Modeling Data Reuse in Deep Neural Networks by Taking Data-Types into Cognizance

Nandan Kumar Jha and Sparsh Mittal

Abstract—In recent years, researchers have focused on reducing the model size and number of computations (measured as "multiply-accumulate" or MAC operations) of DNNs. The energy consumption of a DNN depends on both the number of MAC operations and the energy efficiency of each MAC operation. The former can be estimated at design time; however, the latter depends on the intricate data reuse patterns and underlying hardware architecture. Hence, estimating it at design time is challenging. This article shows that the conventional approach to estimate the data reuse, viz. arithmetic intensity, does not always correctly estimate the degree of data reuse in DNNs since it gives equal importance to all the data types. We propose a novel model, termed "data type aware weighted arithmetic intensity" (DI), which accounts for the unequal importance of different data types in DNNs. We evaluate our model on 25 state-of-the-art DNNs on two GPUs. We show that our model accurately models data-reuse for all possible data reuse patterns for different types of convolution and different types of layers. We show that our model is a better indicator of the energy efficiency of DNNs. We also show its generality using the central limit theorem.

Index Terms—Deep neural networks (DNNs), energy-efficiency, arithmetic intensity, roofline model

---

1 INTRODUCTION

DNNs are now being used in a wide range of cognitive applications. After the success of AlexNet [1], the research in DNNs has focused on achieving higher accuracy even at the cost of large DNN size and computational complexity. The focus on accuracy has led to over-parameterized DNNs, e.g., VGG-16 [2], Inception-v4 [3], ResNet152-v2 [4] etc. By contrast, recent networks such as SqueezeNet [5], MobileNet-V1 [6], MobileNet-V2 [7], ShuffleNets [8], etc. focus on making the DNN compact by reducing the number of parameters, or MACs or both. However, reducing the number of MACs does not necessarily make DNNs energy efficient because energy is dominated by data movement rather than computation [9]. The data movement primarily depends on the degree of data reuse present in the workloads.

To enable the deployment of DNN models in a wide range of applications such as autonomous driving and drones, the energy consumption of DNN inference must be within a prescribed envelope. Hence, DNNs need to be carefully examined based on the number of computations (i.e., MACs) and the energy efficiency of MAC operations. Unfortunately, the latter metric has mainly been overlooked in DNN design because a study of energy efficiency requires precise knowledge of the degree of data reuse and parallelism present in the DNNs, and how the underlying hardware platform exploits

this parallelism. Further, the implications of reducing the number of parameters and MACs on the energy efficiency of DNN is not well-understood.

Traditionally, arithmetic intensity [10] is used to model the degree of data reuse. It is also used in the "roofline model" [11] for predicting whether a workload is compute-bound or memory-bound. Therefore, it represents the *degree of data reuse* available in workloads and hence *bandwidth pressure*. Lower arithmetic intensity implies a lower degree of data reuse and high bandwidth pressure and vice versa. The arithmetic intensity considers the memory footprint and the number of arithmetic operations and shows the degree of data reuse available in a workload. In other words, arithmetic intensity shows how efficiently arithmetic operation can reuse the data fetched from different levels in the memory hierarchy. However, the memory footprint does not always reflect the actual number of off-chip accesses, which largely depends on the data reuse available in workload and how well the underlying platform exploits the data reuse available in the workload. Arithmetic intensity can represent power/energy efficiency only when all the data types have the same access behavior. For example, Choi *et al.* [12] and Ghane *et al.* [13] use arithmetic intensity to model the power/energy efficiency.

DNNs have different types of data such as filter weights, input and output activations, partial sums, which have different reuse patterns [14] and hence, reuse importance. Also, the layers in DNNs have distinct computation and reuse patterns with different bandwidth requirements. For example, convolution (Conv) layers have a high degree of reuse, and they are compute-bound, whereas fully connected (FC) layers have low-reuse and a high number of parameters, and hence, they are memory-bound [15]. Moreover, the same layer with different types of convolution possesses a different degree of data reuse (refer Section 2 for more details). Further, due to the different DNN topologies such as branching, skip connections, and dense connections [16], the number of

- Nandan Kumar Jha is with the Indian Institute of Technology Hyderabad, Sangareddy, Telangana 502285, India.
  E-mail: cs17mtech11010@iith.ac.in.
- Sparsh Mittal is with the Indian Institute of Technology Roorkee, Roorkee, Uttarakhand 247667, India. E-mail: sparshfec@iitr.ac.in.
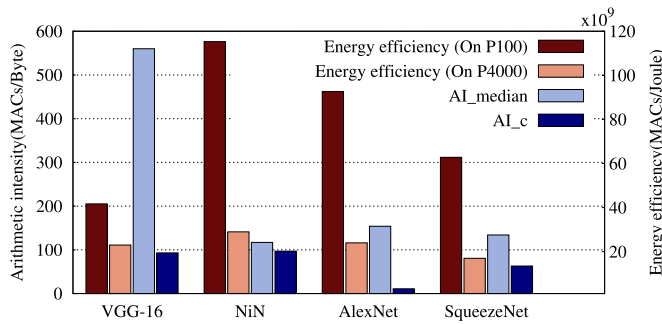
Fig. 1. Neither AI_{median} nor AI_c is a representative of data reuse in DNNs.

