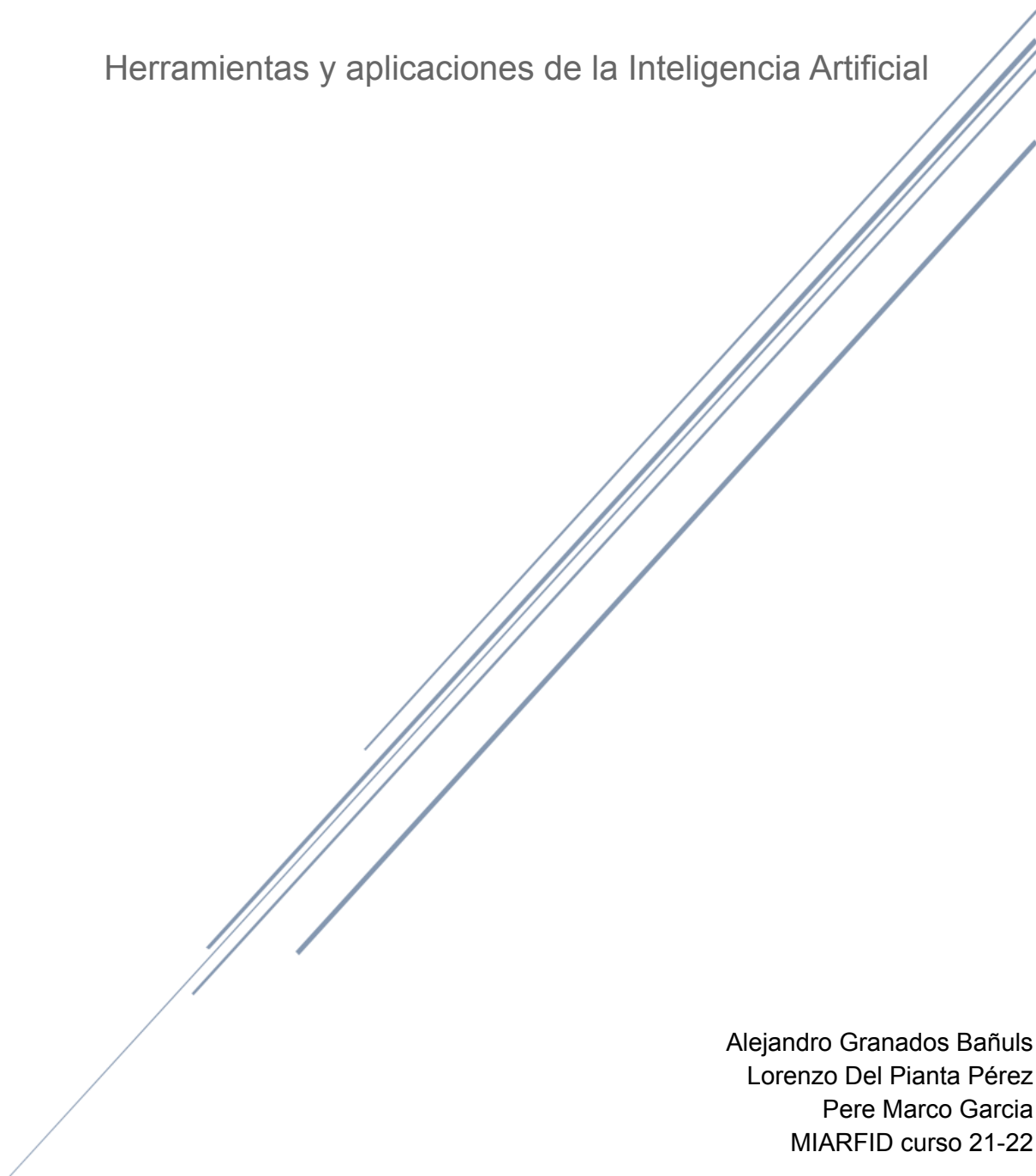


Implementación y aplicación de redes neuronales para reconocimiento de imágenes mediante herramientas en la nube

Herramientas y aplicaciones de la Inteligencia Artificial



Alejandro Granados Bañuls
Lorenzo Del Pianta Pérez
Pere Marco Garcia
MIARFID curso 21-22

Índice

1. Introducción: Plataformas Cloud con Deep Learning, con enfoque en pytorch	3
2. Casos de éxito	4
3. Presentación problema	5
4. Solución, Plataforma y tecnologías	6
4.1. Plataforma: Google Cloud	6
4.2. Tecnología: Vertex AI	7
4.3. Tecnología: Artifact Registry	7
5. Implementación	8
6. Ejecución de la solución	8
7. Conclusión y Trabajo Futuro	8
8. Bibliografía	10

