Universitat d'Alacant
Universidad de Alicante

Retos futuros de la Computación de Alto Rendimiento 2 (CAR)

Asignatura: Computación de Alto Rendimiento (CAR)

Profesor: Ricardo Moreno Rodríguez

Universitat d'Alacant
Universidad de Alicante

Índice

Contenido

Universitat d'Alacant
Universidad de Alicante

1. Introducción

En esta unidad profundizaremos sobre los conceptos fundamentales de la **computación en la nube**, la **computación de alto rendimiento (HPC)** y su relación con la **inteligencia artificial (IA)**. A modo de hilo conductor, utilizaremos ejemplos prácticos con herramientas modernas de IA, como el detector de objetos **YOLOv8**, que se irán introduciendo progresivamente a lo largo de los contenidos teóricos y las prácticas. Esto nos permitirá ilustrar cómo los servicios en la nube y las técnicas de HPC pueden facilitar la ejecución de cargas de trabajo de IA intensivas, dando contexto práctico a los conceptos presentados.

Comenzaremos definiendo el paradigma del **cloud computing** y revisando sus modelos de servicio, para luego explorar tecnologías habilitadoras como la virtualización y los contenedores. A continuación, nos adentraremos en el **campo de la computación de alto rendimiento en la nube**, analizando cómo la nube permite acceder a recursos HPC, las arquitecturas para grandes cargas de trabajo (*Big Compute*), el escalado automático de recursos y conceptos avanzados como el aprendizaje federado. También veremos servicios específicos que ofrecen los proveedores cloud para **IA** y **Big Data**, así como una pincelada sobre computación cuántica aplicada a la IA. Todos estos temas se conectarán con el ejemplo práctico de YOLOv8, que nos permitirá entender de forma aplicada cómo aprovechar la nube para tareas intensivas de procesamiento de imágenes.

2. El Paradigma Cloud Computing

Cloud computing (computación en la nube) es un modelo de computación que ofrece acceso mediante la red, bajo demanda y de forma ubicua, a un conjunto compartido de recursos configurables (servidores, almacenamiento, aplicaciones, servicios) que pueden provisionarse y liberarse rápidamente con un mínimo esfuerzo de gestión .

En palabras del NIST (National Institute of Standards and Technology),

> *"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."*

Es decir, la computación en la nube permite **externalizar servicios de TI** completos: en lugar de mantener infraestructura propia, las organizaciones pueden usar recursos ofrecidos por terceros a través de internet, pagando solo por lo que consumen.

3. Externalización de servicios y *Mobile/Edge Computing*

El paradigma de cloud computing supone la **externalización del procesamiento, del almacenamiento y de los servicios** de TI. Por ejemplo, una empresa puede delegar en la nube el procesamiento de sus aplicaciones intensivas, guardar sus datos en servicios de almacenamiento cloud, o usar servicios en línea (correo, CRM, etc.) en lugar de instalarlos localmente. Esta externalización se ha extendido también a contextos móviles e IoT bajo el paradigma de **Mobile Cloud Computing (MCC)**, en el cual dispositivos con recursos limitados (como smartphones o sensores IoT) aprovechan la nube para procesar datos pesados. Conceptos relacionados son el **Edge Computing** y *Fog Computing*, donde parte del procesamiento se realiza en la periferia de la red (por ejemplo, en gateways locales o en los propios dispositivos) para reducir la latencia y el consumo de ancho de banda, complementando así a la nube.

En arquitecturas modernas suele haber múltiples niveles: dispositivos IoT, nodos de borde (edge), y la nube central, creando una **arquitectura multinivel** de computación en la nube . Esto es especialmente útil en aplicaciones de tiempo real, como coches autónomos o ciudades inteligentes, donde se requiere procesar datos muy cerca de donde se generan para reaccionar con rapidez, manteniendo a la vez la nube como respaldo para procesamiento intensivo o almacenamiento a largo plazo.

Ejemplos:

En el ámbito de *Smart Cities*, se han propuesto arquitecturas distribuidas multinivel para aplicaciones urbanas intensivas en cómputo.

Por ejemplo, **un sistema de iluminación inteligente de ciudad** (iluminación adaptativa en calles) puede usar cámaras en el borde para detectar presencia de personas/vehículos y una capa cloud para coordinación general . En ese caso, los cálculos de visión artificial (como la detección de personas con un modelo tipo **YOLOv8**) podrían realizarse en dispositivos locales potentes o en servidores edge cercanos, mientras que la nube agregaría datos de muchas cámaras para análisis históricos o entrenar mejores modelos .

Otro ejemplo es **el coche autónomo**: estos vehículos generan enormes cantidades de datos de sensores que no siempre pueden enviarse completamente a la nube por limitaciones de latencia y ancho de banda. Las arquitecturas modernas emplean edge computing en el propio vehículo o en estaciones base cercanas para procesar en tiempo real las situaciones de conducción, apoyándose en la nube solo para tareas menos urgentes como actualizar modelos de IA o almacenar información . De este modo, se combina la inmediatez del procesamiento local con la capacidad prácticamente ilimitada del cloud cuando se requiere.

En ambos casos, **cloud computing actúa como facilitador** para extender las capacidades de dispositivos limitados, posibilitando aplicaciones innovadoras en IA, Big Data, simulaciones, etc., que requieren **potencia de cómputo flexible y escalable**.

