SyNERGY: An energy measurement and prediction framework for Convolutional Neural Networks on Jetson TX1

Crefeda Faviola Rodrigues, Graham Riley, and Mikel Luján
School of Computer Science, The University of Manchester, Manchester, United Kingdom

Abstract—There is a huge demand for on-device execution of deep learning algorithms on mobile and embedded platforms. These devices present constraints on the application due to limited hardware resources and power. However, current evaluation studies in existing deep learning frameworks (for example, Caffe, Tensorflow, Torch and others) are limited to performance measurements of these applications on high-end CPUs and GPUs. In this work, we propose "SyNERGY" a fine-grained energy measurement (that is, at specific layers) and prediction framework for deep neural networks on embedded platforms. We integrate ARM's Streamline Performance Analyzer with standard deep learning frameworks such as Caffe and CuDNNv5 to quantify the energy-use of deep convolutional neural networks on the Nvidia Jetson Tegra X1. Our measurement framework provides an accurate breakdown of actual energy consumption and performance across all layers in the neural network while our prediction framework models the energy-use in terms of target-specific performance counters such as SIMD and bus accesses and application specific parameters such as Multiply and Accumulate (MAC) counts. Our experimental results using 9 representative Deep Convolutional Neural Network shows that a multi-variable linear regression model based on hardware performance counters alone achieves an average prediction test error of 8.04 ± 5.96% compared to actual energy measurements. Surprisingly, we find that it is possible to refine the model to predict the number of SIMD instructions and main memory accesses solely from the application's Multiply-Accumulate (MAC) counts with an average prediction test error of 0.81 ± 0.77% and 17.97 ± 15.29% respectively. This alleviates the need for actual measurements giving a final average prediction test error of 7.08 ± 5.05% using solely the application's MAC counts as input.

Keywords: Energy Measurement, Energy prediction, Convolutional Neural Networks, mobile computing

1. Introduction

The aim of contemporary and future computing systems is to deliver higher performance at lower power budgets [1]. This includes embedded systems or "edge-devices" that add further limits to energy-usage due to limited battery life. Measurement of power consumption is an understudied especially when designing software for deep learning algorithms. These algorithms have become core components of applications such as key-word spotting, facial recognition, language translation and others [2]. Specifically, Convolutional Neural Networks (hereafter referred to as ConvNets) have achieved state-of-the-art results in various vision domains and natural language processing domains [2]. To enable such applications for embedded devices there are numerous efforts to optimize these algorithms in terms of computation and memory resources. However, there are very few studies that quantify the actual energy-use for deep learning workloads [3], [4], [5]. These optimization efforts are spread across various levels. At the algorithmic level, newer compact neural network designs [6], [7], compression and pruning techniques [8], [9] or reduced precision [2] are proposed to save memory and increase throughput. At the software level, device-specific software implementations such as TensorRT, ARM Compute library, Qualcomm’s Snapdragon Neural Processing Engine (NPE) [1] and others aim to accelerate deep learning inferences or deployment on existing mobile platforms. These libraries are complementary to existing deep learning frameworks such as Caffe2 [10], Tensorflow, Torch and others in which deep learning models must first be designed and trained. At the hardware level, application-specific hardware have emerged such as specialised GPUs (e.g. Jetson Tegra TX1 and TX2), FPGAs and ASICs [1].

We consolidate our observations from these works and attribute the lack of adoption of energy-use as an evaluation criteria to the following reasons:

- Lack of energy-measurement support in existing deep learning frameworks: currently, popular frameworks such as Caffe, Torch, Tensorflow and others provide designers with the tools to benchmark their application's performance through timing measurements. There is no support for energy measurements as these are challenging to obtain consistently across platforms. For example, power measurement facilities can vary from system-to-system. This includes different types of power meters [5] and power sensors [3].
- Accuracy as a key metric to evaluate models: the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [11] has been a major test-bed for the development of innovative ConvNet models. However, a given published accuracy is often achieved by averaging the accuracy from an ensemble of models that are
executed on desktop or server systems [5]. This implies that more computational resources are used to achieve the desired accuracy. However, on embedded platforms or specialised hardware, resource-budgets are a major concern, leading to pruned versions of existing models or smaller models being chosen for deployment [7].