1. Introducción: Plataformas Cloud con Deep Learning, con enfoque en pytorch

La computación en la nube es un modelo que permite el acceso ubicuo, cómodo y bajo demanda a un conjunto compartido de recursos informáticos configurables (redes, servidores, almacenamiento, aplicaciones, servicios...) que pueden ser rápidamente aprovisionados y liberados con un mínimo esfuerzo de gestión. Hoy en día, además de los servicios clásicos, cada vez se emplean más estas plataformas para proveer servicios de Machine Learning debido a que las características de dichas plataformas se adaptan perfectamente a estos servicios, y son:

- Bajo demanda: a petición del usuario
- A través de Internet: accesibles en todo momento desde cualquier lugar
- Recursos: Proveen los recursos computacionales para poder ejecutar nuestros modelos
- Elasticidad: flexibilidad a la hora de cambiar los recursos.

Muchas de las grandes empresas tecnológicas del sector ofrecen estos servicios y, a pesar de sus diferencias, todas ofrecen una serie de características comunes, como la posibilidad de entrenar modelos sin necesidad de tener una gran experiencia en el sector, la integración con otros proveedores y herramientas en la nube, el uso de recursos de las plataformas en base a las necesidades del usuario.

Hoy en día, las principales plataformas Cloud con deep Learning son las que se escriben a continuación:

- Azure platform (Microsoft)
- GoogleCloud (Google)
- Watson (IBM)
- AWS (Amazon)

En la gran mayoría de estas plataformas, aunque podamos elegir entre varias propuestas, encontramos como framework a Pytorch. Microsoft es uno de los principales impulsores de esta tecnología ya que sus desarrolladores la utilizan ampliamente en el desarrollo de modelos con el fin de ofrecer nuevas experiencias dentro de sus productos, y por lo tanto Azure platform se basa en PyTorch. [1]

¿Qué es PyTorch y qué ventajas tiene frente a sus competidores como TensorFlow?

Originalmente desarrollado por Facebook, PyTorch es un framework de Machine Learning de código abierto creado para ser flexible y modular para la investigación, con la estabilidad y el soporte necesarios para el despliegue de producción. Creado sobre Python, PyTorch proporciona paquetes como NumPy para funciones de alto nivel sobre tensores con una fuerte aceleración de GPU. Con la última versión de PyTorch, el framework proporciona ejecución basada en gráficos, capacitación distribuida, implementación móvil y cuantización.

Junto a TensorFlow, provee abstracciones para reutilizar el código de manera eficiente y agilizar el desarrollo. No obstante, respecto a Tensorflow, la sencillez de su interfaz, y su orientación a objetos, lo convierten en la opción más asequible para crear redes neuronales artificiales. A través del presente trabajo, se pretende implementar una aplicación de reconocimiento de imágenes basada en la plataforma Google Cloud y su herramienta Vertex AI.

2. Casos de éxito

El reconocimiento de objetos o personas en imágenes ha supuesto un gran avance en muchos ámbitos de nuestro entorno. Aunque generalmente no se es consciente de ello, supone un papel indispensable en sectores como la industria, la medicina, la seguridad o incluso la zoología y la meteorología. Los sistemas de reconocimiento además no han dejado de evolucionar en las últimas décadas, siendo cada vez más precisos y rápidos. A día de hoy muchos sistemas con los mejores resultados están basados en redes neuronales o deep Learning. Sistemas con un gran número de capas que permiten aprender, dentro de sus limitaciones, la tarea para la que son entrenados.

Sin embargo, este gran número de parámetros y cálculos no son ejecutables por muchos equipos, y mucho menos por pequeños dispositivos que se puedan encontrar en cualquier entorno. Es en este punto donde la computación en la nube muestra toda su utilidad. Con la aplicación de sistemas de IA en sistemas distribuidos se puede utilizar sistemas con prestaciones muy altas y con un elevado coste computacional y de memoria de forma cómoda y remota, sin preocuparse en exceso de las capacidades computacionales del equipo en el que se aplica el sistema.

