Designing Neural Network Hardware Accelerators with Decoupled Objective Evaluations

José Miguel Hernández-Lobato
University of Cambridge

Michael A. Gelbart
University of British Columbia

Brandon Reagen
Harvard University

Robert Adolf
Harvard University

Daniel Hernandez-Lobato
Universidad Autónoma de Madrid

Paul N. Whatmough
Harvard University

David Brooks
Harvard University

Gu-Yeon Wei
Harvard University

Ryan P. Adams
Twitter & Harvard University

Abstract

Software-based implementations of deep neural network predictions consume large amounts of energy, limiting their deployment in power-constrained environments. Hardware acceleration is a promising alternative. However, it is challenging to efficiently design accelerators that have both low prediction error and low energy consumption. Bayesian optimization can be used to accelerate the design problem. However, most of the existing techniques collect data in a *coupled* way by always evaluating the two objectives (energy and error) jointly at the same input, which is inefficient. Instead, in this work we consider a *decoupled* approach in which, at each iteration, we choose which objective to evaluate next and at which input. We show that considering decoupled evaluations produces better solutions when computational resources are limited. Our results also indicate that evaluating the prediction error is more important than evaluating the energy consumption.

1 Introduction

Making predictions with Deep Neural Networks (DNN) is very expensive in battery-powered devices with limited computational capabilities. Therefore, in these systems it is desirable to speed up DNN predictions. A promising solution for this problem is to use hardware accelerators [10].

When designing accelerators for DNN predictions it is important to achieve an optimal trade-off between objectives. Here we focus on the minimization of the energy consumption and the prediction error. This is a multi-objective optimization problem with expensive black-box functions. Evaluating the prediction error is a costly black-box process because we have to train a DNN on large amounts of data to finally measure the DNN's predictive accuracy on some validation data. Similarly, evaluating the energy consumption is an expensive black-box process because it involves simulating in software the operations performed by the hardware accelerator during prediction [10].

Bayesian optimization (BO) methods can be used to accelerate optimization problems with expensive black-box functions [8]. These methods can also efficiently solve multi-objective optimization problems [9, 5]. However, most of the existing multi-objective BO methods assume that the different objectives will always be evaluated in a *coupled* way, that is, one objective after the other and at the same input location. By contrast, in the case of designing DNN hardware accelerators, the two objectives (energy and error) can be evaluated in a *decoupled* way, that is, at each step we can choose whether to collect data from one objective or the other [3]. Decoupled evaluations are possible because the energy consumption is independent of the value of the DNN weights. This means that

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

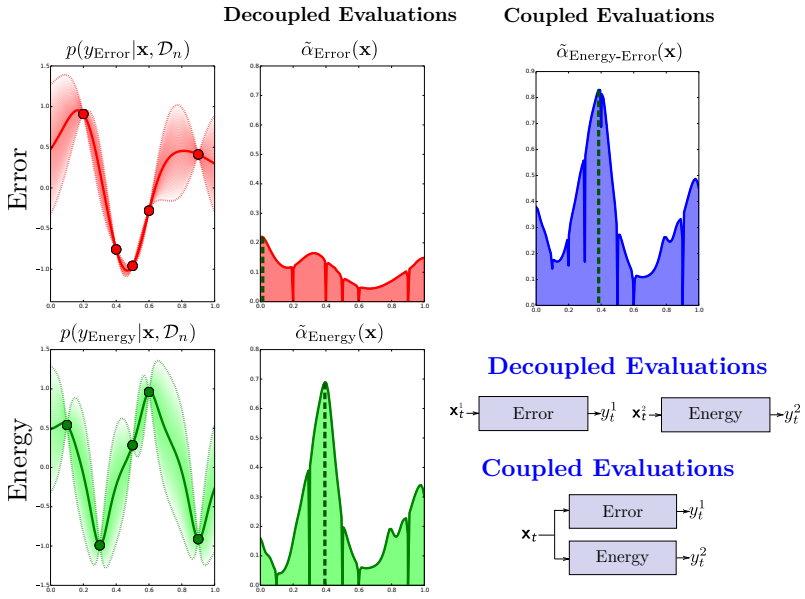

Figure 1: Illustration of the differences between coupled and decoupled evaluations.

we do not have to train the DNN to evaluate the energy consumption since the DNN weights can be assigned random values when we evaluate this objective. The differences between coupled and decoupled evaluations are illustrated in the bottom-right plot in Figure 1.

