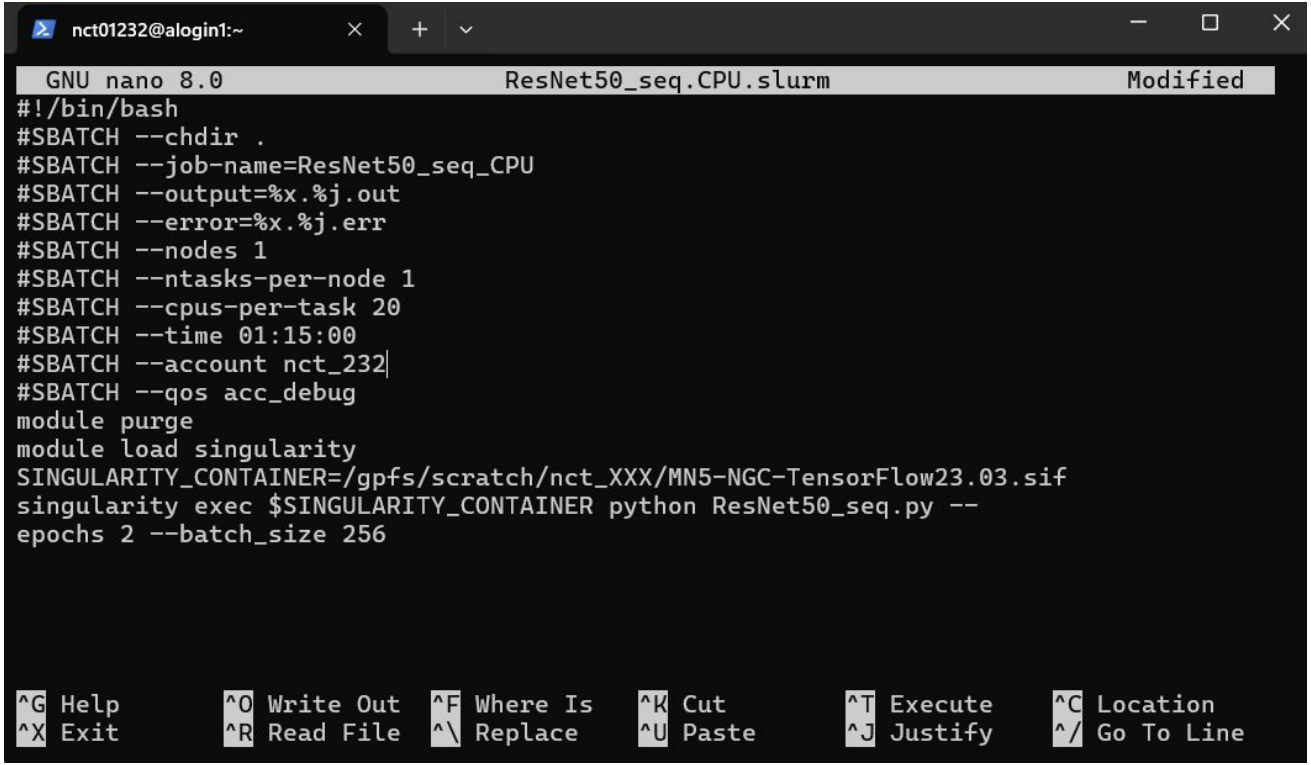
Lab 5: Parallel training of neural networks

Mario Ventura

1. Code Overview

- **Argumentos:** 5 épocas, 2048 tamaño lote
- **Datos:** CIFAR-10 redimensionado en 128x128
- **Modelo:** ResNet 50V2 para 10 clases
- **Entrenamiento:** Usa optimizador SGD

2. Entrenamiento en CPU



2. Entrenamiento en CPU

```
nct01232@alogin1:~
[nct01232@alogin1 ~]$ sbatch ResNet50_seq.CPU.slurm
Submitted batch job 17116453
[nct01232@alogin1 ~]$ squeue --start
             JOBID PARTITION     NAME     USER ST          START_TIME  NODES SCHEDNODES           NODELIST(REASON)
          17116453       acc ResNet50 nct01232 PD                 N/A      1 (null)              (Priority)
[nct01232@alogin1 ~]$
```

```
[nct01232@alogin1 ~]$ ls -l
total 5
-rw-r--r-- 1 nct01232 nct 1538 Mar  9 18:45 MNIST.py
-rw-r--r-- 1 nct01232 nct  450 Mar 10 17:57 MNIST.slurm
-rw-r--r-- 1 nct01232 nct  467 Mar 17 18:10 ResNet50_seq.CPU.slurm
-rw-r--r-- 1 nct01232 nct  407 Mar 17 18:10 ResNet50_seq_CPU.17116453.err
-rw-r--r-- 1 nct01232 nct    0 Mar 17 18:10 ResNet50_seq_CPU.17116453.out
drwxr-xr-x 2 nct01232 nct 4096 Mar  8 14:10 lab4
drwxr-xr-x 2 nct01232 nct 4096 Mar  9 19:07 singularity_cache
drwxr-xr-x 2 nct01232 nct 4096 Mar  9 19:07 singularity_tmp
[nct01232@alogin1 ~]$
```