4. Ventajas y desventajas del cloud computing

El paradigma de computación en la nube aporta una serie of **ventajas** significativas:

- **Abstracción de los recursos:** El usuario no necesita conocer detalles de la infraestructura física subyacente. La nube provee una capa de abstracción que simplifica la gestión de recursos de cómputo, almacenamiento y red .
- **Escalabilidad y elasticidad:** Es posible aumentar o disminuir recursos (CPU, memoria, almacenamiento) bajo demanda de forma rápida. Esto permite adaptarse dinámicamente a cargas variables, manteniendo un rendimiento adecuado sin invertir en infraestructura propia sobredimensionada .
- **Pago por uso:** Los modelos de servicio suelen ser *pay-as-you-go*, es decir, se paga únicamente por los recursos consumidos (tiempo de cómputo, cantidad de datos almacenados o transferidos, etc.). Esto convierte gastos de capital en gastos operativos más predecibles .
- **Acceso ubicuo:** Los servicios cloud están disponibles desde cualquier lugar con conexión a internet. Esto favorece el teletrabajo, la colaboración global y el acceso móvil a aplicaciones y datos .
- **Seguridad y fiabilidad:** Aunque a veces se perciba lo contrario, los grandes proveedores cloud invierten fuertemente en medidas de **seguridad**, copias de respaldo y alta disponibilidad. Para muchas organizaciones, migrar a la nube puede mejorar su postura de seguridad y continuidad del negocio al aprovechar estas inversiones del proveedor.

Por otro lado, también existen **inconvenientes o desafíos** asociados al cloud computing:

- **Infraestructura compartida y potencial menor rendimiento:** En entornos de nube pública los recursos físicos son compartidos entre muchos clientes. Esto puede implicar variabilidad en el rendimiento (por contención de recursos) y la imposibilidad de optimizar hardware específico para una carga de trabajo particular.
- **Dependencia del proveedor (lock-in):** Al construir servicios sobre una plataforma de nube específica, puede ser difícil migrar luego a otra. Esta **dependencia tecnológica** es un riesgo si el proveedor cambia condiciones, sufre caídas o no ofrece alguna característica necesaria .
- **Deslocalización de la información y pérdida de control:** Los datos se almacenan en ubicaciones que controla el proveedor. La organización pierde control directo sobre la localización exacta y el manejo de sus datos, debiendo

confiar en contratos de nivel de servicio (SLA) para asegurar calidad, privacidad y disponibilidad .

- **Riesgo de fuga de información:** Al residir los datos en entornos externos, aumenta la superficie de ataque y posibles brechas de seguridad. Aunque los proveedores suelen tener altos estándares, siempre existe riesgo de accesos no autorizados o errores de configuración que expongan información sensible .
- **Dilución de ventaja competitiva:** En ciertos casos, delegar infraestructura clave a la nube (usada también por competidores) puede hacer que la tecnología deje de ser un diferenciador propio. Por ejemplo, si todas las empresas de un sector usan la misma plataforma cloud para analítica, puede ser más difícil obtener una ventaja exclusiva con la infraestructura .

En balance, la computación en la nube representa un **cambio de paradigma** en cómo las organizaciones conciben sus recursos de TI, ofreciendo grandes beneficios de agilidad y costo, pero requiriendo considerar cuidadosamente aspectos de seguridad, rendimiento y dependencia. En la actualidad, el cloud computing actúa además como **base habilitadora** para muchas otras tendencias tecnológicas, desde la **IA y Big Data** hasta IoT y HPC, al proporcionarles los recursos flexibles que necesitan .

5. Modelos de Despliegue de la Nube

No todas las nubes son iguales. Según quién posea la infraestructura y cómo se gestione, existen distintos **modelos de despliegue de cloud** . Los principales son: **nube privada, nube pública, nube híbrida**, y en años recientes han surgido también esquemas multi-nube y federaciones de nubes.

5.1 Nube privada (Private Cloud)

La infraestructura cloud es utilizada **exclusivamente por una organización** (y sus unidades de negocio). Puede ser propiedad de la propia organización o de un tercero, y puede estar alojada tanto on-premise (en las instalaciones de la empresa) como externamente. La característica clave es que los recursos *no* se comparten con usuarios externos; esto permite mayor control y personalización a costa de perder algunas ventajas de escalado masivo. Por ejemplo, una empresa puede montar su propia nube privada con tecnología OpenStack para uso interno.

5.2 Nube Pública (Public Cloud)

La infraestructura está disponible para el **público en general**, perteneciendo y siendo operada por un proveedor externo (empresas como Amazon, Microsoft, Google, etc.) . Los servicios se ofrecen a cualquier usuario o empresa bajo demanda a través de internet, normalmente con grandes centros de datos multi-inquilino. Es el modelo más común

asociado al término "cloud": recursos compartidos a gran escala, accesibles públicamente (mediante suscripción o pago por uso).

5.3 Nube Híbrida (Hybrid Cloud)

Combina dos o más nubes **distintas** (privadas y/o públicas) que permanecen como entidades únicas pero están interconectadas, permitiendo portabilidad de datos y aplicaciones entre ellas . Por ejemplo, una empresa puede mantener ciertos datos muy sensibles en una nube privada on-premise, pero usar una nube pública para picos de demanda o para exponer servicios públicos, orquestando la comunicación entre ambas. La clave es la interacción coordinada entre las nubes, por ejemplo usando tecnología que balancee carga entre la parte privada y la pública.