Decoupled evaluations are more flexible than coupled ones, since the latter are a particular case of the former. Therefore, we expect the decoupled approach to perform better, especially with limited computational resources. For example, when there is only a single computer node available for function evaluations. In this case, we can use the data collected after each objective evaluation to make better decisions about which objective to evaluate next. In this work, we consider this single-computer-node scenario and analyze the performance of the decoupled approach with respect to other coupled baselines. We show that considering decoupled evaluations produces better solutions and also that evaluating the prediction error is more important than evaluating the energy consumption.

2 Decoupled multi-objective Bayesian optimization

In general, BO methods decide where to collect data by maximizing an acquisition function $\alpha(\mathbf{x})$ with respect to the design parameters $\mathbf{x}$, with $\alpha(\mathbf{x})$ returning the expected utility obtained by evaluating the objective at $\mathbf{x}$ [9]. The expectation in α is computed with respect to the predictions of a probabilistic model trained on the available data, typically a Gaussian process (GP) model.

We now focus on the problem of designing DNN accelerators. In the coupled evaluation case, there is only a single acquisition function $\alpha_{\text{Energy-Error}}(\mathbf{x})$, which is optimized to determine where to evaluate the two objectives (energy consumption and prediction error) next. By contrast, in the decoupled case, there is a different acquisition function for each objective, that is, $\alpha_{\text{Energy}}(\mathbf{x})$ and $\alpha_{\text{Error}}(\mathbf{x})$ [6]. At each iteration, these two acquisition functions are globally optimized and the one achieving the highest value determines which objective function will be evaluated next and at which location.

The differences between the coupled and the decoupled approach are illustrated in a toy example in Figure 1. The plots in the left part of this figure show the available data for the energy and error objectives, along with the corresponding GP predictive distributions. In the decoupled case, the GP predictions are mapped into the acquisition functions $\alpha_{\text{Energy}}(\mathbf{x})$ and $\alpha_{\text{Error}}(\mathbf{x})$, shown in the plots in the middle of Figure 1. Each of these functions is maximized to determine which objective evaluation produces the highest utility on average and at which location. In this case, evaluating the energy consumption at $x = 0.4$ produces the highest expected utility. Therefore, with only a single computer node available, we would then evaluate next the energy objective at $x = 0.4$. After collecting the data, the GP predictive distributions are updated and the whole process repeats. In the coupled case,

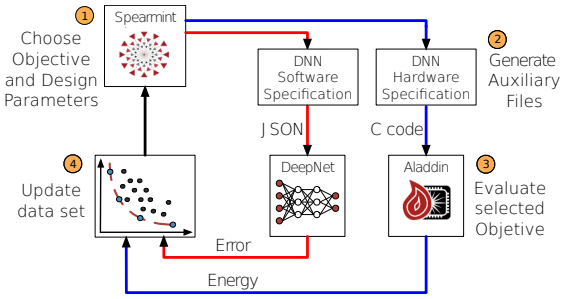

Figure 2: Decoupled evaluations flow.

Parameter	Min	Max	Step	Type
Neurons per layer	50	250	1	Er/En
Learning rate	10^{-3}	1	ϵ	Er
Dropout rate	0	0.4	ϵ	Er
L2 penalty	0	0.1	ϵ	Er
Memory bandwidth	1	32	2^x	En
Loop parallelism	1	32	2^x	En
Total number of bits	1	32	4	Er/En
Fraction integer bits	0	1	ϵ	Er/En

Table 1: Design parameters to be optimized.

the GP predictions are mapped into a single acquisition function $\alpha_{\text{Energy-Error}}(\mathbf{x})$, shown in the plot in the top-right of Figure 1. This acquisition function is then maximized to determine where to collect data next for both objectives. In this case, evaluating both objectives at $x = 0.4$ produces the highest expected utility. After collecting the data, the GP predictions are updated and the process repeats.

2.1 Predictive Entropy Search for Multi-objective BO

Predictive Entropy Search for Multi-objective (PESMO) BO is a state-of-the-art BO method for multi-objective optimization problems. PESMO often outperforms other baselines in the coupled evaluation scenario [5]. Importantly, PESMO is the only existing method that allows to perform decoupled evaluations with multiple objectives. The source code for PESMO is publicly available within the BO tool Spearmint[1] PESMO's acquisition function approximates the expected gain of information on the solution to the optimization problem [13, 4, 7], where information is measured in terms of the entropy of the corresponding random variable. The solution to the multi-objective optimization problem is the Pareto set, a collection of optimal values for the design parameters. The Pareto set is optimal because from each point in that set one cannot improve one of the objectives without deteriorating the other. PESMO's acquisition functions for the decoupled and coupled scenario are