2. Entrenamiento en CPU

Diferencia: 9s

```
[nct01232@alogin1 ~]$ cat ResNet50_seq_CPU.17117579.out
Epoch 1/2
196/196 - 246s - loss: 2.0349 - accuracy: 0.2521 - 246s/epoch - 1s/step
Epoch 2/2
196/196 - 241s - loss: 1.7586 - accuracy: 0.3552 - 241s/epoch - 1s/step
[nct01232@alogin1 ~]$
```

2. Entrenamiento en CPU

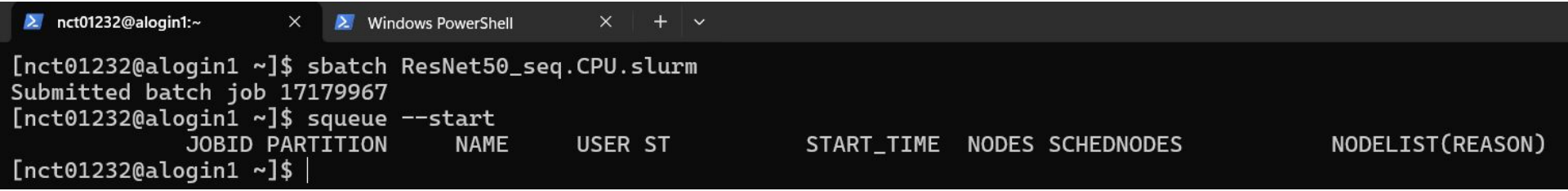
Diferencia: 9s

```
[nct01232@alogin1 ~]$ cat ResNet50_seq_CPU.17117579.out
Epoch 1/2
196/196 - 246s - loss: 2.0349 - accuracy: 0.2521 - 246s/epoch - 1s/step
Epoch 2/2
196/196 - 241s - loss: 1.7586 - accuracy: 0.3552 - 241s/epoch - 1s/step
[nct01232@alogin1 ~]$
```

- Entrenar un modelo sin GPUs no es práctico
- Puede ser que la etapa (epoch) 1 tarde más por la carga inicial

2. Entrenamiento en GPU

¿Y con GPUs?



nct01232@alogin1:~ | Windows PowerShell

GNU nano 8.0 ResNet50_seq.CPU.slurm Modified

```
#!/bin/bash
#SBATCH --chdir .
#SBATCH --job-name=ResNet50_seq_GPU
#SBATCH --output=%x.%j.out
#SBATCH --error=%x.%j.err
#SBATCH --nodes 1
#SBATCH --ntasks-per-node 1
#SBATCH --cpus-per-task 20 #Minimum cpus requested should
 # be (nodes * gpus/node * 20)
#SBATCH --time 01:15:00
#SBATCH --account nct_325
#SBATCH --qos acc_debug
#SBATCH --gres gpu:1

echo " "
echo "Job Name: $SLURM_JOB_NAME"
echo "Job ID: $SLURM_JOB_ID"
echo "Submit Directory: $SLURM_SUBMIT_DIR"
echo "Partition: $SLURM_JOB_PARTITION"
echo "Account: $SLURM_JOB_ACCOUNT"
echo " "
echo "Number of Nodes Allocated: $SLURM_JOB_NUM_NODES"
echo "Total number of tasks: $SLURM_NTASKS"
echo "Number of tasks per node: $SLURM_NTASKS_PER_NODE"
echo "Number of CPU cores per task: $SLURM_CPUS_PER_TASK"
echo " "
echo "Specific GPUs allocated:"

nvidia-smi
module purge
module load singularity/4.1.5

SINGULARITY_CONTAINER=/gpfs/scratch/nct_325/MN5-NGC-TensorFlow-23.03.sif
singularity exec --nv $SINGULARITY_CONTAINER python ResNet50_seq.py --epochs 1 --batch_size 256
```

^G Help ^O Write Out ^F Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-] To Bracket
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy ^B Where Was

2. Entrenamiento en GPU

```
Specific GPUs allocated:
Tue Mar 18 16:29:19 2025
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 535.86.10              Driver Version: 535.86.10    CUDA Version: 12.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA H100                    On  | 00000000:2C:00.0 Off |                    0 |
| N/A   38C    P0              65W / 700W |      2MiB / 65247MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
196/196 - 24s - loss: 2.0328 - accuracy: 0.2547 - 24s/epoch - 125ms/step
[nct01232@alogin1 ~]$
```

3. CPU vs GPU

CPU: 246s / época

GPU: 24s / época

- **11x más rápido en GPU.**
- **2083 imágenes/s (GPU) vs 172 imágenes/s (CPU)**

3. CPU vs GPU

CPU: 246s / época

GPU: 24s / época

- **11x más rápido en GPU.**
- **2083 imágenes/s (GPU) vs 172 imágenes/s (CPU)**
- **Conclusión: GPU is the winner**

4. Métricas de rendimiento

4.1 ACELERACIÓN (speedrun)

Razón entre tiempo secuencial y paralelo (1 vs n GPUs)

- Ejemplo: “1 GPU takes 80 seconds, and with 4 GPUs it takes 25 seconds, then: Speedup=80/25=3.2×”
- Ideal: **4x**

4. Métricas de rendimiento

4.2 THROUGHPUT

Velocidad con la que “algo” se procesa.

Velocidad **EFECTIVA**.

- Proporciona **información valiosa para evaluar rendimiento computacional**.
- Ejemplo: imágenes procesadas / t.
- 100,000 imágenes / 100s = 1000 imgs/s.

4. Métricas de rendimiento

4.3 ESCALABILIDAD

La capacidad de un sistema para manejar eficientemente cantidades de trabajo crecientes/decrecientes.

