

Parte 1

Tensorflow 2

Profesores:

Javier López

Daniel Cano



The background features a vibrant, abstract design composed of overlapping, wavy bands of color transitioning from deep red at the bottom to bright yellow at the top. The TensorFlow logo is centered in the upper half of the image.

TensorFlow
2.2.0

Antes de nada

¿Qué es Tensorflow?

- Librería para **computación numérica eficiente**
- Permite expresar operaciones como grafos
- Pensado para Inteligencia Artificial

¿Dónde y por qué se utiliza?

- Todos los productos e investigación de Google
- Base para otros proyectos y librerías

Antes de nada

¿Qué es Keras?

- Capa abstracta sobre librerías de cómputo
- Especializada en Deep Learning
- Actualizaciones a nivel estado del arte

¿Dónde y por qué se utiliza?

- Simplifica el proceso de elaborar el modelo
- Se utiliza para investigación y experimentación

Abstracciones

Redes Neuronales

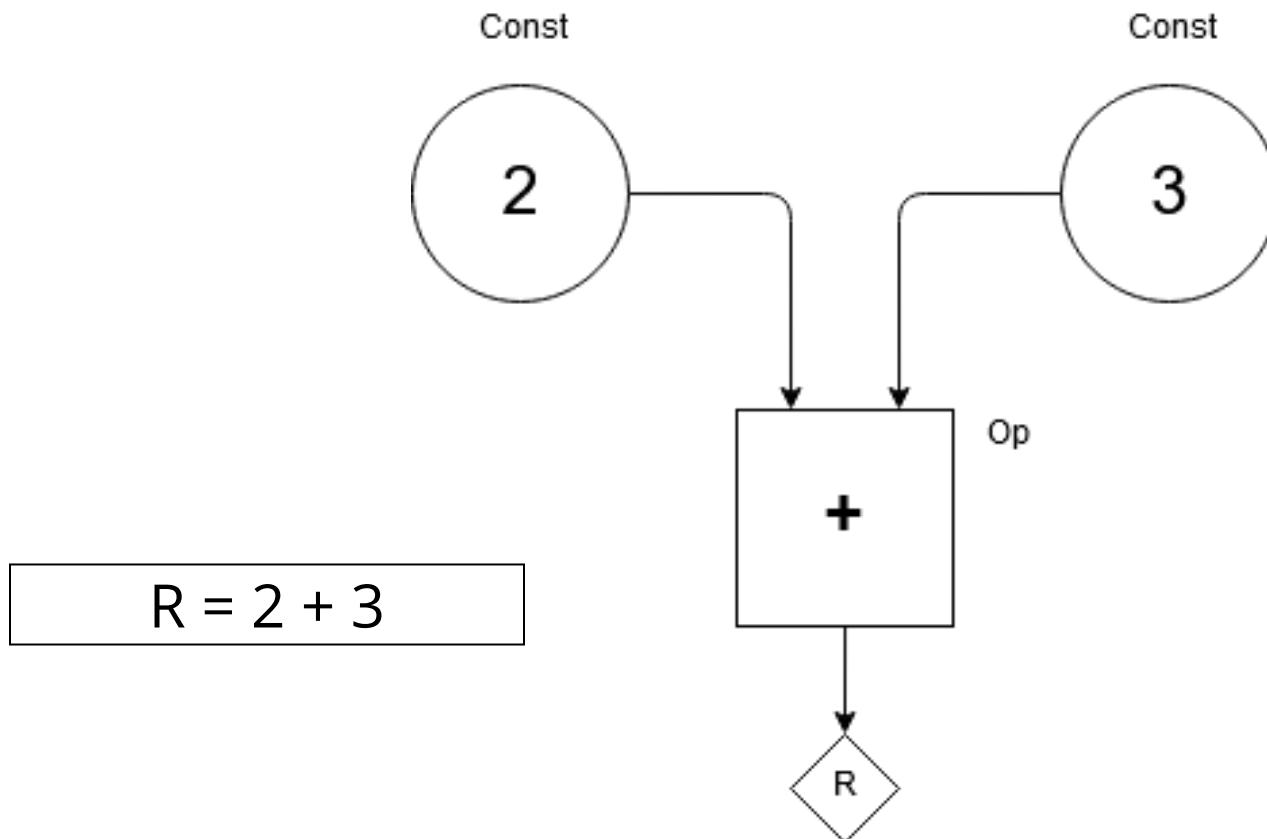
- Densas
- Convolucionales
- Recurrentes
- Secuencias
- Otras

