GPU** = Graphics Processing Unit (núcleo tarjeta gráfica).
- Con el tiempo este procesador ha evolucionado y hoy en día se puede usar para cómputo paralelo.
- Una GPU actual incluye del orden de cientos a miles de núcleos.
- Los núcleos son más básicos que los de una CPU, pero son muchos más!



Hardware paralelo: GPUs

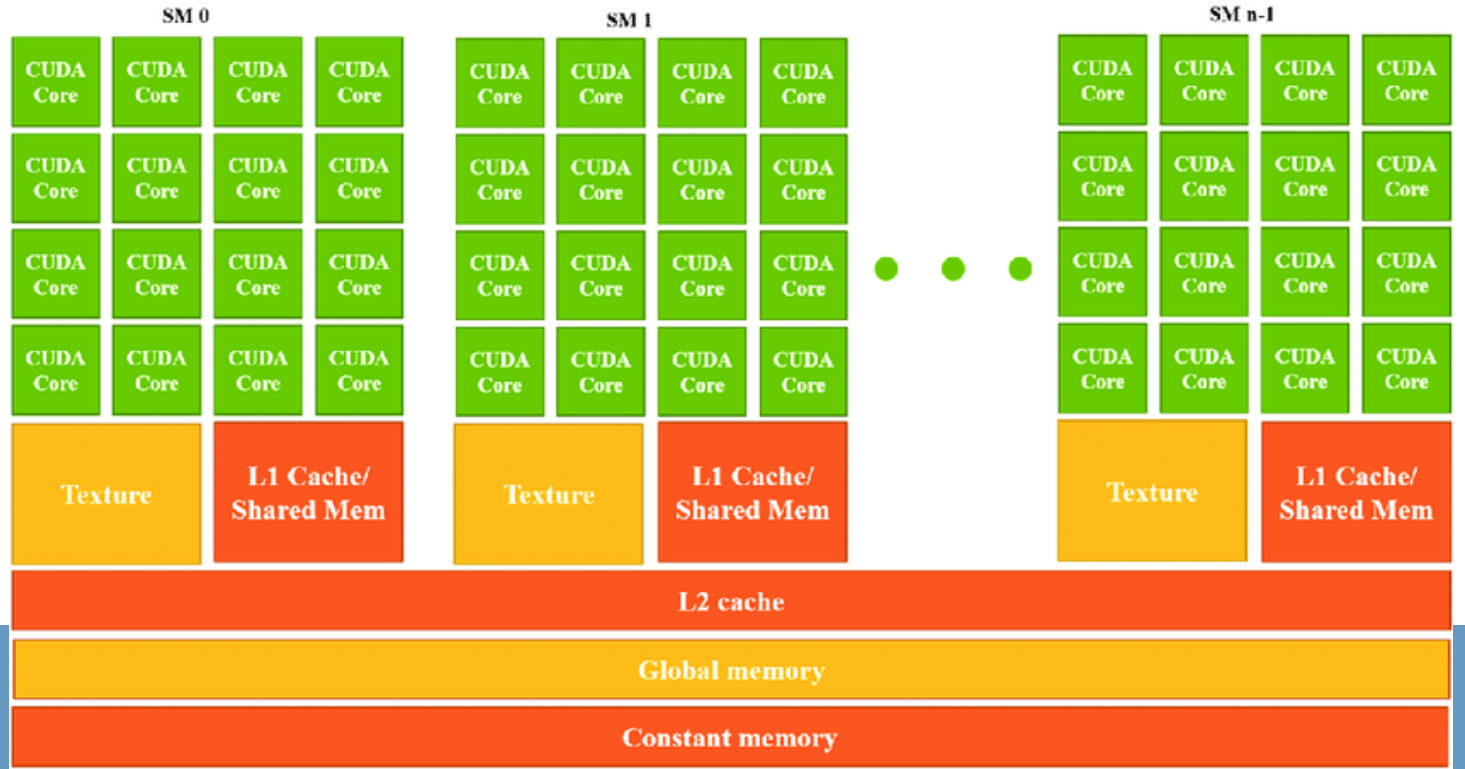
- Tecnología clave para Deep Learning.

