POSTER: Pairing Up CNNs for High Throughput Deep Learning

Babak Zamirai, Salar Latifi, Scott Mahlke
Computer Science and Engineering
University of Michigan
Ann Arbor, MI
zamirai, salar, mahlke@umich.edu

***Abstract*—To facilitate the efficient execution of convolutional neural networks (CNNs) on cloud servers, this paper proposes Yin Yang (YY), an input-driven synergistic deep learning system, which dynamically distributes CNN computation between a complex (Yang) and a simple (Yin) CNN. YY runs most of the inferences on Yin, while Yang is invoked only when Yin has low confidence. On average, compared to the traditional CNN as a service approach, YY improves datacenter throughput by $1.8\times$ and reduces inference latency by 31% on an NVIDIA TITAN X GPU without any accuracy loss across 21 CNNs.**

***Keywords*-efficient neural network; inference; cloud servers**

I. Introduction

Deep learning is becoming an essential part of using and sharing photos and videos taken by smartphone cameras. However, the size and complexity of CNNs are increasing rapidly to improve their accuracy and functionality, which results in computations with energy requirements beyond edge device's battery constraints. Therefore, edge devices offload CNN computation almost entirely to cloud platforms. However, CNN inference queries require significant amounts of compute resources in comparison to traditional text-based web services. Hence, there has been significant research interest to leverage hardware [1] and software [2] techniques to accelerate deep learning on various platforms.

Traditional CNN optimization techniques are input-invariant and accelerate a single CNN for the entire dataset. However, CNNs are usually overprovisioned and most of the inputs do not require the entire computational power of the model to produce an accurate final output. CNNs with early exits [3] take advantage of the observation that features learned at earlier stages can be used to correctly infer a subset of the data population. By exiting these samples with prediction at earlier stages, they reduce the computation required for inference. However, CNNs with early exits cannot be automatically applied to off-the-shelf CNNs and require machine learning expertise and manual design. Conversely, we hypothesize that additional performance improvements could be achieved through input-dependent acceleration techniques while eliminating the manual design by employing multiple off-the-shelf CNNs, instead of one, and offloading most of the computation on the simple ones.

Conventional ensemble methods activate multiple NNs and combine their outputs to obtain accuracy improvements for extra computation [4]. We investigate an opposing approach by taking inspiration from traditional speculation-recovery techniques and present *Yin and Yang (YY)*, which is input-dependent and dynamically distributes CNN computation between a synergistic pair of a complex CNN (Yang) and a simple one (Yin) to achieve maximum performance and energy efficiency while maintaining the target output quality. Yin is selected in a way that it is capable of performing inferences correctly for a large fraction of inputs while requiring less computation per input instance. To recover from unreliable outputs wherein Yin can only provide a low-confidence answer, Yang is selectively invoked.

YY has three main advantages. First, replacing the complex CNN by a simple CNN for most of the inputs reduces the load on the server and improves the throughput and average response time. Second, the server load reduction releases hardware resources to support more users on the cloud platform. Third, the average latency and energy consumption of the system is improved in comparison to single CNN approaches because of the elimination of the excessive complexity of CNNs for marginal accuracy improvements.

II. Input-Driven CNN Pairing

Although ensemble methods combine NNs to achieve higher accuracy for higher cost, it is not necessarily the only way to take advantage of the synergy among CNNs. The size and accuracy of CNNs are changing drastically over the years. However, even the simplest model classifies more than half of the input instances correctly. Hence, different input instances show different behavior patterns and it might not be necessary to activate the most complex model for every single input to achieve the highest possible accuracy.

Comparing outputs of ResNet-152 and AlexNet with top-1 accuracy of 78.25% and 56.63%, respectively, on the ImageNet test set shows that 53.78% of inputs are classified correctly by both CNNs (*common corrects*). In addition, 18.9% of inputs result in wrong answers regardless of the complexity of CNNs. And more surprisingly, 2.85% of inputs are classified correctly by the simple model but misclassified by the complex one (*odd corrects*). Thus, only 24.47% of inputs require the activation of the complex model and the remaining inputs could be processed by the simple one. Moreover, partitioning the input dataset to two subsets

978-1-7281-3613-4/19/$31.00 ©2019 IEEE
DOI 10.1109/PACT.2019.00065

and activating the proper CNN between ResNet-152 and AlexNet for each input could improve the top-1 accuracy of ResNet-152 by 2.85%. The common correct predictions and the additional accuracy benefit the system in two major ways. First, even by using a non-ideal partitioner with a moderate misclassification rate, it is possible to offload part of the computation on a simpler CNN and achieve the same accuracy as the complex one. Second, there is room to further decrease (less than 24.47%) the activation of ResNet-152 while maintaining its accuracy.