In this paper, we propose an energy measurement and prediction framework called SyNERGY, by integrating Caffe, a deep learning framework, and vendor-specific tools such as ARM Streamline Performance Analyzer [12]. Shown in Figure 1, is the set up for measuring energy and performance (or execution time) of ConvNet models. Our methodology focuses on power measurements made using the on-board power monitoring sensor (TI-INA3221x available on the Jetson TX1) for single image inferences on the CPU using an optimised OpenBLAS library. Our contribution include the following:

- To the best of our knowledge, this is the first energy prediction framework that models the energy-use per layer of a neural network to actual measurements from the Jetson TX1 hardware performance counters such as SIMD instruction executed and bus accesses (or main memory accesses), with an average prediction error rate of 8.04 ± 5.96% over actual energy measurements.
- Based on experimental data gathered, SyNERGY, further develops a regression model to predict the energy-use of the convolutional layers solely from the application’s MAC count, with an average prediction test error of 7.08 ± 5.05% over actual energy measurements. This is achieved by developing models to predict the individual performance counters such as SIMD and bus accesses with an average prediction test error rate of 8.1 ± 0.77% and 17.97 ± 15.29% respectively.
- Our energy and performance measurement framework provides detailed breakdown of the energy consumed in every layer of the whole neural network model, thereby, enabling us to identify the energy bottlenecks in the model.

The organisation of the paper is as follows. Section 2, is a general background into convolutional neural networks. Section 3, provides details of the energy measurement framework and per-layer energy breakdown of models. Section 4, covers the prediction framework and the experimental results. Section 5, details the related work in energy measurement and analysis. Finally, section 6 concludes and highlights possible future directions.

2. Primer on Convolutional Neural Networks

To provide computers with ability to perform intelligent tasks such as understanding images and audio, learning, and others, the field of Machine Learning focuses on developing mathematical models or algorithms to acquire knowledge by extracting information from raw data. A Convolutional Neural Network (or ConvNet) extracts information or features such as edges, color blobs through a process called feature extraction from images, and uses this information to provide a classification output (or a decision). It is composed of layers to transform the raw input data into a meaningful probabilistic output. Figure 2, shows the data dimensions involved in a convolution (or conv) operation. Other typical layers found are pooling (pool), batch norm (norm), Rectified linear unit (ReLU) and fully-connected (fc), that lend the model certain properties1. Each layer in a model has a certain number of computation known as computational complexity or storage cost or bandwidth, and each of these have implications in terms of energy-use [2].

The computational complexity for a conv layer is estimated from the number of Multiply-accumulates (MAC), and its storage cost is the sum of the sizes of the input feature map \((I_x \times I_y \times I_z)\) to each layer, the corresponding filter weights \((K_x \times K_y \times I_z \times O_z)\) and biases and the output feature map \((O_x \times O_y \times O_z)\). Multiple input images can be grouped into batches called batch size and is denoted as \(N\). Here, \(x, y\) and \(z\) represent the Cartesian axes. The computational complexity of a standard convolution operation is given by:

\[
O_x \times O_y \times O_z \times K_x \times K_y \times I_z
\]

(1)

1This has not been explained for the sake of simplicity and the reader is advised to refer to [2] for more information.
and the storage cost or bandwidth (in counts) is given by:

\[
(I_x \times I_y \times I_z + K_x \times K_y \times I_z \times O_z + O_x \times O_y \times O_z)
\]  

(2)

Table 1 gives a list of the models chosen for this study, where Column 5 represents the counts of two types of layers: conv and fc present in each model. Recently, in newer ConvNet models, for example, Squeezenet [7], the global average pooling layer has replaced traditional fully-connected layers [6]. Column 7 shows the size of the model which is stored in 32-bit floating point precision and is four times its number of parameters. The top-5 accuracy of the model is the top-5 predictions of the object category in a given image from the ImageNet dataset or CIFAR 10 and is a measure of how well the model performs for the task of image classification [11].

In our study, we target the conv layers as these are known to consume 85-90% of the computational time [1].

### 3. Energy and performance measurement framework

The embedded system chosen for the power and performance measurements is the Jetson TX1 which has a quad-core of ARM Cortex A57/A53 running at 1.9GHz and 256 CUDA core Maxwell GPU running at 1GHz. The TX1 is cross-compiled with a modified Linux kernel version (3.10.96+) using Jetpack 2.3 and has a host operating system of Ubuntu 16.04. The modification enables the on-board power monitor sensor TI-INA3221x and integrates the gator module required by ARM Streamline tool. This power monitor provides system level power, CPU level power and GPU level power. This is the post-regulation power after AC to DC conversion of the wall power and DC to DC conversion (pre-regulation power conversion) as required by the system-on-chip (SoC). Power values (mW) are instantaneous and are accessible from the hardware counters on the /sys file system (/sysfs). We use the system level power to the SoC as this accounts for the power due to the processing core, DRAM memory and peripherals. The power measurements are gathered with the default interactive Linux governor.