Además de estos, se utilizan cada vez más esquemas **Multi-Cloud** y **Cloud Federada**:

- **Multi-Cloud:** Se refiere al uso concurrente de servicios de **más de un proveedor de nube para una solución**. En lugar de depender de un solo proveedor, la organización aprovecha lo mejor de varios (por ejemplo, almacenar datos en AWS pero hacer análisis de Big Data en Azure). Esto reduce dependencia tecnológica y puede mejorar flexibilidad, pero añade complejidad de gestión y potencialmente costo . Para manejar entornos multi-nube a menudo se emplean **orquestadores** capaces de gestionar recursos en diferentes plataformas de manera unificada .
- **Nube Federada:** En una federación de nubes, varios proveedores **interoperan compartiendo servicios de forma transparente** para el usuario . Los recursos de múltiples clouds se integran bajo ciertos acuerdos, **facilitando al cliente acceso a un ecosistema más amplio**. Por ejemplo, en Europa está surgiendo el proyecto **Gaia-X**, una federación de nubes europea que busca interoperabilidad y estándares abiertos entre proveedores regionales. En las federaciones suele haber entidades denominadas **Cloud Broker**, que negocian las condiciones entre los distintos proveedores y presentan un catálogo unificado de servicios al usuario final . Estos brokers también ayudan a gestionar los **Acuerdos de Nivel de Servicio (SLA)** entre todos los participantes, asegurando que se cumplen los compromisos de rendimiento, disponibilidad, seguridad, etc. . La nube federada promete mayor eficiencia y escalabilidad al combinar recursos, aunque implica grandes desafíos en interoperabilidad y confianza entre proveedores .

6. Modelos de Servicio en la Nube

Otra forma de clasificar la computación en la nube es por el **modelo de servicio** que se ofrece. Esta clasificación describe qué tipo de capacidad o funcionalidad se entrega al usuario y cuánta gestión debe encargarse el cliente versus el proveedor. Tradicionalmente se mencionan tres niveles: **Infraestructura, Plataforma y Software como servicio**

(IaaS, PaaS, SaaS), a los que en años recientes se añade un cuarto modelo emergente: **Función como servicio (FaaS)** o *Serverless*. Veamos cada uno:

6.1 IaaS (Infrastructure as a Service, Infraestructura como Servicio)

El proveedor ofrece recursos de cómputo fundamentales – procesadores virtuales, máquinas virtuales, almacenamiento, redes – como servicios básicos . El cliente puede desplegar y ejecutar software arbitrario (sistemas operativos, aplicaciones) sobre esa infraestructura virtualizada. En IaaS **el cliente** tiene control sobre los SO, el almacenamiento, las aplicaciones y, a veces, ciertos componentes de red (como firewalls virtuales), pero **no gestiona** la infraestructura física subyacente de la nube . Ventajas: gran flexibilidad en la gestión de la infraestructura de TI sin tener que poseerla físicamente, lo que ahorra costos y mejora la eficiencia de recursos (por ejemplo, apagar VM cuando no se usan). Inconveniente: aún requiere que el cliente tenga conocimientos para administrar esas máquinas virtuales y conlleva cierta dependencia del proveedor en cuanto a disponibilidad y posibles subidas de precio . *Ejemplos:* Amazon EC2, Microsoft Azure (máquinas virtuales), Google Compute Engine – todos ellos permiten lanzar VM en la nube configurables a medida.

6.2 PaaS (Platform as a Service, Plataforma como Servicio)

En este modelo se proporciona no solo infraestructura sino también un **entorno de desarrollo y ejecución** completo listo para usar . Es decir, **el proveedor ofrece una plataforma con un stack de software** (sistema operativo, middleware, base de datos, runtimes) sobre la cual el cliente puede desarrollar, desplegar y gestionar sus aplicaciones, sin tener que ocuparse de la configuración del sistema operativo o del servidor de aplicaciones. El usuario controla sus aplicaciones y, a veces, ajustes de configuración del entorno, pero **no tiene que administrar** la infraestructura subyacente (red, servidores, almacenamiento). Ventajas: acelera el desarrollo al externalizar la gestión de la plataforma, permitiendo enfocarse en la lógica de negocio; mejora la productividad y reduce costes de mantenimiento (no hay que parchear SO ni gestionar bases de datos manualmente). Inconveniente: puede generar **dependencia del proveedor** a nivel de plataforma (por ejemplo, estar atado a un determinado framework o servicio propietario) . *Ejemplo:* **Google App Engine**, donde los desarrolladores suben su código y Google se encarga de ejecutarlo escalando automáticamente. Otro ejemplo local es **Velneo** , plataforma PaaS española para desarrollar aplicaciones empresariales, donde todo (servidor, base de datos, etc.) lo gestiona el proveedor y el cliente solo se preocupa de la lógica de la aplicación.

6.3 SaaS (Software as a Service, Software como Servicio)

Es el modelo más conocido por usuarios finales. El proveedor ofrece aplicaciones de **software completas** ejecutándose en su nube, accesibles al cliente normalmente a través de internet (por ejemplo, vía un navegador web o app ligera) . El cliente simplemente

utiliza la aplicación; no gestiona la infraestructura ni la plataforma subyacente, e incluso la configuración de la aplicación es limitada salvo opciones menores dentro de la propia aplicación . Ventajas: el software está siempre actualizado sin que el usuario deba instalar nada, y puede accederse globalmente; además, suele posibilitar la colaboración y uniformidad (todos los usuarios usan la misma versión) . Inconvenientes: **dependencia del proveedor** tanto en disponibilidad (requiere internet y que el servicio esté activo) como en confianza (los datos y procesos residen con el proveedor). También existen preocupaciones de seguridad y privacidad al ser servicios compartidos; tecnologías como la *criptografía homomórfica* se investigan para mitigar esto, permitiendo procesar datos cifrados sin desencriptar completamente . *Ejemplos:* Microsoft Office 365, Google Workspace (Gmail, Docs, etc.), Salesforce CRM – en todos ellos el usuario consume software directamente como un servicio web.