Funciones

- Activación
- Coste
- Otras

<https://keras.io/layers/core/>

Ejemplo



Ejemplo 1 (TF 1)

```
1 import tensorflow as tf
2
3 # Define constants
4 a = tf.placeholder(tf.int16)
5 b = tf.placeholder(tf.int16)
6
7 # Define operation
8 add = tf.add(a, b)
9
10 # Compute in session
11 with tf.Session() as sess:
12     print("Addition with variables: ")
13     print("%i" % sess.run(add, feed_dict={a: 2, b: 3}))
```

Modelos

2 Formas de construirlos

Sequential

```
1 model = Sequential()  
2  
3 model.add(Dense(32, activation='relu',  
4                 input_dim=100))  
5  
6 model.add(Dense(1, activation='sigmoid'))  
7  
8 model.compile(optimizer='rmsprop',  
9                 loss='binary_crossentropy',  
10                metrics=['accuracy'])
```

Functional

```
1 inputs = Input(shape=(100,))  
2  
3 x = Dense(32, activation='relu')(inputs)  
4 x = Dense(1, activation='sigmoid')(x)  
5  
6 model = Model(inputs=inputs,  
7                 outputs=x)  
8  
9 model.compile(optimizer='rmsprop',  
10                loss='binary_crossentropy',  
11                metrics=['accuracy'])
```