The measurement framework is composed of two components: the hardware component that includes the target device and the power monitor that provides direct power values, and the software component as shown in Figure 1 that implements the methodology to acquire accurate and consistent power and performance measurements.

To carry out an inference, we chose the Caffe framework, a well-known and widely used deep learning framework. Our Caffe (version 1.0.0-rc3) was compiled for the GPU with Cuda (8.0) and CuDNN (5.1.5) and CPU with Open- BLAS (libopenblas_cortexa57p-r.0.2.20.dev.a) with a max num-thread of 4. The application selected was an inference using a ConvNet model on a single RGB image from ImageNet and CIFAR 10 dataset [11]. The size of the input image is given in the model specification of each ConvNet. All computations are 32-bit floating point. The pre-trained weight file for the selected ConvNet models, given in Table 1, is available in Caffe’s model-zoo repository.

To map per layer functions in Caffe, we use Streamline’s annotation library to demarcate the beginning and end of a layer’s execution. The annotations are collected by the gator and fed to Streamline tool, which is running on a host machine. The sampling frequency was selected to be 1kHz. The power sampler script is created to collect the power values, and further processing is required to obtain the energy as per Equation 3.

\[
E_{\text{inference}} = \sum_{i=0}^{T} E_{t+1} = \sum_{i=0}^{T} P_{t+1} \times dt
\]

(3)

Here, \(E_{t+1}\) and is calculated using the \(i + 1^{th}\) power sample \(P_{t+1}\) over that duration \(dt = t_{i+1} - t_i\). The total energy for the inference \(E_{\text{inference}}\) is the sum of all the rectangular areas over the duration of the inference \(T\). In our study, we report the execution time or performance per image (sec/image) and energy per image (mJ/image) averaged over 5 separate runs for single image inferences.
can be built by extracting the execution profile of the layers memory. Therefore, a simple multi-variable regression model consists of a memory hierarchy with L1, L2 caches and main granularity on the target platform. The Jetson TX1 system training set to capture the execution behaviour at a per-layer level by first running a set of models in the prediction framework focuses on modelling the energy at machine learning to build a regression model [18].

4. Energy prediction framework

4.1 Energy prediction model from measured SIMD and bus accesses

Initially, the models from Table 1 were qualitatively selected to form a training set on the following basis: AlexNet and VggNet-small represent structurally similar models that have fewer layers but relatively large number of parameters (62M-102M). SqueezeNet and SqueezeNetRes are an important class of ConvNets that are trained to keep the model size low without compromising accuracy (1.2M parameters, ∼80% top-5 accuracy). GoogleNet has the best trade-off of size and accuracy (6.9M, 90.85%) and ResNet50 represent the current state-of-the-art model in terms of accuracy in image classification tasks (25M, 93.2%).

To evaluate the robustness of the regression-based prediction model, we train this subset of ConvNet models with SIMD and bus access as the input. The set of prediction models were obtained by excluding a single ConvNet during the training phase, known as Leave one out cross validation [18]. Here, each model given in Column 1, indicates the model excluded during training. The first row represents a single experiment to form a regression model by excluding AlexNet as a data point during training. Column 2, represents the coefficient for the total bus accesses ($x_2$) and Column 3 represents the coefficient for the total SIMD counts ($x_1$) for all the conv layers in a given ConvNet. We then used the coefficients derived to predict the energy consumed for the conv layers and compare it against actual energy measurements.

The relative error, given by Equation 5, quantifies the performance of the predictor with respect to the baseline based on the number of SIMD instructions executed and the number of bus accesses (equivalent to last-level cache misses), on the target platform. While sophisticated prediction models can be developed based on other performance counters such as L1 cache misses and others, it is well known that the amount of computation and main memory accesses are the most expensive, in terms of energy [19].