6.4 FaaS (Function as a Service, Función como Servicio)

Conocido también como **computación *serverless***, es un modelo más reciente en el que las aplicaciones se dividen en funciones individuales que se ejecutan **bajo demanda** en la infraestructura del proveedor . El usuario escribe funciones (código) y las despliega; el proveedor las ejecuta automáticamente *solo cuando son invocadas*, escalando hacia arriba o abajo según sea necesario, y gestionando completamente los recursos de cómputo subyacentes . En FaaS, el cliente no gestiona servidores en absoluto (ni virtuales ni físicos) ni procesos en segundo plano: simplemente define la lógica y el proveedor asegura su ejecución cuando corresponda.

Ventajas: gran agilidad y escalabilidad, ya que las funciones pueden replicarse masivamente en paralelo si llegan muchas peticiones, y **coste eficiente** – solo se paga por el tiempo de ejecución real de las funciones (no por tener un servidor levantado 24/7) . Además, simplifica el despliegue al abstraer por completo el servidor. Inconvenientes: diseñar arquitecturas serverless eficientemente puede ser complejo, pues implica dividir la aplicación en componentes muy pequeños y manejar bien su estado; también aumenta la dependencia del proveedor, ya que la aplicación se construye alrededor de servicios específicos de función en la nube .

Ejemplos: AWS Lambda, Google Cloud Functions, **Azure Functions** – estos servicios permiten subir una porción de código (una función) y configuran automáticamente los recursos para ejecutarla cuando sea necesario . Este modelo es ideal para eventos eventuales o cargas variables, por ejemplo procesar eventos de IoT, responder a peticiones web esporádicas, o tareas de procesamiento en lotes desencadenadas por algún suceso.

(Nota: Además de estos modelos principales, hoy prácticamente cualquier cosa se ofrece "as a Service". Encontramos *Database as a Service*, *Backend as a Service*, *AI as a Service*, etc. El concepto general es siempre delegar en un proveedor una capa de la pila

tecnológica. En nuestra unidad nos centraremos en IaaS, PaaS, SaaS y FaaS por ser los más relevantes para desplegar soluciones de IA y HPC en la nube.)*

7. Virtualización

La **virtualización** es una tecnología clave que posibilita el cloud computing tal como lo conocemos. Consiste en abstraer los recursos físicos de cómputo en múltiples *entidades virtuales* aisladas. Un mismo servidor físico puede albergar varias **máquinas virtuales (VMs)**, cada una con su propio sistema operativo y aplicaciones, funcionando como si fuesen máquinas independientes, cuando en realidad comparten el hardware subyacente. Esto se logra mediante una capa de software llamada **hypervisor** (o monitor de máquina virtual) , que es el encargado de distribuir y aislar los recursos del hardware (CPU, memoria, almacenamiento, dispositivos) entre las distintas VMs.

Ventajas de la virtualización: Permite un uso más eficiente del hardware – en lugar de tener servidores infrautilizados dedicados a una sola aplicación, se pueden consolidar varias VM en el mismo servidor aumentando la utilización. También posibilita flexibilidad y portabilidad: una máquina virtual es básicamente un conjunto de archivos (imagen de disco, configuración) que se puede migrar o copiar a otro host fácilmente, facilitando backup, recuperación ante desastres o balanceo de carga. Además, al estar aisladas, las VMs ofrecen seguridad y estabilidad: un fallo o intrusión en una VM normalmente no afecta a las demás.

Para los administradores, la virtualización **centraliza la gestión** de hardware y reduce costos de energía y equipamiento al necesitar menos máquinas físicas para la misma carga de trabajo.

Existen **diferentes tipos de hypervisors**:

- *Tipo 1 (bare-metal):* Se instalan directamente sobre el hardware, reemplazando al sistema operativo tradicional. Ofrecen alto rendimiento y son usados en entornos de servidores y data centers. *Ejemplos:* VMware ESXi, Microsoft Hyper-V (en modo servidor), XenServer.
- *Tipo 2 (hosted):* Se ejecutan como una aplicación sobre un sistema operativo anfitrión. Son comunes para virtualización de escritorio. *Ejemplos:* VMware Workstation, Oracle VirtualBox .

La virtualización es la base de IaaS en la nube: cuando solicitamos una VM en AWS/Azure/etc., realmente estamos obteniendo una máquina virtual gestionada por un hypervisor en algún servidor del data center del proveedor. Sin esta tecnología, cada cliente necesitaría un servidor físico dedicado, haciendo inviable la compartición eficiente de recursos que caracteriza al cloud.

8. Contenedores

Junto a la virtualización tradicional, en años recientes han cobrado protagonismo los **contenedores**. Un contenedor es una unidad ligera de software que incluye todo lo necesario para ejecutar una aplicación: código, bibliotecas, dependencias y configuraciones del sistema. A diferencia de una VM, un contenedor **no incluye un sistema operativo completo**, sino que todas las contenedorizaciones comparten el kernel del sistema operativo del host. Por tanto, son mucho más eficientes en el uso de recursos: se pueden ejecutar decenas de contenedores en el espacio donde quizás cabrían solo un par de VMs, ya que cada contenedor agrega muy poca sobrecarga.