Un ejemplo de estos sistemas lo encontramos en el reconocimiento de comida a través de imágenes. Esta tarea puede ser de gran utilidad en e-health, especialmente en la elaboración de dietas para gente mayor o con problemas médicos. También es de interés esta tecnología para conocer el nombre o los ingredientes de un plato que no conoces. Por este motivo en los últimos años se ha incorporado la utilización de este tipo de reconocimiento en dispositivos móviles. Debido a la gran variabilidad que supone la presentación de la comida, el número de datos que es necesario para entrenar un sistema con buenos resultados es muy elevado. Por este motivo, algunas aproximaciones al problema contemplan el uso de cluster de procesamiento para tratar tanto el entrenamiento como parte del reconocimiento en fases de uso mediante herramientas de procesamiento distribuido como es Hadoop [2].

Otra aplicación de este tipo de sistemas es el procesamiento de videos mediante sistemas distribuidos. Hoy en día, gracias en parte al IoT y a los dispositivos móviles, miles de dispositivos tienen la tarea de detectar e identificar caras en enormes cantidades de imágenes. Debido al gran flujo de información algunos trabajos han separado la distribución de la tarea en dos pasos, en primer lugar, la detección de las caras se realiza en una unidad de procesamiento cercana a las cámaras, mientras que la identificación de la persona mediante un vector de características se transmite a un sistema de procesamiento en la nube, eliminando la carga del dispositivo donde se captan las imágenes y limitando la carga de la transmisión de información [3].

Otros trabajos también buscan el reconocimiento de caras en imágenes mediante sistemas distribuidos en la nube. En el caso de Zhang [4] se centran en el uso de Hadoop como plataforma sistema para el procesamiento en la nube. Para la clasificación de las imágenes utilizan un SVM, algoritmo que concluyen que se ve beneficiado por las distribuciones en la nube. Este tipo de trabajos nos invita a probar la implementación de diferentes arquitecturas en la nube y utilizarlas en este tipo de entorno con la finalidad tanto de explorar este tipo de plataformas como de estudiar el comportamiento de los modelos en estas circunstancias.

3. Presentación problema

En nuestro trabajo se pretende implementar un sistema de reconocimiento de objetos a partir de modelos de Deep Learning haciendo uso de plataformas de procesamiento distribuido y aprovechando las ventajas que este tipo de entorno ofrece.

Con esta finalidad se ha tomado como punto de partida se ha tomado el dataset PASCAL VOC 2012 [5]. Este corpus está constituido por aproximadamente 11500 imágenes etiquetadas con los objetos que contienen. Como se propone oficialmente se divide en 50% train y 50% validación y los objetos están repartidos en 20 clases distintas donde predominan las relacionadas con vehículos o animales. Para la clasificación de este dataset se ha decidido recurrir a YOLO [6]. YOLO es una arquitectura que, junto con sus versiones posteriores [7][8], ha supuesto el estado del arte del reconocimiento de objetos en imágenes de los últimos años. Sin embargo, debido a su gran cantidad de parámetros, su entrenamiento y uso en el día a día resulta inviable para la mayoría de los dispositivos.

Por este motivo, como en el último caso de éxito mencionado anteriormente, nuestro objetivo consiste en poder entrenar un modelo en un sistema distribuido con dichas imágenes. Una vez entrenado se quiere utilizar sistemas en la nube para el reconocimiento de objetos en las imágenes.

4. Solución, Plataforma y tecnologías

Como se ha mencionado, el tratamiento de una red como YOLO a nivel local, y mucho menos en dispositivos móviles, se hace muchas veces inviable. Por este motivo es necesario la utilización de sistemas de procesamiento en la nube. Debido a lo visto en la asignatura cursada sobre AWS hemos decidido explorar otras tecnologías para familiarizarnos con ellas. Es por este motivo que hemos decidido utilizar Google Cloud y sus herramientas como Vertex AI y Artifact Registry, las cuales describimos a continuación.

4.1. Plataforma: Google Cloud

Google Cloud como su nombre indica es la plataforma que recoge todas las propuestas de google en este ámbito, y ofrece desde servicios de infraestructura que soportan la ejecución de aplicaciones en contenedores, el almacenaje de datos, incluyendo una potente red de distribución, hasta servicios de analíticas streaming de datos, pasando por herramientas de AI como las que se describen a continuación.[9]

En concreto Google Cloud, ofrece tres principales herramientas de Inteligencia Artificial, entre las cuales se encuentra la escogida para el proyecto, Vertex AI. A continuación se citan dichas herramientas:

- Vertex AI: Una plataforma integral y totalmente gestionada para la ciencia de datos y el aprendizaje automático.
- Contact center AI: Herramienta que permite mejorar la eficiencia operativa y la asistencia personalizada que se ofrece a los clientes desde que se ponen en contacto con la empresa
- Document AI: Solución que permite aprovechar los datos sin estructurar para mejorar la experiencia del cliente y tomar decisiones fundamentadas dentro de la empresa.