Hardware paralelo: GPUs

- Librerías para programar GPUs:
  - CUDA (de NVIDIA)
  - OpenCL (Chronos → NVIDIA, AMD, Intel…)
- **NVIDIA** invirtió en Deep Learning desde el principio, y ahora ofrece un **mayor soporte** (todos los frameworks soportan de forma nativa GPUs mediante CUDA).
- OpenCL (y por tanto AMD) es soportado de manera experimental por algunos frameworks.

Hardware paralelo: GPUs

- Contenido de una GPU (en un vistazo):
  - Núcleos distribuidos en multi-procesadores, SM.
  - Memoria del dispositivo, local a la tarjeta, donde se disponen los datos que los núcleos van a procesar (copiados previamente!).

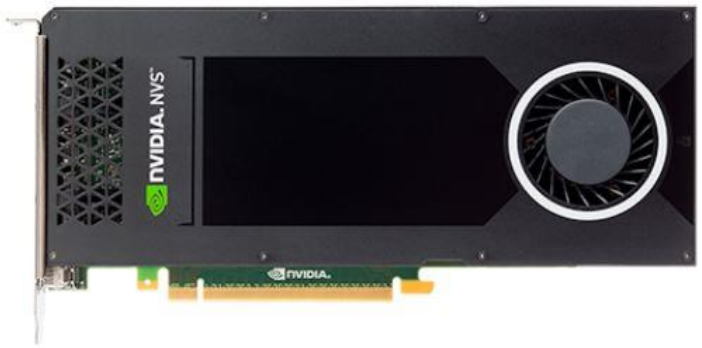


Hardware paralelo: GPUs

- ¿Qué hay que tener en cuenta al comprar una?
  1. **La cantidad de memoria** en la GPU: cuanto más GigaBytes, mayores modelos se podrán entrenar, y se podrán enviar batches de ejemplos más grandes.
  2. Una **memoria** de la tarjeta **rápida**: el acceso a los datos por los núcleos debe ser eficiente (últimos GDDR6, HBM2).
  3. El **número** de **núcleos**: cuantos más, mayor paralelización.
  4. La **tasa de transferencia** de datos a la tarjeta: los ejemplos fluirán continuamente de la CPU a la GPU, y por tanto necesitamos un bus rápido (PCI-e 3.0 x16, NVLink).
- Instalación: **CUDA** Toolkit, CUDA driver y **CuDNN**

