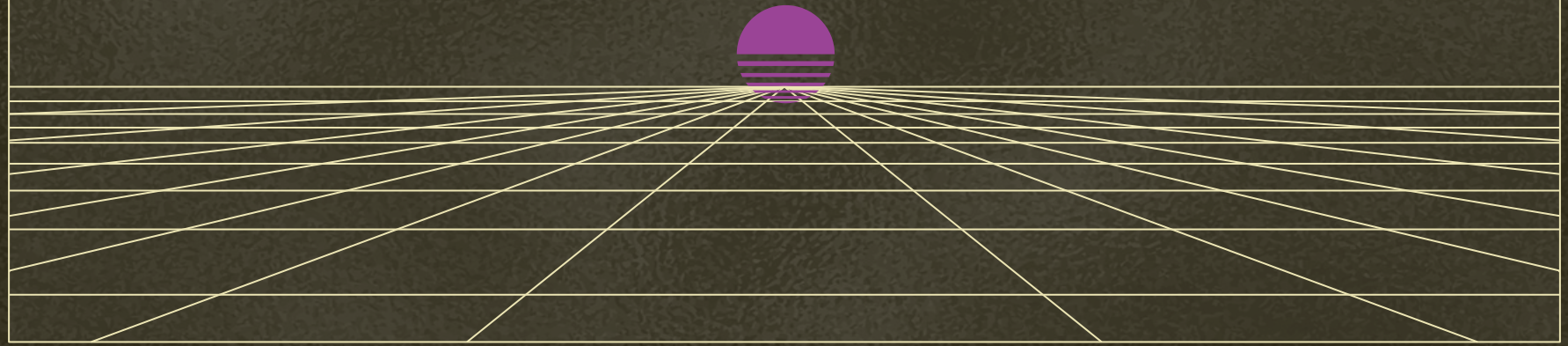
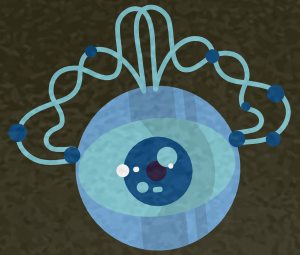
Coral AI
Edge TPU
Teoría básica de la implementación



Presentación creada por: Arturo Sirvent Fresneda (2022)

Contents

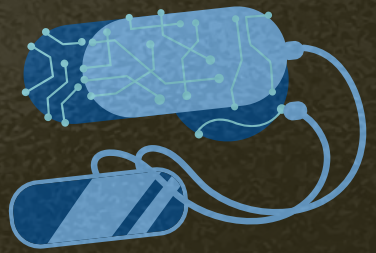
- ¿ Por qué y cuándo hacer uso de TPUs ?
- ¿ Què es la Coral AI ?
- ¿ Cómo se implementa un modelo en Coral AI ?



1º-

TPUs vs GPUs

¿ Por qué y cuándo hacer uso de TPUs ?



Diferencias básicas entre TPU y GPU

TPU (Tensor Processing Unit)

- Optimizado para realizar multiplicaciones de matrices al 100% de eficiencia.
- Todo el proceso es ejecutado en serie sin la necesidad de guardar información en caché.
- Desarrollado por Google para una tarea muy concreta y con un framework muy específico (TensorFlow).

GPU (Graphical Processing Unit)

- Al contrario que la CPUs que son de propósito general, estas están optimizadas para la renderización y procesado de gráficos 2D y 3D.
- Una GPU puede manejar miles de operaciones por ciclo (una GPU puede cientos).
- Aunque es más óptima que la CPU, no está pensada para solo realizar la multiplicación matricial sin más, aun cuenta con cierta versatilidad, al contrario que la TPU.

¿Cuándo usar una TPU ?

Cálculos dominados por álgebra matricial.

La TPU optimiza este tipo de operaciones.

Modelos muy grandes con Batch-sizes muy grandes.

El uso de memoria de caché es más óptimo.

Entrenamientos muy largos (semanas o meses).

Mejor uso de los recursos y estabilidad.

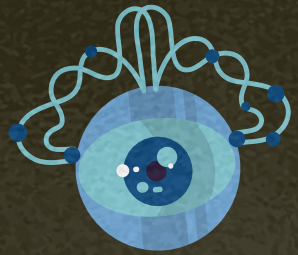
No usar si se requiere aritmética de alta precisión. (e.g. float 32)

Parte de la optimización pasa por cuantizar las operaciones a una precisión más baja.

No usar para trabajos que requieran constantes accesos a memoria.

Eso impide que se ejecute el modelo en un paso continuo.