Capacidad de escalar (manejar más trabajo).

Depende mayoritariamente de factores como configuraciones, tipo de arquitectura de la red, eficiencia del framework de deep learning, etc.

4. Métricas de rendimiento

Son de gran utilidad para evaluar la eficiencia, escalabilidad, etc. de entrenamientos de modelos con multi-GPU

5. Entrenamiento paralelo (TF)

Mirrored Strategy

- Permite aplicar la estrategia llamada “Mirrored Strategy”
- tf.distribute.MirroredStrategy permite entrenamiento paralelo
- *tf.distribute.MirroredStrategy* está incluido en el conjunto de estrategias de *tf.distribute.Strategy*

```
mirrored_strategy = tf.distribute.MirroredStrategy()
```

5. Entrenamiento paralelo (TF)

Funcionamiento del Mirrored Strategy

- Replicar el modelo en cada GPU
- Cada GPU procesa un subconjunto de los datos.
- Actualizaciones sincronizadas

```
devices = tf.config.experimental.list_physical_devices("GPU")
```

```
mirrored_strategy =
tf.distribute.MirroredStrategy(devices=["/gpu:0", "/gpu:1"])
```

6. Ejecución paralela de ResNet50

```
[nct01232@alogin1 ~]$ cat ResNet50.slurm
#!/bin/bash
#SBATCH --chdir .
#SBATCH --job-name=ResNet50
#SBATCH --output=%x.%j.out
#SBATCH --error=%x.%j.err
#SBATCH --nodes 1
#SBATCH --ntasks-per-node 1
#SBATCH --gres gpu:4
#SBATCH --cpus-per-task 80
#SBATCH --time 01:15:00
#SBATCH --account nct_325
#SBATCH --qos acc_debug

echo "PARALLEL EXECUTION USING 1, 2, 4 GPUS"
echo " "
echo "Job Name: $SLURM_JOB_NAME"
echo "Job ID: $SLURM_JOB_ID"
echo "Submit Directory: $SLURM_SUBMIT_DIR"
echo "Partition: $SLURM_JOB_PARTITION"
echo "Account: $SLURM_JOB_ACCOUNT"
echo " "

echo "Number of Nodes Allocated: $SLURM_JOB_NUM_NODES"
echo "Total number of tasks: $SLURM_NTASKS"
echo "Number of tasks per node: $SLURM_NTASKS_PER_NODE"
echo "Number of CPU cores per task: $SLURM_CPUS_PER_TASK"
echo " "
echo "Specific GPUs allocated:"
nvidia-smi

module purge
module load singularity/4.1.5

SINGULARITY_CONTAINER=/gpfs/scratch/nct_325/MN5-NGC-TensorFlow-23.03.sif

singularity exec --nv $SINGULARITY_CONTAINER python ResNet50.py --epochs 5 --batch_size 2048 --n_gpus 1
singularity exec --nv $SINGULARITY_CONTAINER python ResNet50.py --epochs 5 --batch_size 4096 --n_gpus 2
singularity exec --nv $SINGULARITY_CONTAINER python ResNet50.py --epochs 5 --batch_size 8192 --n_gpus 4
[nct01232@alogin1 ~]$
```

6. Ejecución paralela de ResNet50

```
[nct01232@alogin1 ~]$ sbatch ResNet50.slurm
Submitted batch job 17187550
[nct01232@alogin1 ~]$ squeue --start
             JOBID PARTITION     NAME     USER ST          START_TIME  NODES SCHEDNODES           NODELIST(REASON)
          17187550       acc ResNet50 nct01232 PD                 N/A      1 (null)              (None)
[nct01232@alogin1 ~]$
```

```
[nct01232@alogin1 ~]$ ls -l
total 7
-rw-r--r-- 1 nct01232 nct 1538 Mar  9 18:45 MNIST.py
-rw-r--r-- 1 nct01232 nct  450 Mar 10 17:57 MNIST.slurm
-rw-r--r-- 1 nct01232 nct  624 Mar 18 17:42 ResNet50.17187550.err
-rw-r--r-- 1 nct01232 nct 3116 Mar 18 17:42 ResNet50.17187550.out
-rw-r--r-- 1 nct01232 nct 1202 Mar 18 17:35 ResNet50.py
-rw-r--r-- 1 nct01232 nct 1219 Mar 18 17:39 ResNet50.slurm
-rw-r--r-- 1 nct01232 nct 1026 Mar 18 16:28 ResNet50_seq.CPU.slurm
-rw-r--r-- 1 nct01232 nct  848 Mar 17 18:44 ResNet50_seq.py
drwxr-xr-x 2 nct01232 nct 4096 Mar  8 14:10 lab4
drwxr-xr-x 2 nct01232 nct 4096 Mar  9 19:07 singularity_cache
drwxr-xr-x 2 nct01232 nct 4096 Mar  9 19:07 singularity_tmp
[nct01232@alogin1 ~]$
```