The training data helps to establish a relationship, if any, between energy consumption and hardware-specific performance counters. Once this prediction model is built, it can be tested on example ConvNet models not seen during the training phase. The energy values and performance counter data thus obtained is fed into a learning algorithm to learn the regression coefficients for SIMD $x_1$ and bus accesses $x_2$. This captures any relationship between application and hardware, as given in Equation 4.

$$E_{conv}^\hat{} = x_1 \times bus\_accesses_{conv} + x_2 \times SIMD_{conv}$$  (4)
measured energy value.

\[
Rel.\text{Err}(% \, \text{or} \, \%) = \left( \frac{|\text{predicted}_\text{value} - \text{actual}_\text{value}|}{\text{actual}_\text{value}} \right) \times 100
\]

(5)

For example, in the first row, the average relative training error for the all ConvNets (which does not include AlexNet) in the training set is 5.36 ± 3.36% and its relative test error on AlexNet is 2.23%. Figure 5, shows the average bus access and SIMD counts over 5 runs which are used as inputs into the model. The corresponding average measured energy and average measured time is given in Column 5 and Column 6 of Table 2.

Finally, we consider the average relative training error (that is, 4.81 ± 3.19%) by including all the ConvNets (or allNets) in the training set. This is lower than by excluding individual models which implies that the given subset of models sufficiently captures the execution behaviour of the conv layers on the CPU of the target system. *Given the scenario where we can measure performance counters such as SIMD instruction and bus accesses, we are able to predict the energy consumption of unseen test ConvNets with an average relative test error of approximately ∼ 8% compared to actual energy measurements.*

Finally, to alleviate this need of having to measure SIMD and bus accesses counts, we attempted to explore the possibility of building prediction models for the two dependent variables *bus accesses* and *SIMD* themselves. This data was then fed into our current prediction model based on *allNets* to obtain a final estimate of energy consumption of any given ConvNet on the CPU of the Jetson TX1 platform, as discussed in the next subsections.

### 4.2 Predicting conv layer SIMD counts

SIMD operations exploit the data parallelism in matrix-matrix multiplication to obtain higher efficiency. Since the computation in a conv layer can be transformed to a matrix-matrix multiplication operation, we explore the relationship between the application MAC count and its measured SIMD instructions for every conv layer. We use Equation 1 in Section 2, to determine the MAC count for a conv layer. The total MAC for all the conv layers in a given ConvNet, is tabulated in Column 3 of Table 6. This total MAC count can then be used to build a simple linear regression model, as given in Equation 6, to predict the SIMD counts for all the conv layers in a ConvNet. The data corresponding to the measured SIMD \((y)\) and predicted SIMD \((\hat{y})\) counts are shown in Column 2 and 4 of Table 3.

We obtain a slope of 0.24, as show in Figure 6, which confirms that the SIMD width is 4 for the ARM A57 cores on the Jetson TX1. Therefore, given an appropriate calculation of MAC count from the application, we can build a SIMD predictor that obtains an average relative test error of 0.81 ±0.77% compared to actual SIMD measurements.

### 4.3 Predicting conv layer bus accesses

Conv layers are often preceded and succeeded by data transformation operations such as im2col to transform it into a matrix-matrix computation, and col2im to transform it back into the original 2D image layout [1]. This is because these conv layers are interleaved with pool, ReLU and other layers which required data in a specific 2D format. Even though, we can calculate the bandwidth of each conv layer as given in Equation 2, the data re-structuring and associated data movement between layers within the complex cache memory hierarchy make the relationship between bandwidth and performance counters such as cache and bus accesses non-trivial.

Surprisingly, we found that a linear relationship exists
between the total number of measured bus accesses and SIMD counts in the conv layers, which can be seen in Figure 5. Therefore, a similar linear regression predictor, as given in Equation 6, was built to determine the bus access counts for the conv layers from the measured SIMD counts. We find a linear relationship exists with a coefficient of 0.0663 between SIMD and bus access counts. We can now use the predicted SIMD \( \hat{y} \) counts obtained previously, to predict the bus access counts \( z \) for all the ConvNets. The predicted bus access \( \hat{z} \) is given in Column 4 of Table 4 and is thus obtained without the need for actual measurements of bus accesses.

For most ConvNets we are able to obtain a good prediction of bus access counts from predicted SIMD with an average relative test error of 17.97 ± 15.29%.

### 4.4 Energy prediction model from MAC count

Our final step, is to estimate the energy consumption of all the conv layers in a ConvNet directly from the application parameters, which in our case is the conv layer MAC count. The prediction for the SIMD \( \hat{y} \) and bus access counts \( \hat{z} \) for the conv layers can now be fed into our final energy prediction model obtained from the \textit{allNets} training set. This enables us to make prediction using regression models without having to execute and measure the energy of a new ConvNet. This data is tabulated in Table 5, where the predicted energy \( \hat{E} \) is given in Column 2. Therefore, we are able to predict the energy consumption of the conv layers of any given ConvNet, solely using MAC count, with an average relative error test rate of 7.08 ± 5.05%.