$$\alpha_{\text{Error-Energy}}(\mathbf{x}) \approx \mathrm{H}(\mathcal{X}^\star|\mathcal{D}) - \mathbb{E}_{\mathbf{y}|\mathcal{D}}\left[\mathrm{H}(\mathcal{X}^\star|\mathcal{D} \cup \{(\mathbf{x}, \mathbf{y})\})\right] , \tag{1}$$

$$\alpha_{\text{Error}}(\mathbf{x}) \approx \mathrm{H}(\mathcal{X}^\star|\mathcal{D}) - \mathbb{E}_{y_{\text{Error}}|\mathcal{D}}\left[\mathrm{H}(\mathcal{X}^\star|\mathcal{D} \cup \{(\mathbf{x}, y_{\text{Error}})\})\right] , \tag{2}$$

$$\alpha_{\text{Energy}}(\mathbf{x}) \approx \mathrm{H}(\mathcal{X}^\star|\mathcal{D}) - \mathbb{E}_{y_{\text{Energy}}|\mathcal{D}}\left[\mathrm{H}(\mathbf{x}^\star|\mathcal{D} \cup \{(\mathbf{x}, y_{\text{Energy}})\})\right] , \tag{3}$$

where $\mathcal{X}^\star$ represents the Pareto set, $\mathcal{D}$ is the data collected so far, $\mathbf{y}$, y_{Energy} and y_{Error} are respectively the data collected for the two objectives, the energy objective and the error objective for the value of the design parameters $\mathbf{x}$ and finally, $\mathrm{H}(\cdot)$ computes the entropy of a random variable.

3 Experiments

We compared PESMO using decoupled evaluations (decoupled BO) with three other baselines: a version of PESMO that always performs coupled evaluations (coupled BO), an evolutionary strategy based on the NSGA-II algorithm (NSGA-II) [2] and finally, a random search approach (Random). We consider the problem of designing a hardware accelerator for a DNN with 3 hidden layers that makes predictions on the MNIST data set.

Table 1 shows the design parameters that we consider for tuning. Each of them has an effect on the energy objective (En), the error objective (Er) or both. The last two parameters in the table indicate the numerical precision used to encode the DNN weights and the data in the accelerator. To train the DNN, we used the DeepNet[2] python library. The energy consumption was estimated using the Aladdin simulator [11].

The accelerator design flow is shown Figure 2. Each pass through this 4-step process produces one data point by evaluating one of the objectives. First, Spearmint selects one objective function

[1] `https://github.com/HIPS/Spearmint`

[2] `https://github.com/nitishsrivastava/deepnet`

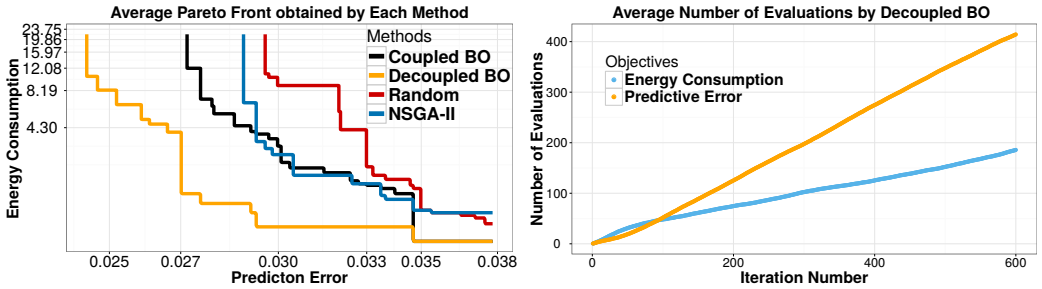

Figure 3: Left, average Pareto front obtained by each method. Right, average number of decoupled evaluations performed by Decoupled BO for each objective

	Random	NSGA-II	BO Coupled	BO Decoupled
Hypervolume	258.90±0.55	259.55±0.57	262.73±0.45	**264.96±0.31**

Table 2: Avg. Hypervolume and corresponding standard errors obtained by each method.

and a value for the design parameters by optimizing (2) and (3). Second, the selected value for the design parameters is translated to a representation that the evaluation tools can interpret. DeepNet uses a declarative JSON file for training DNNs. Aladdin requires a file with C code describing the operations performed by the DNN during prediction. Third, DeepNet or Aladdin are executed to obtain estimates of either the prediction error or the energy consumption. When running DeepNet, the DNN weights obtained during training are then rounded to the numerical precision determined by the last two parameters from Table 1. During last step of the design flow, the resulting data is fed back into Spearmint and then the GP predictions and acquisition functions (2) and (3) are recomputed.