6. Ejecución paralela de ResNet50

```
Epoch 1/5
25/25 - 29s - loss: 2.2495 - accuracy: 0.1618 - 29s/epoch - 1s/step
Epoch 2/5
25/25 - 14s - loss: 2.1402 - accuracy: 0.2211 - 14s/epoch - 544ms/step
Epoch 3/5
25/25 - 14s - loss: 2.0531 - accuracy: 0.2533 - 14s/epoch - 541ms/step
Epoch 4/5
25/25 - 14s - loss: 1.9943 - accuracy: 0.2773 - 14s/epoch - 545ms/step
Epoch 5/5
25/25 - 13s - loss: 1.9144 - accuracy: 0.3053 - 13s/epoch - 540ms/step
Epoch 1/5
13/13 - 32s - loss: 2.2787 - accuracy: 0.1514 - 32s/epoch - 2s/step
Epoch 2/5
13/13 - 8s - loss: 2.1381 - accuracy: 0.2145 - 8s/epoch - 623ms/step
Epoch 3/5
13/13 - 7s - loss: 2.0521 - accuracy: 0.2506 - 7s/epoch - 575ms/step
Epoch 4/5
13/13 - 8s - loss: 1.9814 - accuracy: 0.2790 - 8s/epoch - 623ms/step
Epoch 5/5
13/13 - 8s - loss: 1.9227 - accuracy: 0.3049 - 8s/epoch - 613ms/step
Epoch 1/5
7/7 - 47s - loss: 2.3040 - accuracy: 0.1436 - 47s/epoch - 7s/step
Epoch 2/5
7/7 - 5s - loss: 2.1782 - accuracy: 0.1977 - 5s/epoch - 729ms/step
Epoch 3/5
7/7 - 5s - loss: 2.0979 - accuracy: 0.2302 - 5s/epoch - 703ms/step
Epoch 4/5
7/7 - 4s - loss: 2.0452 - accuracy: 0.2549 - 4s/epoch - 632ms/step
Epoch 5/5
[nct01232@alogin1 ~]$
```