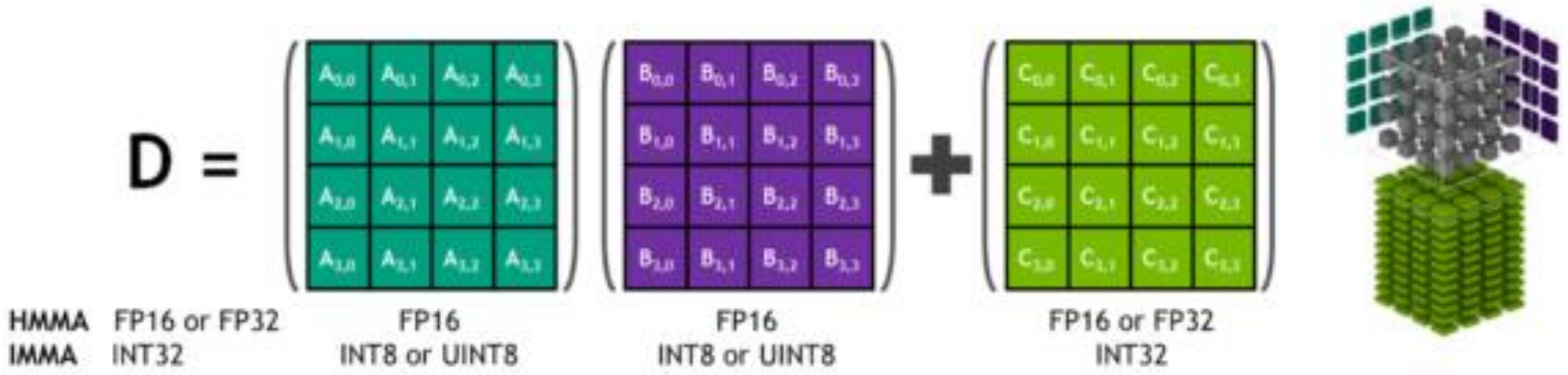
Hardware paralelo: GPUs

- **NVIDIA** nombra cada **generación** de arquitecturas con un científico:
  - Fermi (descatalogado), Kepler, Maxwell, Pascal, Volta, Turing, …
- Las distintas **categorías** de tarjetas:
  - Tesla (cálculo científico, muy caro)
  - GeForce (videojuegos, asequible)
  - Quadro (gráficos profesionales, caro)
  - Jetson (robótica)
- Es posible tener entornos **MultiGPU** (con varias GPUs).



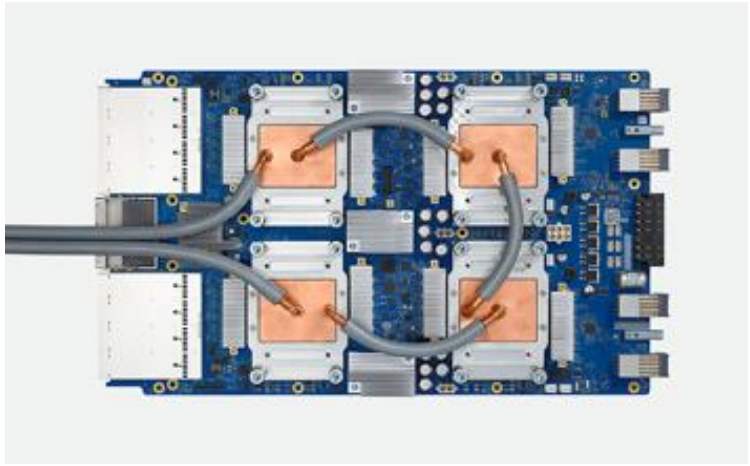
Hardware paralelo: GPUs

- En **2017**, en la arquitectura **Volta**, NVIDIA introdujo unos núcleos nuevos dentro de sus tarjetas (a parte de los dedicados para CUDA) para solo Deep Learning.
- **Tensor Cores**: procesadores para cálculo con tensores.
- **Half precision**: float de 16 bits, int de 8 bits.
- Efectivos con redes convolucionales (las veremos).



Hardware paralelo: TPUs

- **TPU** = Tensor Processing Unit
- **Chip** introducido por Google en 2016 para Deep Learning.
- **Especializados** para acelerar cálculo tensorial (algebra lineal).
- Uso desde **TensorFlow y PyTorch**
- En fase **beta**:
  - De Gigas a Tera Bytes de memoria.
  - De cientos a miles de núcleos.
- Disponible en Cloud: 8$/hora



Plataformas distribuidas: cloud

- Diversas empresas ofrecen servicios donde entrenar nuestros modelos, **evitándonos** tener que tener **hardware** para ello.
- Tecnologías basadas en **contenedores** (Dockers) o **máquinas virtuales**.
- Los tres principales proveedores:
  - Google Cloud
  - Amazon AWS
  - Microsoft Azure

- Google Colaboratory:
  - Basado tan solo en notebooks, requiere cuenta en Google Drive.
  - Gratuito, ofrece GPUs y TPUs.

Plataformas distribuidas: cloud

- Google colab:
  - Acceso gratuito a Tesla T4 y TPU.
  - El trabajo se pierde cuando se cierra sesión: guardar el modelo y datos en Google Drive.
- Google cloud:
  - El trabajo se mantiene.
  - 0,8$/hora para V100. 0,35$/hora para T4.
- AWS EC2:
  - El trabajo se mantiene, y ofrece servicios adicionales.
  - 0,9$/hora para K80. 3$/hora para V100. Spot instance (recursos bajo demanda) 1,2$/hora.

Recapitulación

- Para poder entrenar modelos grandes sobre conjuntos de datos grandes necesitaremos de **potencia computacional.**
- Las **GPUs** proveen una solución hardware eficiente respecto a lo que cuestan. Las TPUs aún están por distribuirse.
- Si no se dispone capacidad de compra de GPUs, mejor usar un servicio **en la nube** (Google Colab?).