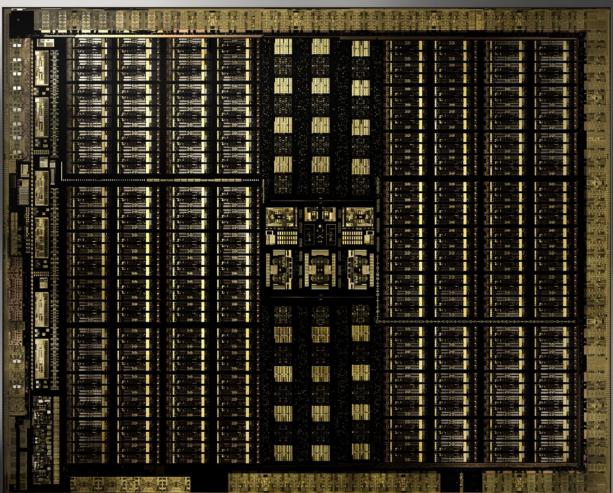
Ejemplo 2

Regresión

Ejemplo de algoritmo de regresión
utilizando la librería Tensorflow

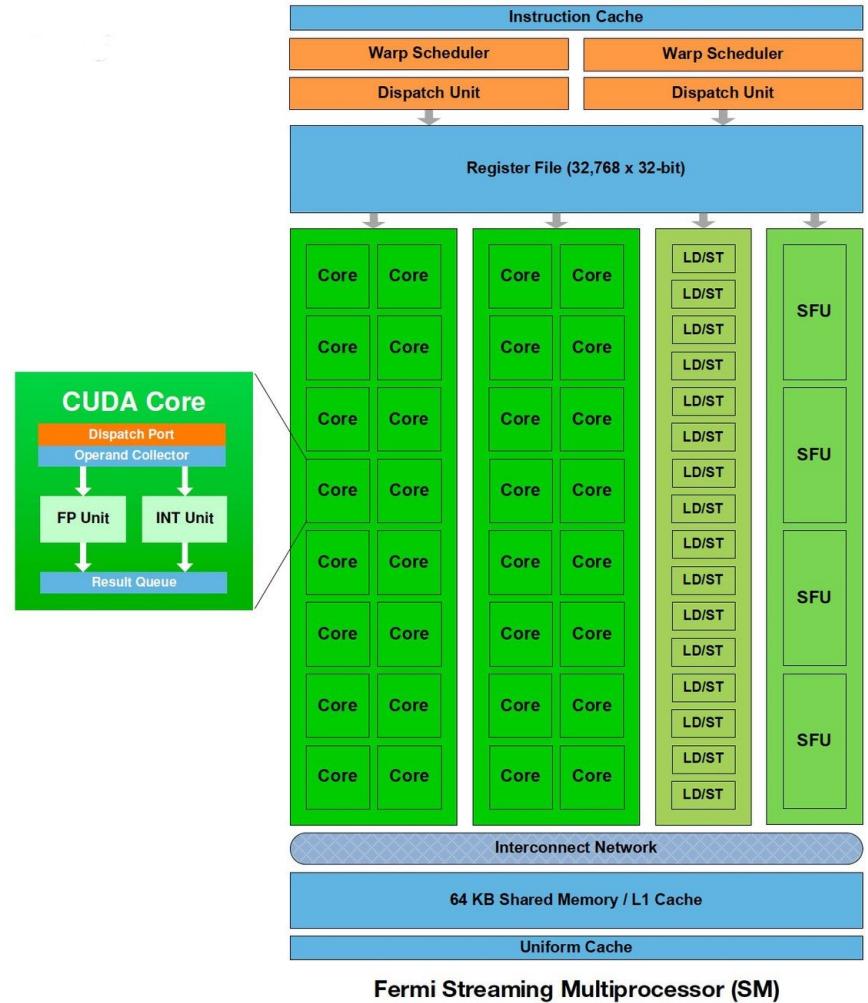
Computación Paralela

Tarjetas Gráficas



TURING

18.6 Billion xtors | 754 mm² | 48+48 GB 14GHz



Computación Paralela

TENSOR CORE

Mixed Precision Matrix Math - 4x4 matrices

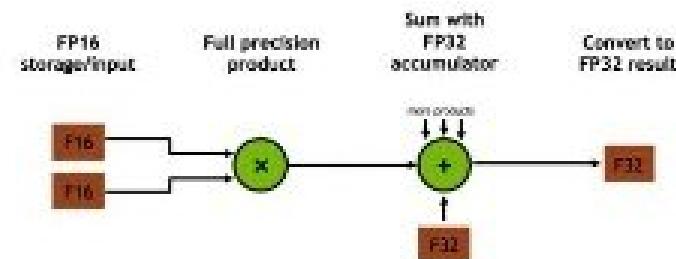
$$D = \begin{pmatrix} A_{00} & A_{01} & A_{02} & A_{03} \\ A_{10} & A_{11} & A_{12} & A_{13} \\ A_{20} & A_{21} & A_{22} & A_{23} \\ A_{30} & A_{31} & A_{32} & A_{33} \end{pmatrix}_{\text{FP16 or FP32}} \begin{pmatrix} B_{00} & B_{01} & B_{02} & B_{03} \\ B_{10} & B_{11} & B_{12} & B_{13} \\ B_{20} & B_{21} & B_{22} & B_{23} \\ B_{30} & B_{31} & B_{32} & B_{33} \end{pmatrix}_{\text{FP16}} + \begin{pmatrix} C_{00} & C_{01} & C_{02} & C_{03} \\ C_{10} & C_{11} & C_{12} & C_{13} \\ C_{20} & C_{21} & C_{22} & C_{23} \\ C_{30} & C_{31} & C_{32} & C_{33} \end{pmatrix}_{\text{FP16 or FP32}}$$

Legend: █ Activation Inputs █ Weight Inputs █ Output Results

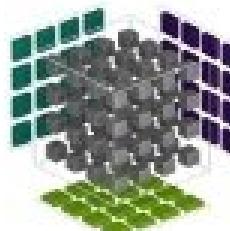
New CUDA TensorOp instructions
& data formats