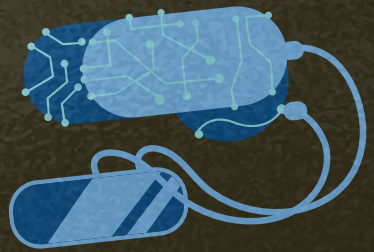
En general los modelos que se pueden correr en una TPU tiene unas características muy específicas.
(lo cual no quita que siga habiendo mucha versatilidad en la construcción de los modelos a ejecutar)



2º-

Coral AI

¿Qué es una Edge TPU?



¿ Qué es Edge TPU ?

Con la popularización del llamado **Internet de las Cosas** (IoT), y debido a que estos dispositivos a veces deben funcionar con energía limitada (incluida la batería), Google diseñó el coprocesador Edge TPU para acelerar la inferencia en dispositivos de bajo consumo.

Permite la ejecución de modelos AI en el perímetro, es decir, local.

Una Edge TPU individual puede realizar 4 billones de operaciones por segundo (4 TOPS) con solo 2 watios de potencia.

Por ejemplo, la Edge TPU puede ejecutar modelos de visión móvil de vanguardia como MobileNet V2, a casi 400 fotogramas* por segundo y de manera eficiente.

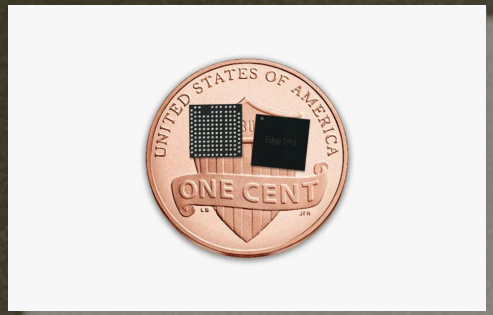
Coral AI

Coral AI es un **acelerador USB** que nos da acceso a un coprocesador **Edge TPU** para la ejecución de modelos de *machine learning* a altas velocidades de cálculo.

Para usar este acelerador solo hace falta:

- Conectarlo al USB.
- Instalar las librerías (drivers) que conectan hardware y software.
- Usar un modelo compatible con las TPUs, e indicarle en código que está disponible para su uso una Edge TPU.

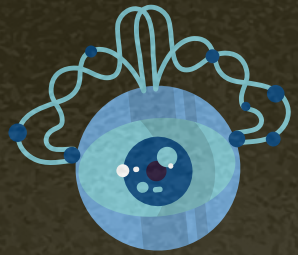
¿ Cómo es el Edge TPU ?



Edge TPU

Productos: Placa de desarrollo, SoM, chip sencillo, **Acelerador USB (Coral AI).**

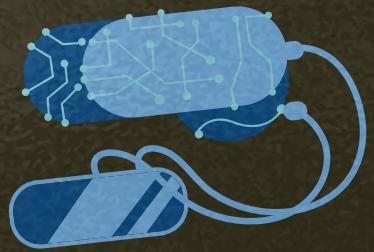
Los precios van desde casi 20$ hasta los 130$.



3º-

Procedimiento para implementar un modelo con Coral AI.

¿Cómo se ejecutan los modelos en una TPU?



¿ Qué pasos seguir al implementar el modelo ?

Instalar el *Edge TPU runtime*

Crear modelo cumpliendo con las limitaciones.

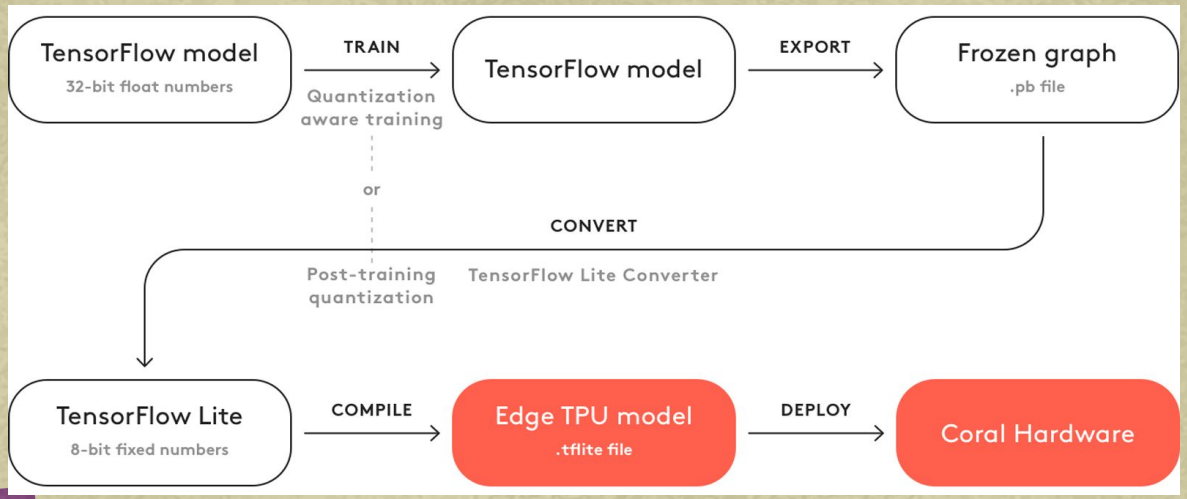
Exportar el modelo como TF-lite.