6. Ejecución paralela de ResNet50

1 GPU

Promedio: 13’75s/época

Throughput: 50.000 / 13’75 ≃ 3.636 img/s

6. Ejecución paralela de ResNet50

2 GPUs

Promedio: 7’75s/época

Throughput: 50.000 / 7’75 ≃ 6.452 img/s

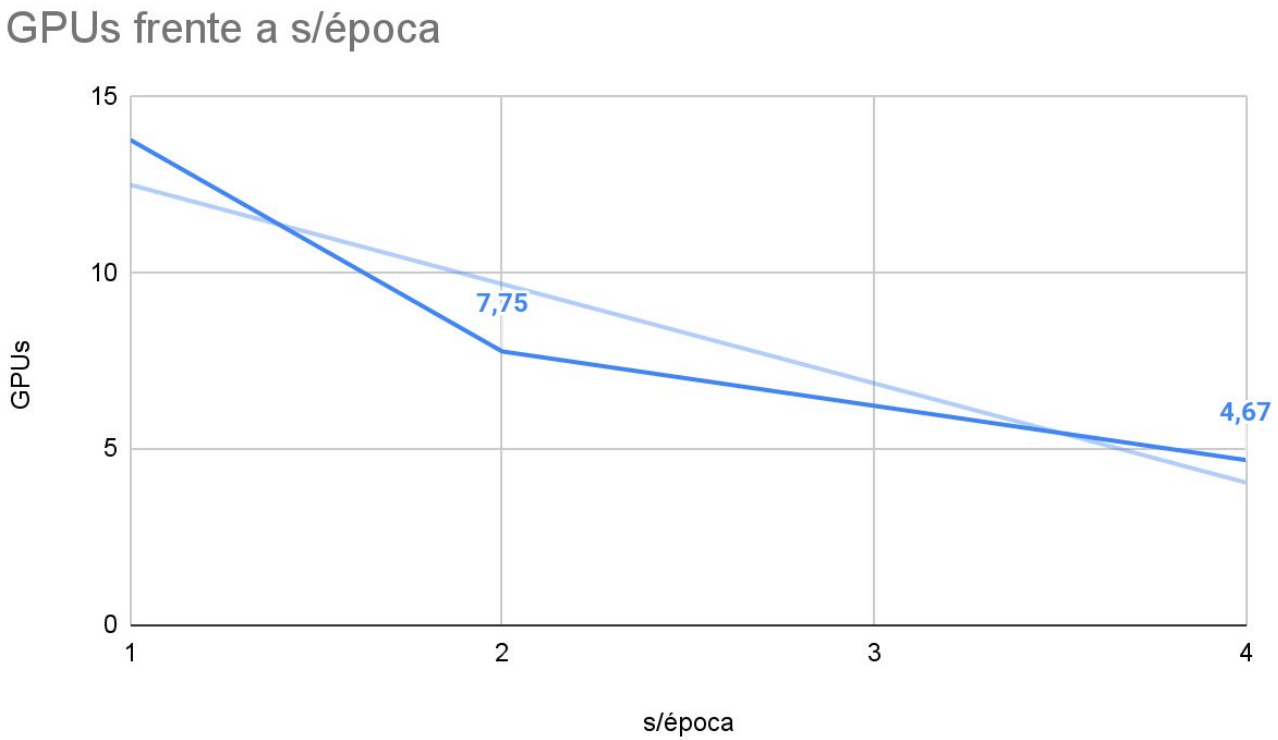
6. Ejecución paralela de ResNet50

4 GPUs

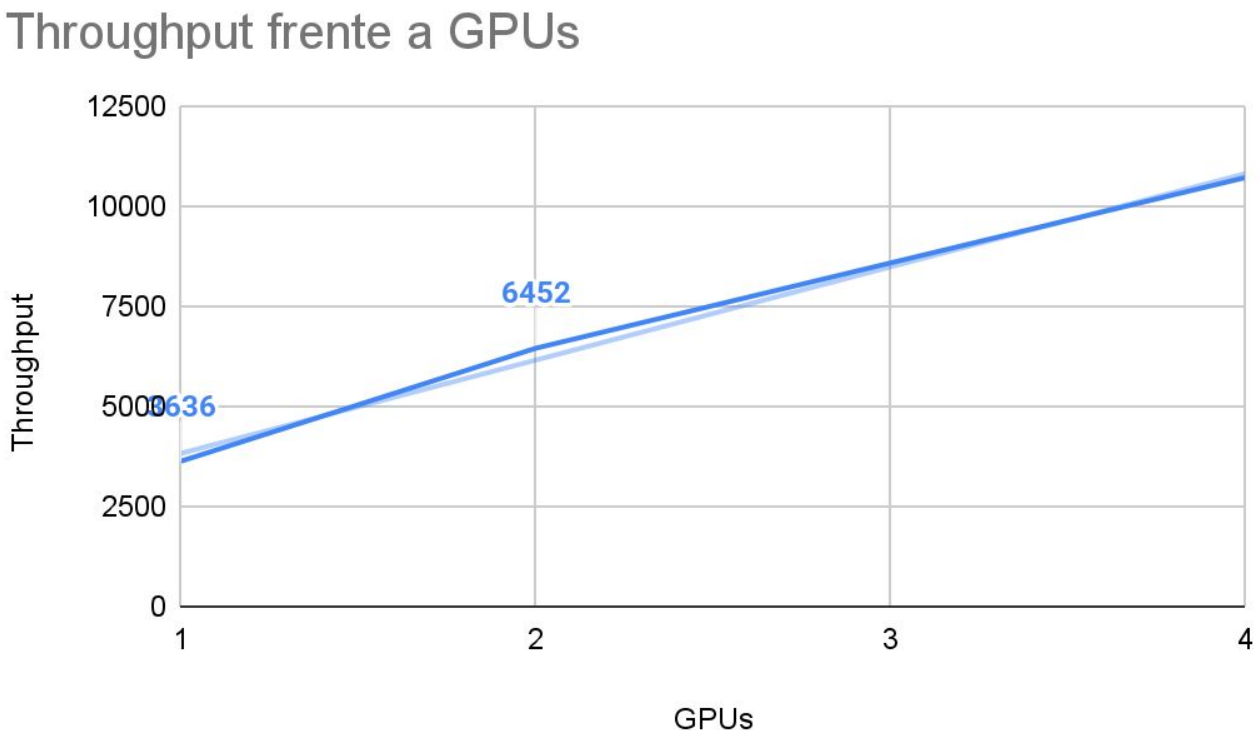
Promedio: 4’67s/época

Throughput: 50.000 / 4’67 ≃ 10.707 img/s

6. Ejecución paralela de ResNet50

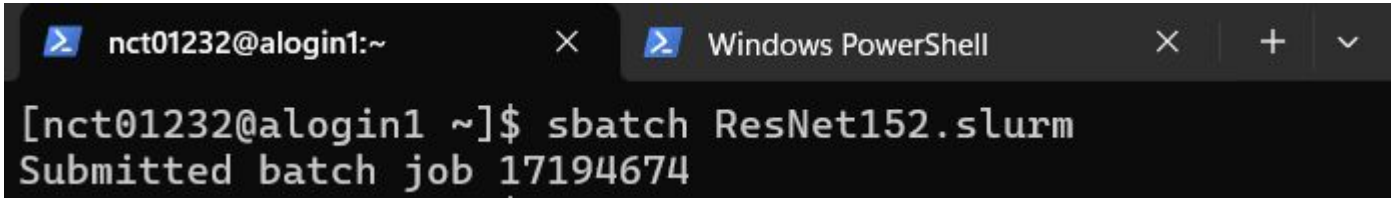


6. Ejecución paralela de ResNet50



7. ResNet152

- Modificar **.slurm**
- Modificar **.py**
- **Disminuir batch size** (512, 1024, 2048)