4x4x4 matrix processing array

$$D[\text{FP32}] = A[\text{FP16}] * B[\text{FP16}] + C[\text{FP32}]$$



1000x1000x1000 TENSORS



Computación Paralela

Hardware específico

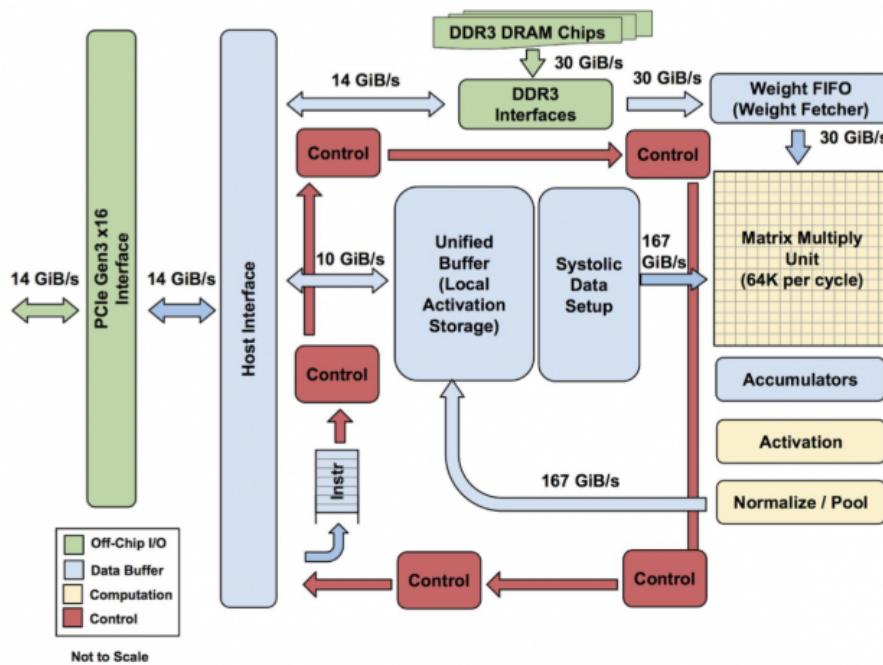


Figure 1. TPU Block Diagram. The main computation part is the yellow Matrix Multiply unit in the upper right hand corner. Its inputs are the blue Weight FIFO and the blue Unified Buffer (UB) and its output is the blue Accumulators (Acc). The yellow Activation Unit performs the nonlinear functions on the Acc, which go to the UB.