Para utilizar la plataforma, se puede crear una cuenta para evaluar el rendimiento de los productos de Google Cloud en situaciones reales recibiendo 300 USD en crédito gratis. Además, todos los clientes pueden usar más de 20 productos de forma gratuita hasta alcanzar los límites de uso mensuales y también existen ventajas para los estudiantes.

4.2. Tecnología: Vertex AI

Google Cloud ha presentado recientemente Vertex AI, una plataforma de aprendizaje automático administrada y unificada, diseñada para acelerar la implementación y el mantenimiento de modelos de IA, en efecto, la herramienta proporciona APIs previamente entrenadas para ámbitos de visión, vídeo, lenguaje natural y para otros usos.[10]

Otra característica relevante de vertex AI es su compatibilidad con todos los frameworks de código abierto como TensorFlow, PyTorch y scikit-learn, además de admitir todos los frameworks de aprendizaje automático y las ramas de la inteligencia artificial, mediante contenedores personalizados para el entrenamiento y la predicción.

Adicionalmente, ha influido en su elección su gran facilidad de uso ya que, gracias al uso de AutoML se pueden entrenar modelos de alta calidad sin tener una amplia experiencia en dicho ámbito.[10]

Vertex AI es una herramienta de pago que cobra por el entrenamiento de modelos, las predicciones y el uso de recursos de productos de Google Cloud, no obstante, están disponibles cuentas para estudiantes con un saldo inicial suficiente para probar la herramienta.

En conclusión, las características a destacar de Vertex AI son las siguientes:

- Capacidad de crear y escalar modelos de aprendizaje automático personalizados en una plataforma de inteligencia artificial unificada.
- Realizar tareas de desarrollo con herramientas de aprendizaje automático empleadas por el mismo Google.

- Desplegar más modelos en menos tiempo y reducir las líneas de código necesarias para crear modelos personalizados. (reducción del 80% según el productor)
- Emplear herramientas de operaciones de aprendizaje automático para gestionar los datos y modelos de manera fácil y segura. (AutoML)

4.3. Tecnología: Artifact Registry

Artifact Registry es un administrador unificado que gestiona imágenes de contenedores (docker u otros) y paquetes de lenguajes como por ejemplo Maven y npm, que pertenece al entorno de google Cloud, por lo que está totalmente integrado con las otras herramientas del ecosistema. Adicionalmente ofrece compatibilidad con protocolos de artefactos nativos, obteniendo la ventaja de que la integración con herramientas de despliegue continuo (CI/CD), con el fin de configurar flujos de procesamiento automático, es más ágil.[11]

Dicho administrador se utiliza en el proyecto como herramienta donde subir y almacenar la imagen del contenedor que se ha creado anteriormente. Una vez esta se encuentra almacenada en Artifact Registry, esta es accesible desde Vertex AI y por lo tanto se pueden realizar los entrenamientos del modelo.

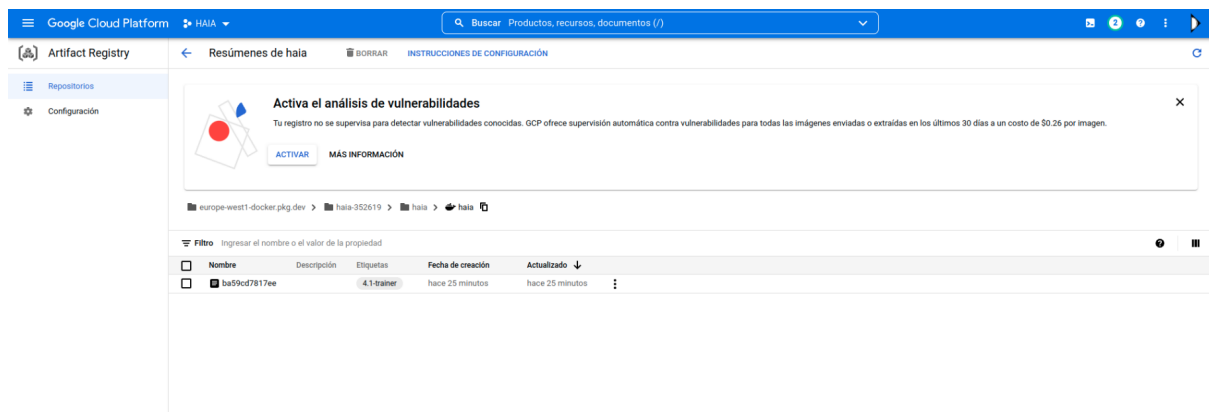


Figura 1 - Panel de control de artifact registry, imagen del proyecto guardada en la nube.

