Integração de sensores de baixo-nível com TensorBoard

Julio Kiyoshi R. Matsoui[1], Matheus Fernandes[1], Lucas Wanner[1], Sandro Rigo[1]

[1]Instituto de Computação – Universidade Estadual de Campinas (UNICAMP)
Campinas – SP – Brasil
j200483@dac.unicamp.br, m222228@dac.unicamp.br
wanner@unicamp.br, srigo@unicamp.br

Abstract. *Machine learning systems require substantial computational and energy resources, particularly for training tasks. Profilers for these systems, such as the available on the TensorBoard tool for TensorFlow, provide insight into performance hotspots and optimization opportunities. Nevertheless, such tools typically lack energy profiling capabilities. In this work we present an power dissipation profiling plugin for machine learning tasks based on low-level IPMI sensors. The plugin is integrated with TensorBoard and demonstrated with an IBM POWER machine.*

Resumo. *Sistemas de aprendizado de máquina exigem grandes recursos computacionais e energéticos, especialmente para tarefas de treinamento. Perfiladores para estes sistemas, como o disponível na ferramenta TensorBoard para TensorFlow, dão informações sobre* hotspots *e oportunidades para otimização. Ainda assim, estas ferramentas tipicamente não incluem perfilamento de energia. Neste trabalho apresentamos um plugin para perfilamento de dissipação de potência para tarefas de aprendizado de máquina baseado em sensores IPMI de baixo nível. O plugin é integrado com TensorBoard e demonstrado com uma máquina IBM POWER.*

1. Introdução

O TensorFlow[12] foi desenvolvido pela Google como uma API (Application Programming Interface) para Python e se consolidou como uma das principais bibliotecas de código aberto para desenvolver modelos de aprendizado de máquina (*Machine Learning*) e apredizado profundo (*Deep Learning*). Para visualizar os modelos criados temos o TensorBoard[10], que é um kit de ferramentas que permite visualização gráfica de seus modelos, sendo possível compreender o modelo utilizado, depurar gargalos e, a partir dessa análise, otimizá-lo. O Tensorboard utiliza uma estratégia de plugins para adicionar novas ferramentas como, por exemplo, o plugin *Profile*. Dentro do Profile temos algumas ferramentas como o *Trace Viewer*, mostrando uma linha do tempo dos diferentes eventos que ocorreram na CPU e na GPU durante o período de criação do seu modelo, assim é possível diagnosticar e corrigir gargalos no *pipeline* de entrada. Neste artigo iremos mostrar a criação de um plugin de sofware livre para o Tensorboard com o intuito de adicionar um recurso ao Trace Viewer. Dessa maneira, além de mostrar a linha temporal dos eventos que estão acontecendo na CPU e GPU, irá mostrar o dissipação de potência naquele instante de tempo.

Para obter os dados de dissipação de potência em Watts, utilizamos o software ipmitool[3], o qual permite acesso aos sensores e controladores disponibilizados pela

implementação da interface IPMI[2], a qual por sua vez representa um conjunto de especificações para um sistema autônomo de gerenciamento e obtenção de informação sobre o estado do hardware. Em nosso caso, foram utilizados os sensores de dissipação de potência das CPUs POWER9[5] e da máquina como um todo. No entanto, a partir do ipmitool é possível obter uma lista completa dos sensores disponíveis em POWER pelo IPMI.

2. Projeto e Implementação

Para obtenção de valores gerais de potência (usaremos este termo no artigo como sinônimo de dissipação de potência em Watts) através do ipmitool, usamos o comando de terminal *$ sudo ipmitool dcmi power reading*. Esse comando tem como saída medidas do tipo: leitura instantânea de dissipação de potência, período de coleta, valor máximo, médio e mínimo durante este perído, além da marca temporal da medição e do status da leitura de potência.