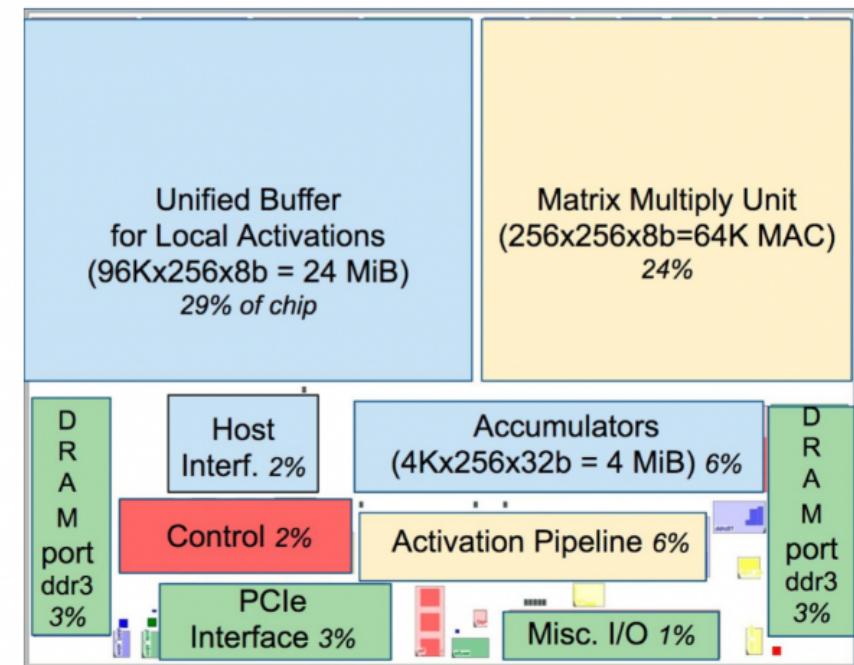
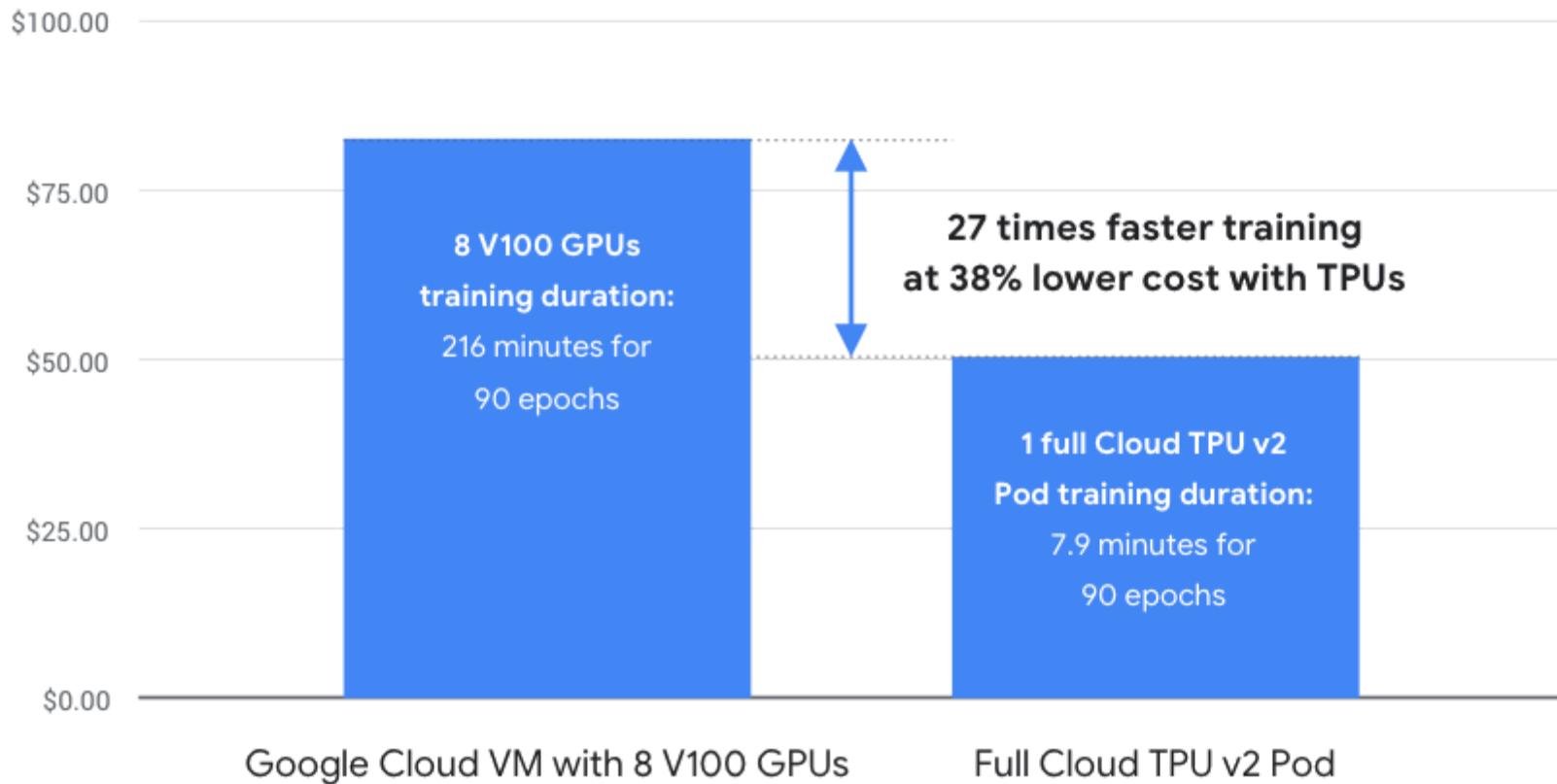


Figure 2. Floor Plan of TPU die. The shading follows Figure 1. The light (blue) data buffers are 37% of the die, the light (yellow) compute is 30%, the medium (green) I/O is 10%, and the dark (red) control is just 2%. Control is much larger (and much more difficult to design) in a CPU or GPU

Computación Paralela

Hardware específico

ResNet-50 Training Cost Comparison



Experimento 1

Cómputo en CPU

- Vamos a simular el número de cálculos de una red

¿Cómo calcular eficientemente M^n ?