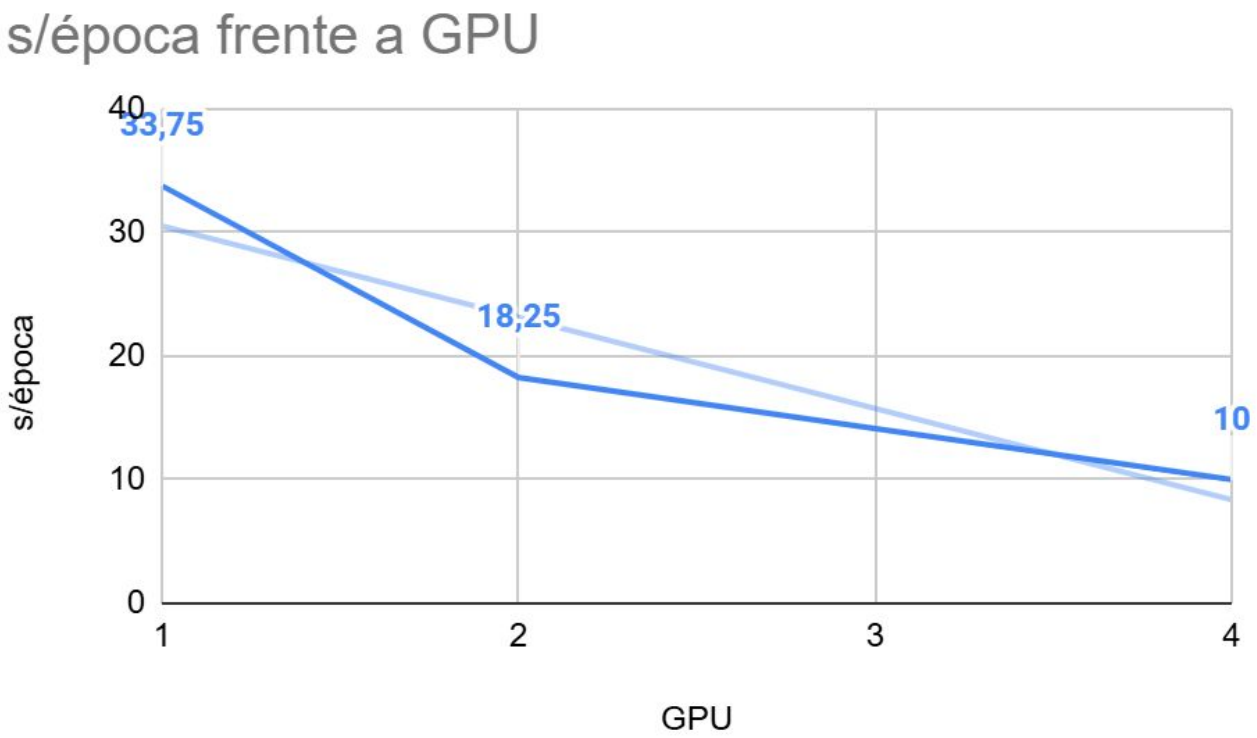
7. ResNet152

Información de archivos .err

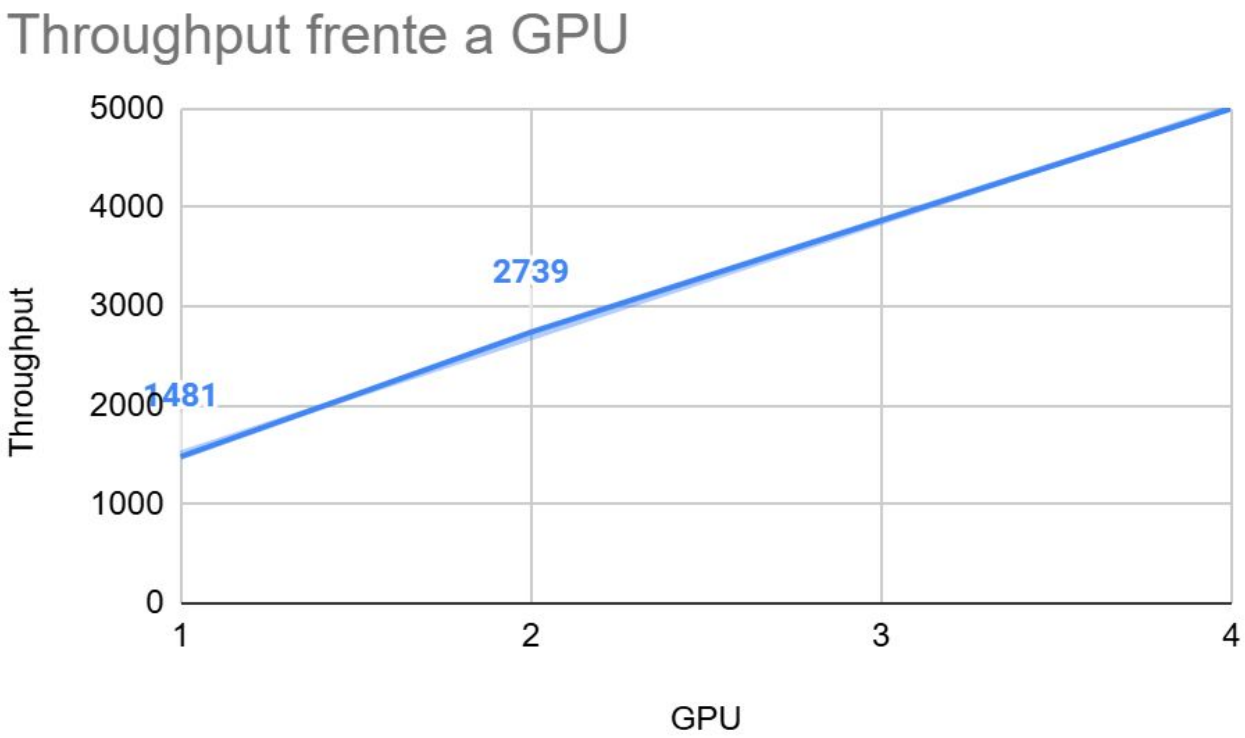
```
  key: "replicate_on_split"
  value {
    b: false
  }
}
experimental_type {
  type_id: TFT_PRODUCT
  args {
    type_id: TFT_DATASET
    args {
      type_id: TFT_PRODUCT
      args {
        type_id: TFT_TENSOR
        args {
          type_id: TFT_UINT8
        }
      }
      args {
        type_id: TFT_TENSOR
        args {
          type_id: TFT_UINT8
        }
      }
    }
  }
}

2025-03-18 19:11:50.345170: I tensorflow/compiler/xla/stream_executor/cuda/cuda_dnn.cc:428] Loaded cuDNN version 8801
2025-03-18 19:11:51.588974: I tensorflow/compiler/xla/stream_executor/cuda/cuda_blas.cc:648] TensorFloat-32 will be used for the matrix multiplication. This
 will only be logged once.
2025-03-18 19:11:51.881842: I tensorflow/compiler/xla/service/service.cc:173] XLA service 0xba0df70 initialized for platform CUDA (this does not guarantee t
hat XLA will be used). Devices:
2025-03-18 19:11:51.881881: I tensorflow/compiler/xla/service/service.cc:181]   StreamExecutor device (0): NVIDIA H100, Compute Capability 9.0
2025-03-18 19:11:51.883501: I tensorflow/compiler/xla/service/service.cc:181]   StreamExecutor device (1): NVIDIA H100, Compute Capability 9.0
2025-03-18 19:11:51.883508: I tensorflow/compiler/xla/service/service.cc:181]   StreamExecutor device (2): NVIDIA H100, Compute Capability 9.0
2025-03-18 19:11:51.883511: I tensorflow/compiler/xla/service/service.cc:181]   StreamExecutor device (3): NVIDIA H100, Compute Capability 9.0