Estas limitaciones son las requeridas más tarde por el compilador, para que todo vaya bien.

Compilar el modelo para su ejecución en la TPU (**TPU-compiler**)

Usar el módulo ***tflite_runtime*** para implementar el modelo en el código.

O usar la api de Google **PyCoral** la cual simplifica un poco la implementación.



Esquema de: https://coral.ai/docs/edgetpu/models-intro/#compatibility-overview

Algunas limitaciones de los modelos

Tensor parameters are quantized.

(8-bit fixed-point numbers; int8 or uint8)

Tensor sizes are constant at compile-time

(no dynamic sizes)

Model parameters are constant at compile-time

(such as bias tensors)

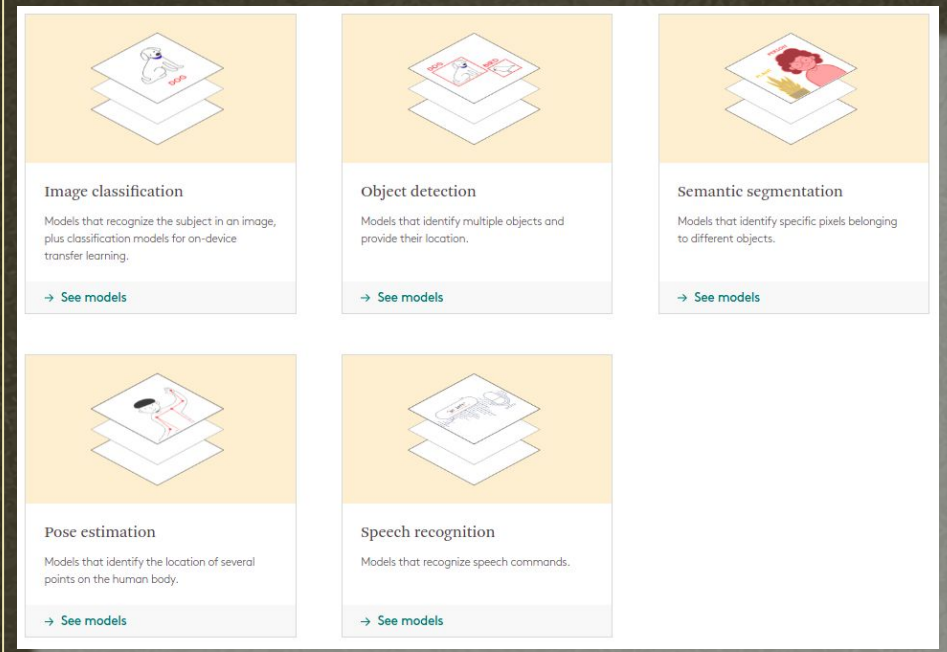
Tensors are either 1-, 2-, or 3-dimensional.

If a tensor has more than 3 dimensions, then only the 3 innermost dimensions may have a size greater than 1.

The model uses only the operations supported by the Edge TPU

Hay una lista en la página web:
https://coral.ai/docs/edgetpu/models-intro/#quantization

Modelos aptos y ya entrenados.



El transfer learning debería ser nuestra primera opción si tenemos recursos limitados y nuestro problema se parece a alguno ya resuelto.

Disponibles en: https://coral.ai/models

También hay tutoriales para entrenar los propios modelos en Colab, Docker y on-board (para la dev-board).
https://coral.ai/docs/edgetpu/models-intro/

Gracias

por su atención

¿Preguntas?

Referencias

- https://cloud.google.com/tpu/docs/tpus?hl=es-419
- https://serverguy.com/comparison/cpu-vs-gpu-vs-tpu/
- https://coral.ai

- https://coral.ai/docs/edgetpu/tflite-python/#overview
- https://coral.ai/docs/edgetpu/inference/
- https://coral.ai/models
- https://coral.ai/docs/edgetpu/models-intro/

Trabajo **creado por Arturo Sirvent Fresneda**
como parte de la evaluación de la asignatura de
Aprendizaje Profundo del **Máster de Ciencia de Datos (UV)**
Abril de *2022*