Na obtenção de valores de sensores específicos, outros comandos são nessários. Pode-se obter a lista completa de sensores disponibilizados pelo IPMI através do comando *$ sudo ipmitool sensor list*, o qual devolve além dos IDs dos sensores, também as suas medições. Evita-se, porém, o uso deste último método para coleta de dados, visto que obtê-los todos de uma vez tal como o comando faz é mais custoso do que coletá-los individualmente. Para isso, é utilizado o comando *$ ipmitool sensor get Sensor ID* , em que troca-se *Sensor ID* pelo ID obtido através da listagem anterior. Através desse método, obtém-se a leitura do sensor, o seu tipo, status e uma série de limites usados para controle do estado da máquina.

Como a saída dos comandos do ipmitool são em formato de texto, é preciso fazer um filtro deste, eliminando pulos de linhas, espaços e outras informações desnecessárias, de modo a obter apenas os valores necessários. No nosso caso, foi decidido usar um script em Python para fazer *parsing* desses valores e armazená-los em um arquivo CSV para consulta e plotagem posterior.

Com o intuito de integrar esses sensores com *profilers* de alto nível, tal como o oferecido pelo TensorBoard, decidimos criar um plugin para o TensorBoard que pudesse obter a dissipação de potência durante a execução de algortimos que usam a API do TensorFlow, de modo a verificar, por exemplo, como se comporta a potência de uma máquina ao treinar modelos de Machine Learning. Nomeamos este plugin de PowerBoard[6]. Como o plugin de Profile[8] citado consegue produzir uma visualização de cada processo executado ao longo do tempo, decidimos integrar com essa visualização um outro gráfico de dissipação de potência também ao longo do tempo, permitindo, assim, uma fácil comparação de modo a verificar, por exemplo, quais funções dentro de um modelo apresentam valores maiores de potência.

Como o TensorBoard atua como um programa de servidor web, o PowerBoard funciona como uma aplicação para esse servidor, a qual possui tanto Back-End como Front-End.

O Back-End, representado na Figura 1 como a caixa `plugin.py`, possui uma série de regras determinadas pelo TensorBoard, as quais podem ser encontradas na referência[11] deste artigo, mas cujo funcionamento pode ser resumido em funções que

representam rotas dentro da aplicação. O Front-End, portanto, poderá acessar dados passados do Back-End para o Front-End a partir dessas rotas.

O Front-End, no entanto, já possui certa liberdade, podendo ser feito em qualquer *framework* desejado, contanto que haja um ES Module[1] como ponto de entrada. Preferiu-se utilizar um script em Javascript como base para o nosso Front-end.

Por último, tem-se mais duas coleções de aquivos importantes para o PowerBoard. Um delas é representada pela caixa `summary_v2.py`, a qual adiciona, dentro das regras de sua arquitetura, uma forma customizada do script do modelo (representado pela caixa `script.py`) se comunicar com o plugin, assim é possivel utilizar as funções dentro desse arquivo para obtenção de dados para alimentar o plugin.

A outra coleção, representada pela caixa `setup.py` representa os arquivos necessários para a instalação do PowerBoard no ambiente do TensorBoard.

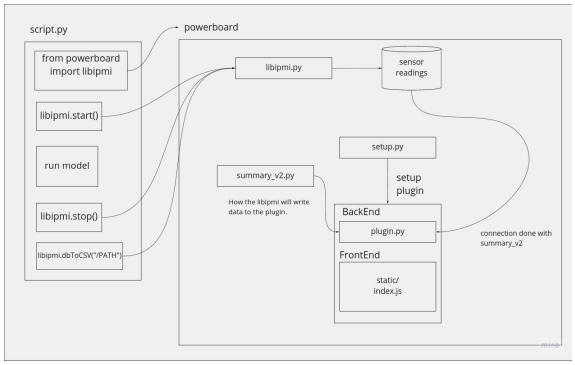

Figura 1. Representação da arquitetura do plugin

3. Discussão

O estado atual do PowerBoard fornece um gráfico com dados próprios passível de comparação com o Profile disponível para o TensorBoard, como representado pela Figura 2, a qual mostra a dissipação de potência por tempo. O programa pode ser instalado usando o sistema de pacote pip[9] do Python através do comando *pip install powerboard*.