Los contenedores garantizan que una aplicación se ejecute de manera **idéntica** sin importar el entorno, ya que encapsulan todas sus dependencias. Si un contenedor funciona en el entorno de desarrollo, funcionará igual en producción, evitando el clásico "pero en mi máquina funciona".

Esta portabilidad los ha hecho muy populares para despliegues en la nube y para metodologías DevOps. **Docker** es la tecnología de contenedores más conocida, que estandarizó el formato de imagen de contenedor. Una **imagen de contenedor** es básicamente una plantilla de la cual se instancian contenedores en tiempo de ejecución . Por ejemplo, puedo tener una imagen con una aplicación web Python, que incluye el runtime de Python y todas las librerías; al ejecutarla, obtengo un contenedor aislado con esa aplicación corriendo, independiente de lo que haya en el sistema host.

En comparación con las máquinas virtuales, los contenedores ofrecen un aislamiento menos completo (comparten kernel, así que no aislan fallos de kernel ni ciertas métricas de recursos), pero suelen ser suficientes para la mayoría de aplicaciones. Son **mucho más ligeros y rápidos** de iniciar o detener que las VMs, por lo que habilitan paradigmas como **microservicios** (aplicaciones divididas en muchos contenedores pequeños) y escalado casi instantáneo de componentes individuales. Por ello, actualmente los contenedores complementan (y a veces sustituyen) a la virtualización tradicional en la nube.

Muchas plataformas PaaS y FaaS internamente usan contenedores para empaquetar las funciones de los clientes, gestionados por sistemas de orquestación como **Kubernetes**. En resumen, los contenedores proporcionan a desarrolladores y operadores una forma **más ligera** de lograr el aislamiento y la portabilidad de aplicaciones que antes solo se conseguía virtualizando todo un sistema operativo .

9. Computación de Alto Rendimiento (HPC) e Inteligencia Artificial en la Nube

La **Computación de Alto Rendimiento (HPC, High Performance Computing)** se refiere al conjunto de tecnologías y prácticas para resolver problemas computacionales

muy complejos utilizando **infraestructuras de gran potencia**. Tradicionalmente, HPC implica supercomputadores o clústeres de cientos o miles de núcleos de CPU, frecuentemente complementados con aceleradores como GPUs, redes de interconexión de baja latencia, almacenamiento de altas prestaciones, etc. . Áreas como la simulación científica, la bioinformática, la climatología, el análisis de big data y el entrenamiento de modelos avanzados de IA requieren capacidades HPC para procesar enormes volúmenes de datos y cálculos en paralelo.

Con la irrupción del cloud computing, ha surgido la posibilidad de **acceder a HPC como servicio**. El paradigma cloud ha permitido que HPC evolucione ofreciendo su potencia bajo demanda a través de la nube. Esto se suele denominar **HPC en la nube** o incluso *HPCaaS (HPC as a Service)*. En lugar de invertir en un supercomputador propio, hoy una organización puede alquilar tiempo de cómputo en los centros de datos de proveedores cloud que ofrecen nodos muy potentes o incluso clústeres completos dedicados.

Por ejemplo, todos los grandes proveedores tienen ofertas en este sentido: **IBM Cloud HPC** , **Google Cloud HPC** , **AWS HPC** , **Oracle Cloud HPC** , **Azure HPC** , entre otros. Estas soluciones proporcionan instancias optimizadas para cómputo intensivo, con CPUs de alta gama, GPUs especializadas y redes rápidas, integradas en los ecosistemas cloud habituales.

10 Características de HPC en la nube

HPC en entornos cloud mantiene las características esenciales de HPC tradicional, pero adaptadas al entorno virtualizado y escalable de la nube. Algunas características destacables:

10.1 Grandes cargas de trabajo (Big Compute)

Nos referimos a *Big Compute* para describir trabajos a gran escala que requieren muchísimos recursos de cálculo simultáneamente, a diferencia de *Big Data* que enfatiza grandes volúmenes de datos. En HPC cloud es posible lanzar trabajos que involucren decenas o cientos de núcleos o aceleradores en paralelo para acortar tiempos de ejecución en problemas enormes (por ejemplo, simular el clima, renderizar animaciones 3D, entrenar un modelo de deep learning con un dataset masivo, etc.).

10.2 Arquitecturas de escalado automático

Las soluciones de HPC cloud suelen aprovechar la **elasticidad** de la nube para ajustar la escala de los recursos según las necesidades. Por ejemplo, se puede configurar un clúster autoscalable que añada nodos de cómputo automáticamente cuando la cola de trabajos crece, y los libere cuando baja la carga. Esto maximiza la eficiencia, ya que se dispone de gran potencia cuando hace falta, sin tenerla encendida en vano cuando no se usa. Las

arquitecturas *cloud-native* para HPC se diseñan para integrarse con estos mecanismos de escalado automático y orquestación dinámica de recursos.

10.3 HPC como servicio especializado

Los proveedores integran HPC en su catálogo de servicios cloud de forma que es accesible a usuarios no expertos en infraestructura. Por ejemplo, Azure ofrece plantillas para desplegar un clúster **Slurm** (un gestor de colas HPC común) sobre máquinas virtuales Azure, de manera relativamente sencilla. O Google Cloud y AWS permiten elegir instancias con GPUs de última generación y conectarlas con un alto ancho de banda. Así, HPC cloud acerca la computación de alto rendimiento a un público más amplio bajo un modelo de autoservicio.