2025-03-18 19:11:52.181796: I tensorflow/compiler/jit/xla_compilation_cache.cc:480] Compiled cluster using XLA!  This line is logged at most once for the li
fetime of the process.
[nct01232@alogin1 ~]$
```

```
Epoch 1/5
98/98 - 62s - loss: 2.2160 - accuracy: 0.1742 - 62s/epoch - 637ms/step
Epoch 2/5
98/98 - 34s - loss: 2.0167 - accuracy: 0.2574 - 34s/epoch - 342ms/step
Epoch 3/5
98/98 - 33s - loss: 1.8923 - accuracy: 0.3058 - 33s/epoch - 341ms/step
Epoch 4/5
98/98 - 34s - loss: 1.7938 - accuracy: 0.3440 - 34s/epoch - 342ms/step
Epoch 5/5
98/98 - 33s - loss: 1.7049 - accuracy: 0.3751 - 33s/epoch - 341ms/step
Epoch 1/5
49/49 - 75s - loss: 2.2119 - accuracy: 0.1739 - 75s/epoch - 2s/step
Epoch 2/5
49/49 - 19s - loss: 2.0039 - accuracy: 0.2596 - 19s/epoch - 379ms/step
Epoch 3/5
49/49 - 18s - loss: 1.9089 - accuracy: 0.2987 - 18s/epoch - 371ms/step
Epoch 4/5
49/49 - 18s - loss: 1.8232 - accuracy: 0.3368 - 18s/epoch - 369ms/step
Epoch 5/5
49/49 - 18s - loss: 1.7447 - accuracy: 0.3650 - 18s/epoch - 372ms/step
Epoch 1/5
25/25 - 124s - loss: 2.2577 - accuracy: 0.1657 - 124s/epoch - 5s/step
Epoch 2/5
25/25 - 10s - loss: 2.0728 - accuracy: 0.2417 - 10s/epoch - 399ms/step
Epoch 3/5
25/25 - 10s - loss: 1.9614 - accuracy: 0.2797 - 10s/epoch - 394ms/step
Epoch 4/5
25/25 - 10s - loss: 1.8309 - accuracy: 0.3267 - 10s/epoch - 400ms/step
Epoch 5/5
25/25 - 10s - loss: 1.7452 - accuracy: 0.3537 - 10s/epoch - 403ms/step
[nct01232@alogin1 ~]$
```

7. ResNet152



7. ResNet152


Throughput frente a GPU
5000
4000
3000
2000
1000
0
Throughput
1481
2739
1
2
3
4
GPU