The previous process is repeated until 600 function evaluations have been completed. Then each method gives a recommendation in the form of a Pareto set obtained by optimizing the posterior means of GPs fitted to the collected data [1]. When running all the analyzed methods, we discretize the search space by considering only 10,000 points which are selected by using a Halton sequence.

	Coupled BO	Decoupled BO
Avg. Time	554,098 s.	**374,369 s.**

Table 3: Average time spent by each BO method evaluating the objective functions.

3.1 Results

We repeated the optimization problem 20 times and then report average results. The left plot in Figure 3 shows the average Pareto front obtained by each method. The decoupled BO approach is the best performing technique, producing accelerators with much better trade-offs between prediction error and energy consumption. The coupled BO approach performs slightly better than the evolutionary strategy and the random approach is the worse performing one. Table 2 reports the average hypervolume [14] of the Pareto fronts produced by each method. Again, the ranking of the different methods is Decoupled BO, Coupled BO, NSGA-II and Random.

To better understand the superior performance of the decoupled approach, we report in the right plot of Figure 3 the average number of evaluations of each objective by Decoupled BO. This plot clearly shows that Decoupled BO evaluates more often the prediction error objective, which seems to be more difficult to learn than the energy consumption. Interestingly, the energy consumption is also the objective that is most expensive to evaluate. Therefore, besides finding better solutions, Decoupled BO also spends less time performing function evaluations. This is illustrated by the results from Table 3, which shows the average time spent by each BO method evaluating the objective functions. Note that in our experiments Decoupled BO assumes that each objective function takes the same amount of time to be evaluated. Nevertheless, it is straightforward to account for evaluation time by scaling the acquisition functions (2) and (3) by an estimate of the time required to perform the corresponding function evaluations [12].

Acknowledgments

JMHL acknowledges support from the Rafael del Pino Foundation. DHL acknowledges financial support from the Spanish Plan Nacional I+D+i, Grants TIN2013-42351-P and TIN2015-70308-REDT, and from Comunidad de Madrid, Grant S2013/ICE-2845 CASI-CAM-CM This work was partially supported by C-FAR, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA. The work was also supported in part by DARPA under Contract #: HR0011-13-C-0022. This research was, in part, funded by the U.S. Government. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

References

[1] Roberto Calandra, Jan Peters, and MP Deisenrothy. Pareto front modeling for sensitivity analysis in multi-objective bayesian optimization. In *NIPS Workshop on Bayesian Optimization*, volume 5, 2014.

[2] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[3] Michael A. Gelbart, Jasper Snoek, and Ryan P. Adams. Bayesian optimization with unknown constraints. In *UAI*, pages 250–259, 2014.

[4] Philipp Hennig and Christian J Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13(Jun):1809–1837, 2012.

[5] Daniel Hernández-Lobato, José Miguel Hernández-Lobato, Amar Shah, and Ryan P. Adams. Predictive entropy search for multi-objective bayesian optimization. In *ICML*, pages 1492–1501, 2016.

[6] José Miguel Hernández-Lobato, Michael A Gelbart, Ryan P Adams, Matthew W Hoffman, and Zoubin Ghahramani. A general framework for constrained Bayesian optimization using information-based search. *Journal of Machine Learning Research*.

[7] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in neural information processing systems*, pages 918–926, 2014.

[8] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

[9] Joshua Knowles. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, 2006.

[10] Brandon Reagen, Paul Whatmough, Robert Adolf, Saketh Rama, Hyunkwang Lee, Sae Kyu Lee, José Miguel Hernández-Lobato, Gu-Yeon Wei, and David Brooks. Minerva: Enabling low-power, highly-accurate deep neural network accelerators. In *Proceedings of the 43rd International Symposium on Computer Architecture*, pages 267–278. IEEE Press, 2016.

[11] Yakun Sophia Shao, Brandon Reagen, Gu-Yeon Wei, and David Brooks. Aladdin: A pre-rtl, power-performance accelerator simulator enabling large design space exploration of customized architectures. In *ISCA*, pages 97–108, 2014.

[12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *NIPS*, pages 2951–2959. 2012.

[13] Julien Villemonteix, Emmanuel Vazquez, and Eric Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534, 2009.

[14] Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271, 1999.