10.4 Costo y enfoque empresarial

Mover HPC a la nube cambia el modelo de costos: de invertir millones en un supercomputador que se deprecia en 5 años, a pagar mes a mes por recursos en la nube. Esto puede ser ventajoso para empresas que solo requieren HPC esporádicamente, pero puede ser más costoso a largo plazo para quienes necesitan computación 24/7. No obstante, permite experimentar y arrancar proyectos HPC sin gran inversión inicial, y escalarlos rápidamente si resultan exitosos.

En resumen, **la HPC en la nube** combina lo mejor de dos mundos: **la potencia masiva de cómputo paralelizable y la flexibilidad**, disponibilidad global y modelo de servicio de la nube. A continuación, profundizaremos en algunos componentes y tendencias relacionados con HPC e IA en entornos cloud.

11. Arquitecturas y recursos *Big Compute*

Para soportar cargas HPC en la nube, los proveedores ofrecen **recursos de procesamiento masivamente paralelos** y arquitecturas especializadas. Algunos elementos típicos son:

11.1 Máquinas virtuales con GPUs y aceleradores

Hoy en día gran parte de las cargas HPC e IA aprovechan aceleración por GPU (p. ej., cómputo CUDA para simulaciones o entrenamiento de redes neuronales). Las nubes ofrecen instancias con GPUs de alto rendimiento. Por ejemplo, instancias con **NVIDIA A100** o **H100** – GPUs muy potentes orientadas a data centers. Estas GPUs suelen venir en nodos con múltiples GPUs interconectadas mediante buses de alta velocidad (NVLink, etc.) y con CPU de altas prestaciones.

11.2 Plataformas de supercomputación dedicadas en la nube

Algunos proveedores cuentan con servidores especialmente diseñados para HPC/IA, como la plataforma **NVIDIA DGX/HGX** que integra varias GPUs, interconexiones rápidas y optimizaciones para cálculos intensivos. Estas plataformas disponibles bajo demanda permiten ejecutar cargas masivas sin poseer la costosa máquina.

11.3 Interconexiones de alta velocidad

En HPC, la comunicación entre nodos es crítica (por ejemplo, en un clúster MPI, los procesos en distintos nodos se envían mensajes constantemente). Por ello, en entornos HPC cloud se emplean redes avanzadas (como InfiniBand) para interconectar las instancias con **baja latencia y alto ancho de banda**. Algunos cloud vendors ofrecen opciones de redes HPC con latencias de microsegundos, comparables a las de un supercomputador tradicional.

11.4 Almacenamiento de altas prestaciones

Complementando el cómputo, se proveen sistemas de ficheros paralelos o almacenamiento SSD de gran IOPS para que la entrada/salida no sea un cuello de botella. Por ejemplo, AWS tiene FSx for Lustre (un servicio gestionado del sistema de ficheros paralelo Lustre, común en HPC). Azure ofrece Azure NetApp Files para almacenamiento compartido de alto rendimiento, etc.

En esencia, los componentes principales en HPC (procesamiento, almacenamiento, red) se reinventan en la nube con ofertas que intentan igualar las prestaciones de un entorno on-premise de HPC. Esto ha difuminado la línea entre *supercomputadores tradicionales* y *clústeres en la nube*. De hecho, en la lista TOP500 de supercomputadoras del mundo ya aparecen sistemas cloud híbridos, y es posible construir un supercomputador virtual alquilando suficientes nodos en la nube.

12. Escalabilidad y auto-escalado

Un beneficio clave del cloud para HPC/IA es la **capacidad de escalado automático** de los recursos (*autoscaling*). En un entorno tradicional HPC, el clúster tiene un tamaño fijo; en la nube, podemos hacer que el "clúster" crezca o disminuya dinámicamente. Esto se logra mediante políticas de escalado que se configuran típicamente en sistemas de orquestación:

12.1 Escalado programado

Añade o quita recursos en momentos predefinidos. Por ejemplo, podemos programar que todos los días a las 9am se instancien 10 nodos adicionales porque se espera que los investigadores lancen sus trabajos matutinos, y apagarlos a las 8pm . Es útil para patrones de carga conocidos (horarios pico, cálculos mensuales, etc.).

12.3 Escalado dinámico (reactivo)

El sistema monitoriza métricas en tiempo real (uso de CPU, longitud de colas de trabajo, etc.) y **asigna recursos dinámicamente** cuando detecta que se necesitan para mantener el rendimiento. Por ejemplo, si la CPU de los nodos existentes está al 100% o hay 50 tareas esperando en cola, se encienden más instancias automáticamente hasta que los indicadores vuelvan a un rango normal; si luego la demanda cae, esos nodos extras se apagan para **minimizar costos**. Este tipo de escalado aprovecha la **elasticidad** de la nube al máximo: recursos casi infinitos pero pagando solo lo necesario en cada momento.

También se distingue entre **escalado vertical** y **horizontal** :

- El **escalado vertical** implica cambiar la capacidad de un recurso dado, por ejemplo, mover una aplicación de una VM pequeña a una VM más grande (más CPU/RAM). Es útil cuando un proceso no puede dividirse fácilmente en varios (no es paralelo), pero tiene el inconveniente de que usualmente requiere reinicios y no es tan automático (además de límites físicos, no puedes crecer indefinidamente una sola máquina).
- El **escalado horizontal** significa añadir más instancias de un recurso, es decir, en vez de uno muy grande, tener muchos pequeños trabajando en paralelo. En la nube, este es el enfoque preferido ya que se puede hacer sin interrumpir el servicio: por ejemplo, si tu aplicación web necesita atender más usuarios, lanzas más copias (más contenedores o VMs) detrás de un balanceador de carga. Luego, cuando baja la demanda, simplemente apagas las copias sobrantes.