8. ResNet152 vs ResNet50

Comparación de resultados

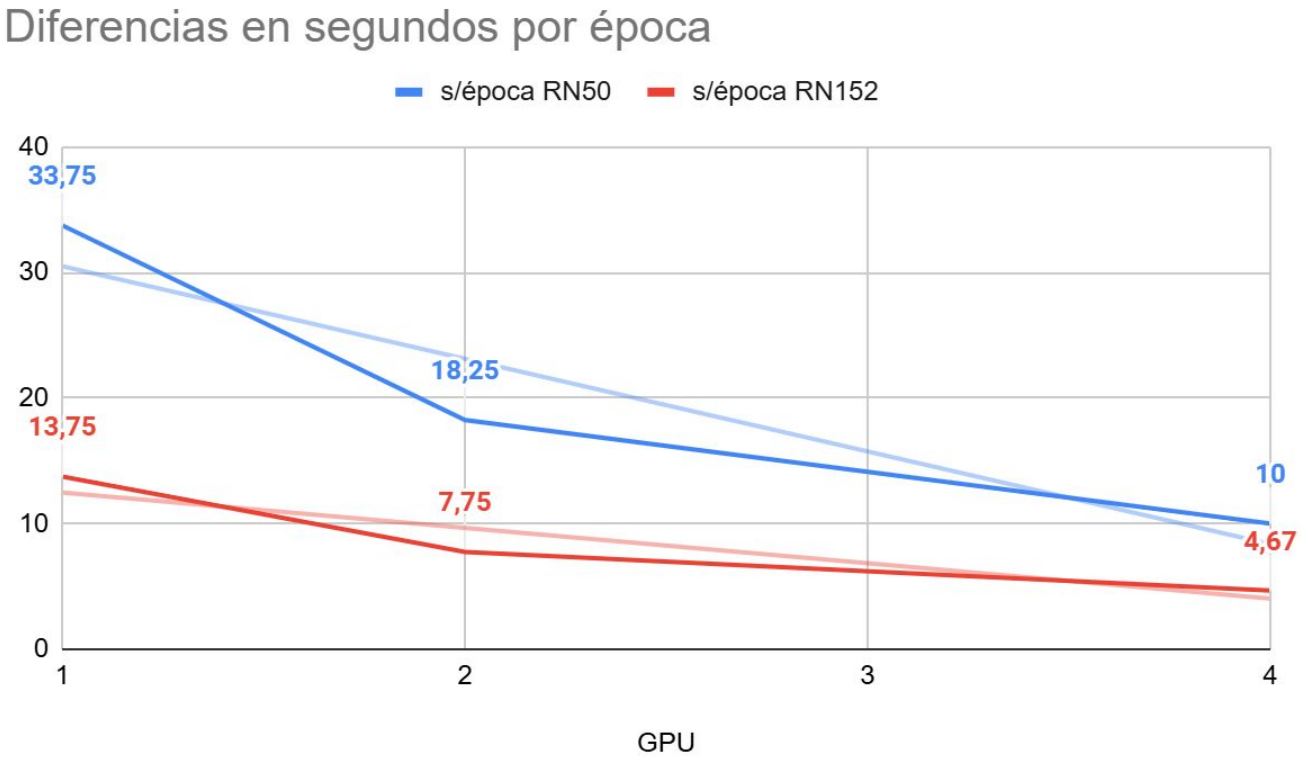
ResNet152V2

GPU	s/época	Throughput
1	33,75	1481
2	18,25	2739
4	10	5000

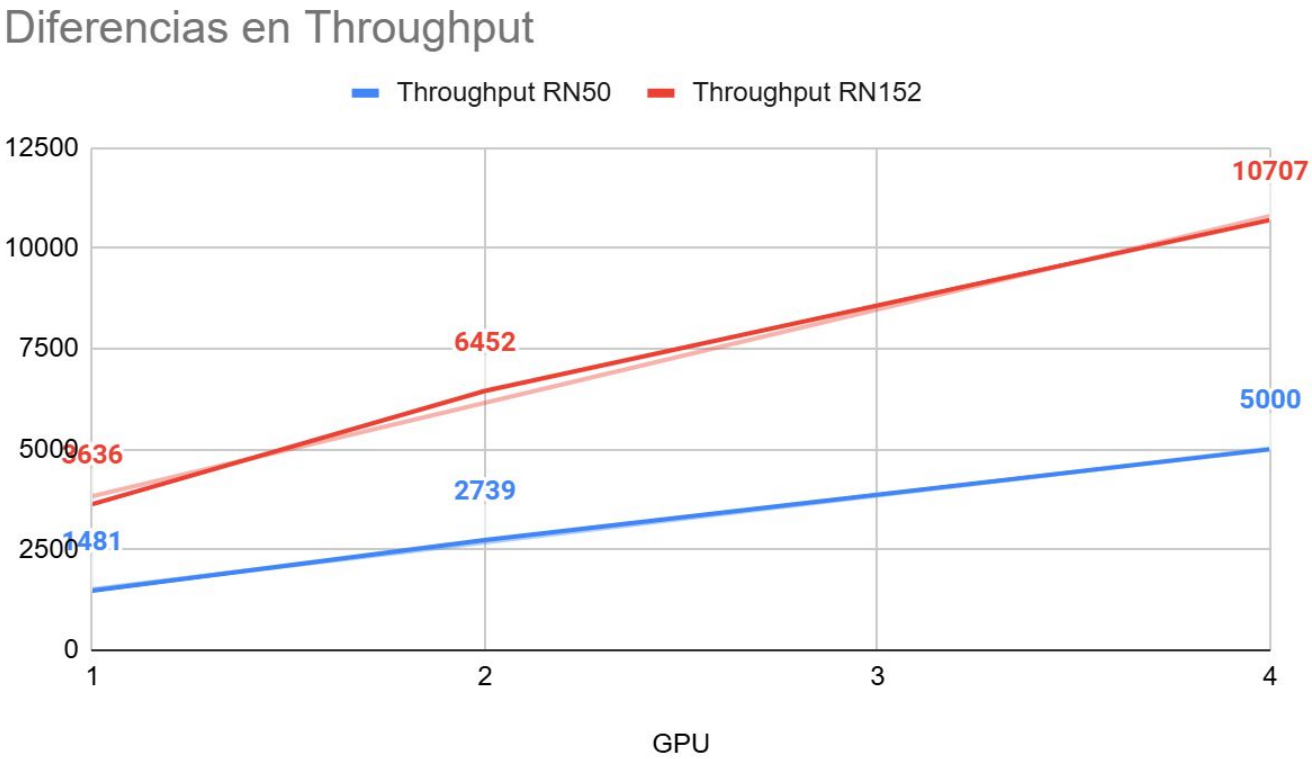
ResNet50

GPUs	s/época	Throughput
1	13,75	3636
2	7,75	6452
4	4,67	10.707

8. ResNet152 vs ResNet50



8. ResNet152 vs ResNet50



Gracias