5. Implementación

A continuación, se describe la implementación de la solución y las plataformas y tecnologías empleadas. La red neuronal se ha creado desde 0 usando pytorch. No se va a entrar en detalle en la implementación de la arquitectura ni la experimentación, sólo mencionar que la red creada es la llamada YoloV3.

Para poder realizar un entrenamiento en la nube se debía crear una Imagen y un contenedor de Docker que permitiera ejecutar un script de entrenamiento. Para ello se desarrolló un Dockerfile en el que se instalaban todas librerías y paquetes necesarios. El Dockerfile puede verse en la Figura 2.

```

FROM pytorch/pytorch:1.11.0-cuda11.3-cudnn8-devel

RUN mkdir /opt/DDVIE
RUN mkdir /opt/DDVIE/ddvie

ADD ddvie /opt/DDVIE/ddvie
ADD __main__.py /opt/DDVIE
ADD requirements.txt /opt/DDVIE
ADD .env /opt/DDVIE

WORKDIR /opt/DDVIE

RUN rm /etc/apt/sources.list.d/cuda.list
RUN rm /etc/apt/sources.list.d/nvidia-ml.list
RUN apt-get update
RUN apt-get install software-properties-common -y
RUN add-apt-repository ppa:deadsnakes/ppa
RUN DEBIAN_FRONTEND=noninteractive TZ=Europe/London apt-get -y install tzdata
RUN apt-get install -y nvidia-cuda-toolkit
RUN apt-get install python3.9 -y
RUN apt-get install python3-pip -y
RUN apt-get install python3.9-distutils -y
RUN python3.9 -m pip install --upgrade setuptools
RUN python3.9 -m pip install --upgrade pip
RUN python3.9 -m pip install --upgrade distlib
RUN python3.9 -m pip install torch==1.11.0+cu113 torchvision --extra-index-url https://download.pytorch.org/whl/cu113
RUN python3.9 -m pip install -r /opt/DDVIE/requirements.txt

ENTRYPOINT [ "/usr/bin/python3.9", "__main__.py" ]

```

Figura 2: Dockerfile

Una vez creado el Dockerfile se procedería a subir el contenedor al *Artificat Registry* de google, que es un repositorio de contenedores. Esto se realizaría de forma sencilla, pues primero habría que *buildear* el contenedor con el comando *docker build* y a continuación subir con *docker push* el contenedor a *Artifact Registry*.

Una vez subido el código en un contenedor a Google Cloud se debería añadir los datos para que el sistema pudiera entrenar. Para ello se usaría la funcionalidad de Google Cloud de almacenamiento, denominada *Cloud Storage*. En este caso habían dos formas de subir los datos, una por terminal y otra por el navegador. Se optó por subir los datos mediante el terminal porque eran muchos archivos (40.000) y el navegador fallaba. La interfaz con los datos se puede ver en la Figura 3.

La forma de obtener datos desde el contenedor es cómo si se abrieran fichero de forma normal, pero con el prefijo *gcs://*. Esto es así porque *Google Cloud* posee un sistema llamado *FUSE* que permite acceder a un sistema de archivos en *Cloud Storage* de forma convencional si tanto el contenedor como los datos de *Cloud Storage* están alojados en la misma región (europe-west1 Bélgica en nuestro caso).

Con los datos y el contenedor preparados, se procedería a crear un trabajo de entrenamiento que permitiera ejecutar el *training* de la red neuronal YoloV3. Para ello se lanzaría el el comando especificado en el *ENTRYPOINT* del *Dockerfile*. Para ello se debe acceder a *Vertex AI*, al apartado de *Capacitación y Custom Jobs*. Ahí se podría listar todos los entrenamientos realizados y los que están en curso. La interfaz se puede ver en la Figura 4.