En contextos HPC, muchas aplicaciones ya están pensadas para escalado horizontal (jobs MPI, por ejemplo, que usan N procesos en paralelo). Sin embargo, lograr un escalado eficiente no es trivial: es necesario diseñar las aplicaciones teniendo en cuenta que puedan repartirse en múltiples nodos sin introducir cuellos de botella .

Por ejemplo, una **base de datos monolítica** no se puede partir en 10 trozos fácilmente; en HPC, ciertos algoritmos no paralelizables limitarán el beneficio de añadir más nodos (Ley de Amdahl). Por ello, incluso con autoescalado, hay que **refactorizar software y arquitecturas** para que realmente aprovechen la elasticidad cloud.

En la práctica, las **arquitecturas cloud-native** suelen dividir aplicaciones en microservicios o pipelines donde cada paso escala por separado, y usar colas/eventos para desacoplar componentes, facilitando escalar horizontalmente partes individuales.

14. Aprendizaje Federado (*Federated Learning*)

Un concepto moderno relacionado con HPC e IA distribuida es el **aprendizaje federado**. Se trata de una técnica de machine learning en la cual el entrenamiento de un modelo se realiza de forma **colaborativa entre múltiples nodos** (dispositivos, servidores, etc.) sin que estos compartan sus datos brutos, solo compartiendo aprendizajes parciales (gradientes, parámetros del modelo). Es decir, en lugar de centralizar todos los datos en un único supercomputador o nube para entrenar, cada nodo entrena con sus datos locales y solo envía al servidor central actualizaciones de modelo, que luego son agregadas y devueltas a los nodos.

El aprendizaje federado cobra importancia en escenarios donde los datos son sensibles (por ejemplo, información de usuarios en smartphones, historiales médicos en hospitales) y no pueden consolidarse por privacidad, o simplemente cuando los datos están distribuidos geográficamente. Esta técnica requiere **capacidades de cómputo distribuido** considerables: muchos dispositivos entrenando en paralelo y un nodo central (que podría ser una instancia en la nube) agregando resultados. Se considera una forma de HPC/Big Data distribuido, y la nube puede proveer la infraestructura para orquestar esa federación (p.ej., gestionando conexiones, agregando modelos, etc.).

Desde el punto de vista de la arquitectura, el aprendizaje federado implica retos de comunicación de red (enviar y recibir actualizaciones frecuentes), sincronización, y heterogeneidad de recursos (cada nodo puede tener diferente capacidad). Es un ejemplo de cómo **IA y computación distribuida** se combinan: requiere HPC para procesar grandes modelos, y cloud/edge computing para manejar la distribución en muchos nodos.

Estudios recientes revisan numerosas aplicaciones del aprendizaje federado en distintos campos , demostrando que es una tendencia al alza que beneficiará de la infraestructura cloud para escalar.

14 Servicios en la nube para IA y Big Data

Los principales proveedores de nube no solo ofrecen recursos crudos (VMs, almacenamiento) sino también **servicios gestionados de nivel superior** orientados a casos de uso comunes de IA, Big Data y analítica. Estos servicios facilitan a las empresas desplegar soluciones avanzadas sin tener que reinventar componentes complejos. Algunos ejemplos relevantes:

14.1 Lagos de datos (Data Lakes)

Son repositorios centralizados de datos a gran escala que pueden almacenar datos sin estructurar o semiestructurados en su formato nativo. Azure, AWS y Google Cloud ofrecen servicios de Data Lake (por ejemplo, **Azure Data Lake Storage Gen2**). Estas plataformas permiten ingerir datos masivamente y luego procesarlos con diversas herramientas analíticas. Características como separar almacenamiento y cómputo (escalar independientemente), gestión de políticas de ciclo de vida, *tiering* (mover datos fríos a almacenamiento más barato automáticamente), integración con sistemas tipo Hadoop, y controles de seguridad (ACLs, POSIX) las hacen muy útiles para **Big Data** . En un entorno HPC/Big Data en la nube, un Data Lake actúa como fuente común desde la cual múltiples jobs pueden leer/escribir grandes datasets eficientemente.

14.2 Servicios de análisis y Big Data

Las nubes proporcionan entornos listos para frameworks populares de procesamiento de datos. Por ejemplo, **AWS EMR**, **Azure HDInsight** o **Google Dataproc** permiten desplegar clusters Hadoop/Spark gestionados en minutos. También existen servicios serverless para procesamiento de big data, como **Google BigQuery** (consulta SQL sobre petabytes sin gestionar servidores) o **AWS Athena**. Para flujos de datos en tiempo real, hay servicios como **Azure Stream Analytics** o **AWS Kinesis**. Todos estos servicios encapsulan el poder de cómputo paralelo masivo detrás de interfaces más simples (SQL, APIs), democratizando su uso.

14.3 Servicios Cloud de IA (AI as a Service)

Un aspecto cada vez más fuerte es la disponibilidad de **modelos de IA pre-entrenados y APIs de IA** en la nube. Por ejemplo, **Azure AI Services** ofrece una colección de servicios cognitivos (muchos derivados de la tecnología de OpenAI) para tareas como análisis de texto, detección de anomalías, reconocimiento de imágenes, voz, traducción, etc. Google Cloud tiene su *AI Platform* y APIs como Vision AI, Speech-to-Text, etc., y AWS sus AWS AI Services (Rekognition para visión, Comprehend para NLP, etc.).