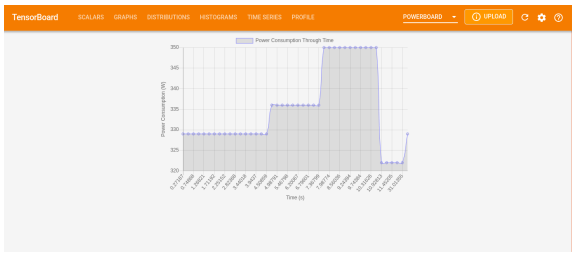

Figura 2. Representação do estado do plugin

Podemos utilizar o plugin com auxilio da ferramenta do trace-viewer, que está contido no profile do TensorBoard [8]. O trace-viewer é uma ferramenta que mostra uma linha do tempo dos diferentes eventos que ocorreram na CPU e na GPU durante o período de treinamento da rede neural. Assim podemos fazer algumas análises, por

exemplo na figura 2 no intervalo entre $(7.36, 7.98)$ segundos temos um pico de potência, agora podemos ir no trace-viewer e verificar quais são as operações que são responsáveis pelo pico de potência. Outra análise possivel é verificar o que acontece de uma *epoch* para outra, durante o treinamento do seu modelo. Esta parte está melhor documentada na seguinte referência [7].

Tendo em vista que este é um projeto em andamento, um próximo passo para a melhoria do PowerBoard seria integrar também potência consumida exclusivamente pela GPU, além de utilizar diferentes bibliotecas que posssam informar dados de sensores de energia e potência, tal como a *PAPI*[4], por exemplo, ou até mesmo a disponibilização de um gráfico de consumo de energia, o que já pode ser feito pelo usuário, uma vez que ele determina o diretório a ser salvo o CSV com os dados coletados.

Referências

[1] ESModule. Standardized module system to javascript. https://hacks.mozilla.org/2018/03/es-modules-a-cartoon-deep-dive/.

[2] IPMI. Intelligent platform management interface specification v2.0 rev. 1.1. https://www.intel.com/content/www/us/en/products/docs/servers/ipmi/ipmi-second-gen-interface-spec-v2-rev1-1.html.

[3] ipmitool. ipmitool is a utility for managing and configuring devices that support the intelligent platform management interface. https://github.com/ipmitool/ipmitool.

[4] Performance Application Programming Interface. Papi provides the tool designer and application engineer with a consistent interface and methodology for use of the performance counter hardware found in most major microprocessors. https://icl.utk.edu/papi/.

[5] POWER9. Ibm processor architcture. https://www.ibm.com/br-pt/it-infrastructure/power/power9.

[6] PowerBoard. Power profiling plugin project for tensorboard. https://github.com/Unicamp-OpenPower/PowerBoard.

[7] PowerBoardPlugin. Powerboard plugin for tensorboard. https://openpower.ic.unicamp.br/post/powerboard-plugin-for-tensorboard/.

[8] Profiler. Tensorflow profiler to profile the execution of tensorflow code. https://www.tensorflow.org/tensorboard/tensorboard_profiling_keras.

[9] PyPi. The python package index. https://pypi.org/.

[10] TensorBoard. Tensorboard: kit de ferramentas de visualização do tensorflow. https://www.tensorflow.org/tensorboard?hl=pt-br.

[11] TensorBoard Plugin. Custom tensorboard visualization. https://github.com/tensorflow/tensorboard/blob/javascript/ADDING_A_PLUGIN.md.

[12] TensorFlow. Tensorflow is an end-to-end open source platform for machine learning. https://www.tensorflow.org/.