Google Cloud Platform

HAIA

Cloud Storage

Navegador

Supervisión

Configuración

←

Detalles del bucket

vpc-data

Ubicación

Clase de almacenamiento

Acceso público

Protección

europa-west1 (Bélgica)

Standard

No público

Ninguna

OBJETOS

CONFIGURACIÓN

PERMISOS

PROTECCIÓN

Depósitos > vpc-data > data

SUBIR ARCHIVOS

SUBIR CARPETA

CREAR CARPETA

ADMINISTRAR

Filtrar solo por prefijo de nombre

Filtro

Filtrar objetos y carpetas

<input type="checkbox"/>	Nombre	Tamaño	Tipo
<input type="checkbox"/>	VOCdevkit/	—	Carpeta
<input type="checkbox"/>	VOCtrainval_11-May-2012.tar	1.9 GB	application/x-tar

Figura 3: Interfaz Cloud Storage

Google Cloud Platform

HAIA

Vertex AI

Panel

Conjuntos de datos

Características

Tareas de etiquetado

Workbench

Canalizaciones

Capacitación

Experimentos

Modelos

Extremos

Predicciones por lotes

Metadatos

Motor de coincidencias

Entrenamiento

+ CREAR

TRAINING PIPELINES

CUSTOM JOBS

HYPERPARAMETER TUNING JOBS

Los trabajos personalizados especifican cómo Vertex AI ejecuta tu código de entrenamiento personalizado, incluidos los grupos de trabajadores, los tipos de máquinas y la configuración relacionada con la aplicación de entrenamiento de Python y el contenedor personalizado. Solo los modelos con entrenamiento personalizado usan los trabajos personalizados, no así los modelos entrenados con AutoML. [Más información](#)

Región

europa-west1 (Bélgica)

Filtro

Ingresa un nombre de propiedad

Nombre	ID	Estado	Tipo de trabajo
HAIA-custom-job	2524313770629529600	Detenido	Trabajo personalizado
HAIA-custom-job	9056257264549101568	Detenido	Trabajo personalizado
HAIA-custom-job	6124413907130908672	Detenido	Trabajo personalizado
HAIA-custom-job	4444571246121713664	Detenido	Trabajo personalizado
HAIA-custom-job	3356529723646672896	Con errores	Trabajo personalizado

Figura 4: Listado de entrenamientos de Vertex AI

9

Para crear un trabajo sólo hay que especificar el nombre, el contenedor y la configuración de las máquinas que se usarán para el entrenamiento. A parte hay varias opciones más de configuración (*Tuning* de hiperparámetros o lanzar un *Endpoint* para predicciones) pero salía del alcance de este trabajo. En las Figuras 5 y 6 se muestran la elección del contenedor y la elección de la configuración de las máquinas.

Figura 5: Elección contenedor para el entrenamiento

Cómo se puede ver en la Figura 6, Google Cloud brinda varias opciones para configurar las máquinas que gestionarán el entrenamiento. Se pueden elegir varios *workers* cada uno con una o más tarjetas gráficas que proporciona Google. A parte hay muchos tipos de máquinas CPU que pueden ayudar al entrenamiento. Por ejemplo, en muchos casos los entrenamientos tienen cuellos de botella en mandar y recibir las imágenes a GPU, por lo que hay máquinas con configuración *HighMem* que tienen en cuenta este problema.

El precio de una ejecución de entrenamiento dependerá de la configuración de las máquinas elegidas así como del tiempo que tarde esa ejecución. En el presente proyecto se ha decidido no usar aceleradores ni más de un *Worker*, así cómo usar una máquina sencilla para que el coste de la ejecución no fuera muy alto.

Train new model

- ✓ Método de entrenamiento
- ✓ Detalles del modelo
- ✓ Contenedor de entrenamiento
- ✓ Hiperparámetros (opcional)
- 5 Procesamiento y precios**
- 6 Contenedor de predicción (opcional)

COMENZAR EL ENTRENAMIENTO

CANCELAR

Los precios del entrenamiento de modelos se basan en la cantidad de tiempo empleado para el entrenamiento, los tipos de máquinas y los aceleradores que se usaron. [Más información](#)

Región
europe-west1 (Bélgica)

Compute settings

Select the type of virtual machine to use for your worker pool. You can add up to 4 worker pools. To learn about compute costs and how to map your ML framework's roles to specific worker pools, consult the [documentación](#)

Worker pool 0

Machine type *
n1-standard-8, 8 vCPUs, 30 GiB memory

Accelerator type

NVIDIA_TESLA_K80
NVIDIA_TESLA_P100
NVIDIA_TESLA_T4

Disk type
SSD

Disk size (GB)
100

ADD MORE WORKER POOLS (OPTIONAL)

CONTINUAR

Figura 6: Elección de la configuración de las máquinas

6. Ejecución de la solución

En este punto se va a mostrar cómo es el sistema de *logs* de *Google Cloud* así como visualización del estado de un entrenamiento. Además se procederá a mostrar cómo es el uso de la CPU y la memoria de la configuración usada en el entrenamiento. La ejecución que se va a mostrar ha estado 8h ejecutándose.

