Deep-Learning sobre Hardware de propósito general, análisis de rendimiento basado en contadores hardware

1.- Introducción y Objetivos

El objetivo marcado para este TFG es el análisis de rendimiento de aplicaciones del entorno Deep-Learning. Se trata de aplicaciones que se han extendido de manera significativa en los últimos años, y cuyas peculiaridades han propiciado la aparición de múltiples plataformas hardware para optimizar su ejecución (desde las GPUs hasta aceleradores como las TPU). A pesar de la existencia de dicho hardware, en muchas ocasiones este tipo de aplicaciones se ejecutarán sobre hardware de propósito general, por lo que resulta necesario analizar el rendimiento de este tipo de aplicaciones sobre una CPU.

En este TFG estudiaremos la forma de evaluar el rendimiento de aplicaciones escritas en TensorFlow a través de los contadores hardware de un procesador actual. El objetivo será doble: por un lado analizar aspectos generales a través de la metodología Top-Down, por otro lado, estudiar la evolución temporal (iteraciones, fases) de los eventos asociados al rendimiento.

2.- Fases de Desarrollo

Vamos a intentar definir los pasos a dar con la mayor precisión posible, para que el alumno tenga claro en todo momento en qué punto se encuentra y los próximos objetivos.

2.1.- Interacción del código Python con contadores hardware.

En esta primera fase analizaremos las distintas formas que existen de interaccionar, desde un código escrito en Python, con los contadores hardware del procesador. Se analizarán dos alternativas:

1. Analizar el código de Toplev [1], escrito en Python, para ver el uso que hace del comando perf [2].
2. Se buscará la forma de interactuar con la librería escrita en C/C++ PAPI [4], evaluando las distintas formas de interacción entre Python y C [3].

En esta fase se utilizarán códigos en Python muy sencillos para comprobar que el acceso a los contadores es correcto. Habrá que buscar la forma de validar que las cuentas se hacen de manera correcta. Es posible que haya que estudiar qué pasa con códigos Python con múltiples threads [5](será necesario para más adelante).

2.2.- Primeros pasos con aplicaciones de Deep Learning

En este punto será necesario aprender algunos aspectos básicos de Deep learning [6]. Qué son las redes neuronales profundas, cómo funcionan, etc. Se buscará material de apoyo apropiado para el alumno.

Simultáneamente, se empezará a trabajar con TensorFlow [7], para familiarizarse con el entorno. Instalación, entrenamiento e inferencia en un modelo sencillo.

2.3.- Evaluación de rendimiento en aplicaciones de Deep Learning

El siguiente paso consistirá en el análisis detallado de alguna aplicación real. Se utilizarán las correspondientes a los TFModels [8] de tensorflow. El objetivo será entender, mediante la medida de contadores hardware, aspectos tales como:

- Parte del procesador que impone el cuello de botella al rendimiento [9].
- Evolución de las métricas con las diferentes etapas de ejecución (forwarding vs. backpropagation, etapas de una CNN: convolución, pooling, reducción, etc.).

2.4.- Implicaciones sobre el rendimiento de la compartición de recursos

En muchas ocasiones, la ejecución de este tipo de aplicaciones se lleva a cabo en entornos compartidos (tipo cloud) en los que comparte recursos con otras aplicaciones. En este escenario, es interesante analizar la pérdida de rendimiento que se produce, así como el efecto de diferentes herramientas para paliarla.

Se comenzará trabajando con la herramienta del grupo BenchCast [10], que ya lleva a cabo este tipo de medidas de rendimiento, y se intentará incluir una aplicación de TFModels en la metodología.

Se medirá la degradación de rendimiento debido a las interacciones en la LLC y se estudiará el efecto de tecnologías de repartición de LLC proporcionadas en algunos procesadores (Intel Resource Director [11]).

3.- Finalización

La parte de desarrollo debería terminar con un conjunto de resultados y un repositorio software donde consultar el trabajo realizado. En este punto, se pasaría a la redacción de la memoria.

4.- Referencias

[1] Toplev: https://github.com/andikleen/pmu-tools/wiki/toplev-manual
[2] Linux perf: https://perf.wiki.kernel.org/index.php/Main_Page
[3] C-Python binding: https://realpython.com/python-bindings-overview/
[4] PAPI: https://icl.utk.edu/papi/
[5] Python threading: https://realpython.com/intro-to-python-threading/
[6] Deep Learning: https://www.tensorflow.org/resources/learn-ml?hl=es-419
[7] Tutoriales TensorFlow: https://www.tensorflow.org/tutorials?hl=es-419
[8] TF Model Garden: https://github.com/tensorflow/models
[9] A Top-Down method for performance analysis and counters architecture: https://ieeexplore.ieee.org/document/6844459
[10] BenchCast: https://github.com/prietop/BenchCast
[11] Intel Resource Director: https://software.intel.com/content/www/us/en/develop/articles/introduction-to-cache-allocation-technology.html