Estos servicios permiten a un desarrollador integrar capacidades de IA en sus aplicaciones **sin entrenar modelos desde cero**, simplemente consumiendo APIs. Internamente, los proveedores tienen modelos ya entrenados (muchos con técnicas HPC en sus propios laboratorios) y cobran a los usuarios por cada petición de inferencia. Esto resulta muy útil para añadir, por ejemplo, reconocimiento de objetos en imágenes o predicción de series temporales a una app, con mínima inversión de desarrollo.

14.4 Plataformas de Machine Learning en la nube

Para quienes necesitan entrenar sus propios modelos de IA con sus datos (lo cual puede requerir mucha computación), existen servicios como **Azure Machine Learning** , **AWS SageMaker**, **Google AI Platform Pipelines**, etc. Estas plataformas proporcionan un entorno completo para *machine learning* que incluye gestión de datasets, entrenamiento distribuido en clusters de GPU/CPU, auto-ML (entrenamiento automatizado), despliegue de modelos en endpoints escalables, seguimiento de experimentos, etc.

Por ejemplo, Azure Machine Learning permite crear *clusters* de máquinas virtuales de GPU bajo demanda, ejecutar experimentos de entrenamiento en paralelo, y luego desplegar el modelo resultante como un servicio web, todo integrado con su estudio y SDK. Así, el usuario aprovecha HPC en la nube (pudiendo, por ejemplo, usar 20 GPU A100 para entrenar un modelo en un par de horas) sin tener que gestionar manualmente esas máquinas; la plataforma se encarga. Muchas de estas plataformas también ofrecen entornos tipo *notebook* en la nube para desarrollo interactivo (p.ej., Azure ML Studio, Google Colab, etc.).

En definitiva, son un nivel intermedio entre hacerlo todo uno mismo en VMs (IaaS) y consumir APIs listas (AI as a Service) – aquí el usuario entrena sus propios modelos pero con mucha ayuda de herramientas gestionadas.

En nuestra práctica, de hecho, exploraremos algunos de estos servicios: en la parte final sustituiremos el modelo YOLOv8 local por un **servicio de Visión por Computador de Azure**, aprovechando *AI as a Service* en la nube en lugar de computación local (ver Práctica 0 más adelante).

Esto nos permitirá experimentar cómo la nube puede entregar capacidades de IA listas para usar y cómo se comparan con ejecutar un modelo de IA manualmente en infraestructura propia.

15. Computación Cuántica para IA

La computación cuántica promete una nueva era de capacidad de cómputo al aprovechar fenómenos cuánticos (superposición, entrelazamiento) para realizar ciertas operaciones muchísimo más rápido que los ordenadores tradicionales. Si bien aún no es parte del día a día de la nube, los grandes actores tecnológicos ya ofrecen **recursos cuánticos accesibles a través de la nube** para experimentación.

Por ejemplo, **Google Quantum AI** ofrece plataformas en la nube para desarrollar algoritmos cuánticos, e incluso frameworks como TensorFlow Quantum para explorar modelos de IA híbridos cuánticos-clásicos. **IBM Quantum** proporciona acceso vía nube a computadores cuánticos reales y simuladores, incluyendo un ecosistema de software

(Qiskit) para escribir circuitos cuánticos. Iniciativas como **Quantum Spain** buscan crear un ecosistema nacional de computación cuántica en la nube .

En cuanto a IA, la **Quantum AI** estudia cómo algoritmos cuánticos pueden acelerar tareas de inteligencia artificial. Algunas ventajas teóricas de la computación cuántica aplicadas a IA serían: una **capacidad de paralelismo masivo** mucho mayor (debido a la superposición de estados), velocidades superiores en cálculos matriciales o de optimización, y posibilidad de reducir la complejidad algorítmica de ciertos problemas intratables clásicamente.

Por ejemplo, algoritmos de *machine learning* podrían beneficiarse de la aceleración cuántica para analizar bases de datos enormes o encontrar patrones complejos con menos pasos. Sin embargo, actualmente las **limitaciones prácticas** (número de *qubits* reducido, decoherencia, error en operaciones) hacen que la Quantum AI esté en fase experimental.

Aun así, los principales proveedores cloud comienzan a integrar la computación cuántica: **Azure Quantum**, **Amazon Braket**, **Google Quantum Computing Service**, ofrecen entornos unificados donde un desarrollador puede lanzar algoritmos en simuladores o hardware cuántico real a través de la nube. Es de esperar que en el futuro, a medida que la tecnología madure, los **recursos cuánticos** se conviertan en otro elemento más del ecosistema cloud para acelerar cargas HPC e IA particulares. Por ahora, representan un campo altamente especializado pero **prometedor** para potenciar la inteligencia artificial y la computación de alto rendimiento más allá de las capacidades clásicas.

Hasta aquí hemos cubierto los apuntes teóricos de computación en la nube, HPC e IA, estableciendo conceptos que necesitaremos para la parte práctica. A continuación, puedes descargar la **Práctica 0**, que pone en acción muchos de estos temas. En esta práctica, implementaremos un sistema sencillo de *smart city* (iluminación inteligente) utilizando herramientas de visión artificial (YOLOv8) y exploraremos cómo podemos externalizar su procesamiento primero a una máquina virtual en la nube y luego a servicios serverless y de IA en Azure. Esta práctica servirá para consolidar conocimientos, dándonos una experiencia directa en el uso de la nube para computación intensiva en IA.