En la Figura 7 se pueden ver cómo son los *logs* que proporciona Google cuando se está ejecutando un entrenamiento. Los *prints* que se hagan adecuadamente (usando la librería de *logs* de Google preferiblemente) saldrán en esta herramienta. De esta forma se puede ver el estado de la ejecución de forma sencilla.

Para observar el estado del uso de la CPU, la GPU y/o la memoria de las máquinas usadas se debería entrar al detalle del entrenamiento en curso. Dentro del detalle, en la parte de abajo aparecerían gráficas con el uso de cada uno de los componentes. En las Figuras 8 y 9 puede verse un ejemplo de esto.

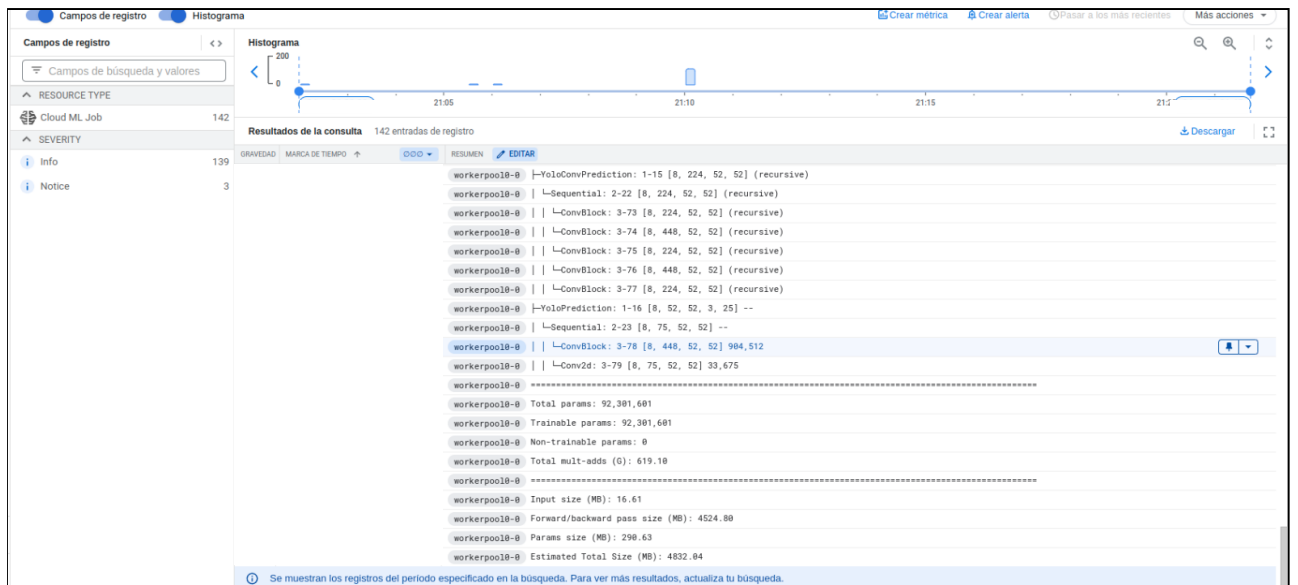


Figura 7 - Logs del entrenamiento en Vertex AI.



Figura 8: Uso de la CPU en el entrenamiento

Gracias a las gráficas se puede saber que el flujo de datos enviado en el paso de validación (es decir, enviar datos a la red sin hacer *backpropagation*) es el que más consume a la hora de enviar datos por la red, pero que no afecta en el uso de CPU.