III. Yin and Yang System

Overview: In a traditional approach, the server chooses the most efficient CNN among existing trained models to meet the minimum required accuracy by the user. After that, it occupies the minimum hardware resources to run the model and satisfy the maximum response time restrictions. On the other hand, YY replaces the single CNN with a pair of a complex and a simple CNNs to improve performance and reduce energy consumption while maintaining the target output quality considering the available hardware resources. In addition, it adds a misclassification detector to the system and configures it with the proper strategy to detect and recover the unreliable outputs of Yin.

Design: Based on Section II, a key component to exploit the inherent synergy between a pair of CNNs is employing a proper input partitioning mechanism to offload most of the computation on Yin. A system with a partitioner requires activation for each input instance for selecting the proper CNN to process that input. If Yin is activated, the system will need to check its output reliability and activate Yang for recovering the unreliable outputs. Since inputs of CNN applications are usually non-trivial images, the partitioning task itself requires a CNN to achieve acceptable results. Considering the complexity and overhead of designing, training and running a partitioner per CNN pair, and the fact that an efficient YY design tends to run most of the inputs on Yin, we decide to eliminate the partitioner and keep Yin and the confidence probe always active. Consequently, the key component to exploit the synergy using YY is utilizing a lightweight and accurate-enough confidence probe.

Runtime: Figure 1 shows the runtime overview of YY. For each input instance, YY activates Yin, which is a simpler version of Yang and sacrifices the accuracy for higher performance. To bridge this accuracy gap, the confidence probe evaluates the reliability of Yin's output and sends the likely faulty (low confidence) ones to Yang for recomputation.

Output Confidence: For classification tasks, CNNs convert an input instance (x) to an output vector of K elements (K is the number of classes). Using a softmax ($\frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$) activation function after the output layer provides the estimated probability of that the correct output is j for $j = 1, ..., K$ ($P_j = P(y = j|x)$). Consequently, $\max_{1 \leqslant j \leqslant K} P_j$ represents the confidence level of the CNN in the final output.

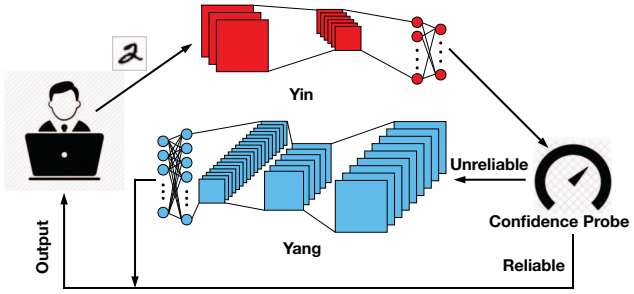

Figure 1. Runtime of YY. All inputs are run on Yin. The confidence probe examines all outputs and triggers Yang for recovering the unreliable ones.

Confidence Probe: Since our design eliminates the partitioner and keeps Yin always active, for each input instance, the lightweight confidence probe compares the output confidence of Yin with a predetermined threshold to detect possible errors and activate Yang for recovery.

IV. Evaluation and Conclusion

In this work, we introduce *Yin and Yang (YY)*, a novel *input-driven synergistic deep learning system*. It pairs an efficient less accurate CNN (Yin) with a complex more accurate CNN (Yang) and offloads most of the computation on Yin. However, for a subset of the inputs, Yin cannot yield confident predictions and is often wrong. Hence, a dynamic confidence examination technique is employed to probe the outputs of Yin and invoke Yang to recover from low-confidence inferences. Compared to the traditional approach, YY improves the datacenter throughput by 1.8× and reduces the average inference latency by 31% on an NVIDIA TITAN X GPU with 100% accuracy of the baseline across a range of 21 CNNs on ImageNet dataset.

Acknowledgment

This research was supported by the National Science Foundation via CCF-1628991.

References

[1] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun *et al.*, "Dadiannao: A machine-learning supercomputer," in *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society, 2014, pp. 609–622.

[2] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[3] S. Teerapittayanon, B. McDanel, and H. Kung, "Branchynet: Fast inference via early exiting from deep neural networks," in *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE, 2016, pp. 2464–2469.

[4] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1, pp. 1–39, 2010.