### 5. Related Work

**Performance measurement and analysis:** Several studies such as \textit{Fathom} [20], ConvNet-Benchmarks [21], GPU performance [22], characterize the behaviour of deep learning models based on execution time. While these performance studies explore the space of desktop and server based CPUs and GPUs, they offer limited insight on the execution behaviour of deep learning models on resource constrained embedded platforms.

**Energy measurement and analysis:** Few studies have emerged that report energy and performance of deep learning models on the TX1 platform [9], [5]. However, these studies are often adhoc and with a limited set of deep learning models with the aim of platform-to-platform comparisons. These studies lack in a consistent methodology to acquire power for the Jetson TX1 and provide minimum details of the adopted method. Our work instead develops a detailed methodology to acquire power measurements using the power sensor onboard the TX1.

\textit{BenchIP} [23], aims at evaluating the efficiency deep learning workloads comprising ConvNets and Long Short Term Memory networks (LSTM) models on representative platforms from desktop, server and embedded domains. The authors evaluate each layer in isolation and overall end-to-end model executions in Caffe. However, it differs from our approach, where we study each layer in isolation in the context of an actual inference. To the best of our knowledge, this benchmark suite is yet to be open-sourced.

Finally, our work shares similarity to an energy profiling methodology adopted by [4]. The authors provide the energy consumption at a functional level through the use of code-annotations to demarcate specific phases of a decision-tree application. However, it is unclear on how transferable the approach would be as the tool is not open-sourced. Our work instead builds an energy measurement and prediction framework on top of existing open-sourced deep learning
software framework such as Caffe that is widely adopted by the machine learning community.

**Energy estimation methodologies:** While benchmarking efforts are continuing to grow, it is a time-consuming effort. Therefore, energy estimation tools have been proposed to evaluate deep learning models [9]. Here, the authors propose a model of energy consumption based on the number of MACs and bandwidth, and associates an energy for each operation to estimate the energy consumption of a neural net on a specialised ASIC. However, such an estimation methodology may not be transferable to models executing on general-purpose hardware. Our work focuses on building a model of energy consumption based on the actual execution behaviour for the CPU on the target platform and deriving a model to estimate the energy of a layer solely on its MAC count.

### 6. Conclusions and Future Work

Deployment of deep learning applications on mobile and embedded platforms remains a challenge due to limited power budgets available on such devices. Efforts to improve energy consumption of deep learning applications have begun to emerge from the development of compact ConvNet models to building specialised hardware.

We propose a framework to enable fine-grained performance and energy measurement of deep learning applications targeting embedded platforms such as the Jetson TX1. We demonstrate a systematic methodology by integrating a widely used deep learning framework such as Caffe and ARM’s Streamline Performance Analyst. With the help of our energy measurement framework, we were able to build an initial energy prediction model using 9 representative ConvNet models.

Our initial energy prediction model was based on hardware performance counters such as SIMD and bus access for the CPU obtained from actual execution runs of these models. Using this approach, we are able to predict the energy-use of all the conv layers in a ConvNet model with an average relative test error rate of 8.04 ± 5.96% over actual energy measurements. Furthermore, we refine this model to make predictions directly from the application parameters. This is achieved by developing predictors for each of the hardware performance counters such as SIMD and bus access. We obtain a prediction error of 0.81 ± 0.77% and 17.97 ± 15.29% for each respectively. Using this as input, our final energy predictor achieves 7.08 ± 5.05% average relative test-error by using solely Multiply-Accumulate (MAC) counts obtained from the application description.

Future work, includes extending this energy prediction model to target other performance counters by including L1 and L2 cache access and other hardware platforms. Opportunities for in-depth analysis work to guide the use of power management techniques such as DVFS to reduce energy consumption at specific layers could also be explored.

### Acknowledgment

This research was conducted with support for C. Rodrigues and G.D. Riley from the IS-ENES2 project, funded under the European FP7-INFRASTRUCTURES-2012-1 call (GA No: 312979). C. Rodrigues is also part-funded by ARM Ltd. under a PhD Studentship Agreement. Mikel Luján is supported by a Royal Society University Research Fellowship. The author would also like to acknowledge the valuable insights and guidance of the colleagues from the Advanced Processors Technology group at The University of Manchester and ARM Ltd. research group, Cambridge.

### References