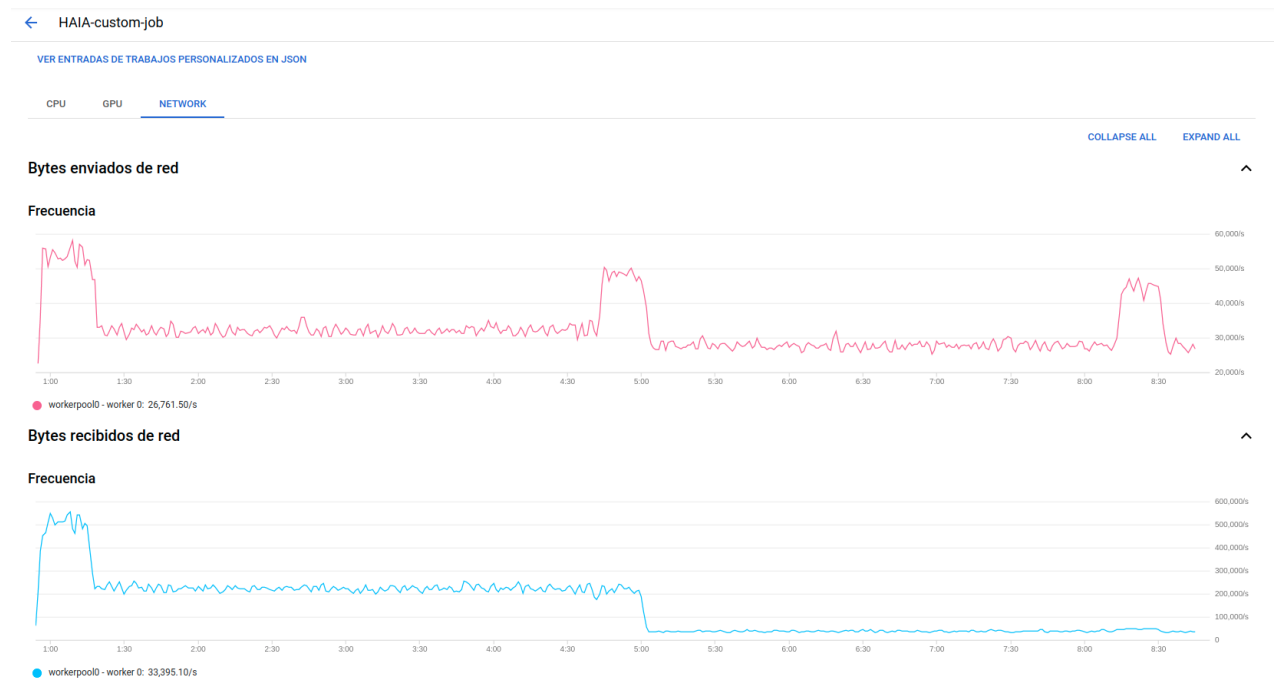


Figura 9: Envío de datos en la red de las máquinas.

7. Conclusión

Google Cloud es una muy buena plataforma para gestionar tareas relacionadas con la Inteligencia Artificial. Permite abstraerse de tener que realizar tareas de gestión de máquinas o siquiera el tener que comprar gráficas para el entrenamiento. Con un acercamiento de pago por uso, ahorra la necesidad de tener contratada una máquina 24/7 o tener que comprarla físicamente.

La interfaz es intuitiva y contiene un montón de documentación. Es necesario decir que de tanta documentación que tiene Google Cloud es fácil perderse y no encontrar lo que se busca. La forma de autenticarse para poder trabajar en una máquina local es muy sencilla, con tener un navegador y un terminal es suficiente.

A parte Google tiene sistemas de inteligencia artificial preentrenados o preparados para entrenar (*AutoML*). Si no se necesita desarrollar un sistema nuevo y se poseen los datos siempre se puede usar este camino.

El sistema *FUSE* también es muy buena ventaja a la hora de desarrollar una solución, pues si se tiene la constancia de que tanto el almacenamiento de los datos así como el contenedor van a estar almacenados en la misma región, la gestión de ficheros se puede tratar como si se trabajara en una máquina local.

8. Bibliografía

- [1] Microsoft, PyTorch on Azure – Deep Learning With PyTorch, 2022, <https://azure.microsoft.com/en-us/develop/pytorch/#what-is-pytorch>
- [2] Duan, P., Wang, W., Zhang, W., Gong, F., Zhang, P., & Rao, Y. (2013, August). Food image recognition using pervasive cloud computing. In 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing (pp. 1631-1637). IEEE.
- [3] Amin, A. H. M., Ahmad, N. M., & Ali, A. M. M. (2016, May). Decentralized face recognition scheme for distributed video surveillance in IoT-cloud infrastructure. In 2016 IEEE region 10 symposium (TENSYP) (pp. 119-124). IEEE.
- [4] Zhang, B. (2019). Distributed SVM face recognition based on Hadoop. *Cluster Computing*, 22(1), 827-834.
- [5] Everingham, M (2012), Visual Object Classes Challenge 2012 (VOC2012), <http://host.robots.ox.ac.uk/pascal/VOC/voc2012/>
- [6] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the *IEEE conference on computer vision and pattern recognition* (pp. 779-788).
- [7] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7263-7271).
- [8] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.
- [9] Google Cloud, Servicios de cloud computing - IA y aprendizaje automatico, 2022, <https://cloud.google.com/?hl=es>.
- [10] Google Cloud, Vertex AI, 2022, <https://cloud.google.com/vertex-ai?hl=es>
- [11] Google Cloud, Artifact Registry, 2022, <https://cloud.google.com/artifact-registry?hl=es>