- Necesitamos multiplicar matrices grandes
- Necesitamos mover una gran cantidad de datos
- Necesitamos paralelizar al máximo

Experimento 2

Cómputo en GPU

- Mismo ejemplo que en experimento 1

¿A que se deben estos cambios?

- Unidades dedicadas para cálculo matricial
- Uso de memoria RAM
- Distribución completa del problema

Conclusiones

Uso de CPU

- Lento para entrenar/experimentar
- Útil para realizar agregaciones/elaborar respuesta

Uso de GPU

- Buen rendimiento de entrenamiento
- Escalable

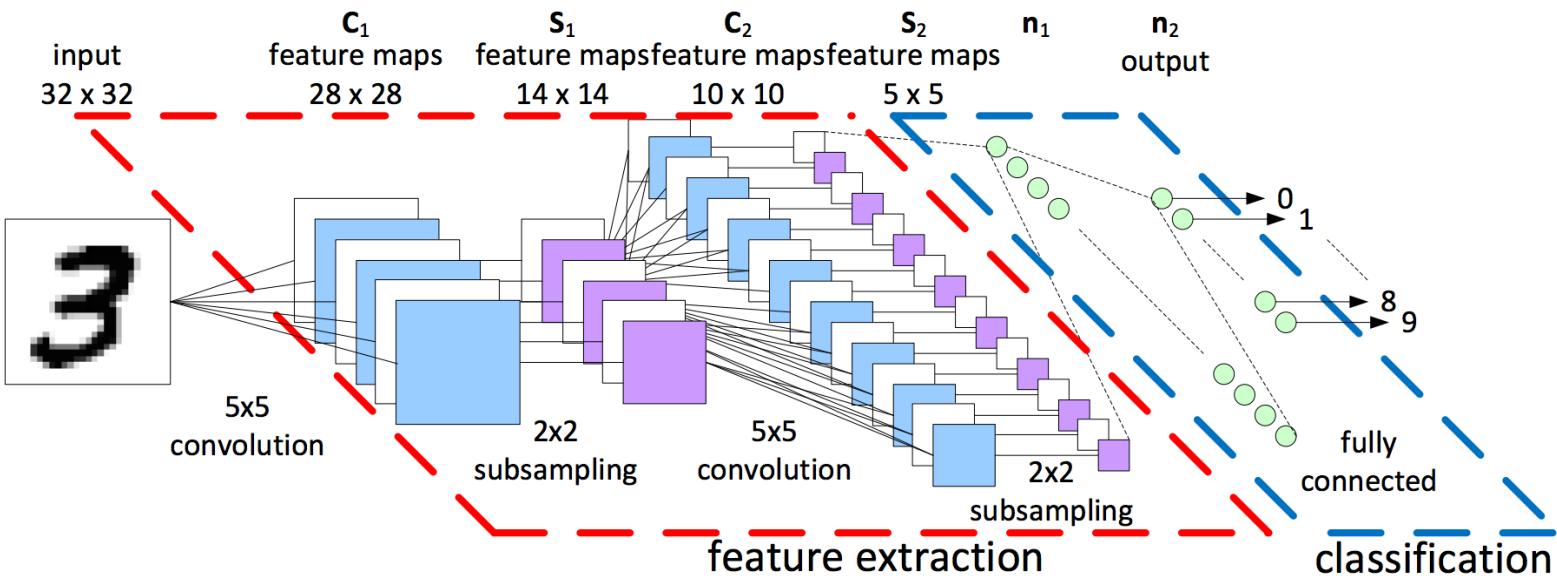
Ejemplo 3

Predecir diabetes

Implementar con TF2 una red MLP que sirva para predecir si una persona puede padecer diabetes en 5 años o no

Ejemplo 4

Red convolucional con TF2



Ecosistema TF2