TABLE 2
Data Reuse Characteristics of Convolution

Convolution	Arithmetic intensity	$\frac{M_c}{W}$	$\frac{M_c}{A}$
Standard	$\frac{M \times N \times S_k^2 \times S_o^2}{M \times N \times S_k^2 + (M+N) \times S_o^2}$	S_o^2	$\left(\frac{M \times N}{M+N}\right) \times S_k^2$
Pointwise	$\frac{M \times N \times S_o^2}{M \times N + (M+N) \times S_o^2}$	S_o^2	$\left(\frac{M \times N}{M+N}\right)$
Group	$\frac{M \times N \times S_k^2 \times S_o^2}{M \times N \times S_k^2 + g \times (M+N) \times S_o^2}$	S_o^2	$\left(\frac{M \times N}{M+N}\right) \times \frac{S_k^2}{g}$
Depthwise	$\frac{M \times S_k^2 \times S_o^2}{M \times S_k^2 + (M+M) \times S_o^2}$	S_o^2	$\left(\frac{M}{M+M}\right) \times S_k^2$

concurrent activations varies during the runtime [17] which in turn, affects the reuse behavior of DNNs. Given these factors, *it is interesting to investigate whether arithmetic intensity can be used as a representative of energy efficiency in DNN models*, or, is there a need for a more accurate metric.

In Fig. 1, cumulative arithmetic intensity (AI_c) and median value of layer-wise arithmetic intensity (AI_{median}) of DNNs are shown (these metrics are explained in Section 4). VGG-16 [2] and NiN [18] have almost equal AI_c but NiN has significantly higher energy efficiency (measured on both GPU P100 and P4000) than VGG-16, even when VGG-16 has quite higher AI_{median}. Also, AlexNet has lower AI_c and AI_{median} than VGG-16 but has higher energy efficiency than VGG-16. It shows that both layer-wise arithmetic intensity and cumulative arithmetic intensity are not good indicators of energy efficiency of DNNs and hence, *there is a need for better model/metric to estimate the data reuse in DNNs to understand their energy efficiency*.

Our main *contributions* can be summarized as follows.

1) We first explore the intricacies of data reuse patterns (Section 2) and energy-efficiency of various DNNs (Section 3). We also perform a detailed analysis of layer-wise data reuse patterns in DNNs (Section 4). We include all the possible variations of data reuse patterns arising from (a) different types of convolutions such as standard, group, pointwise, depthwise, etc., (b) different types of layers such as Conv, FC, and others, and (c) different design heuristics such as feed-forward/skip connections. Our comprehensive experiment and analysis show that *data reuse estimated by arithmetic intensity is not tightly-coupled with the energy efficiency of MAC operations in DNNs*.

2) We experimentally measure the energy consumption and energy efficiency of DNNs on two GPUs, P100 and P4000, and one CPU. We show that *activation reuse has a higher impact on energy efficiency than weight reuse*.

3) We propose a novel metric, termed "data type aware weighted arithmetic intensity" (DI), which takes data types into cognizance and accounts for the unequal importance of different data types in DNNs. Our proposed metric (DI) more accurately quantifies the intrinsic relationship between data reuse and energy efficiency of MAC operations (Section 5).

4) We validate our model on 25 state-of-the-art DNNs, including both highly-accurate DNNs and compact DNNs, which have between 221 to 15,470 million weights and between 0.54 to 138 million MACs.

5) We also use the "central limit theorem" to prove the generality of our proposed model (Section 6).

2 BACKGROUND AND MOTIVATION

Table 1 lists the symbols used. For simplicity and ease of comparison, we assume that a) height and width of filter are same, b) height and width of output feature map (ofmap) are same, and c) spatial size of input feature map (ifmap) and ofmap are equal.

2.1 Data Reuse Patterns in DNNs

Types of Convolution. Broadly, there are four types of convolutions. They are discussed below and their properties are summarized in Table 2. Here, weight (learnable filter coefficients) reuse and activation (ifmaps and ofmaps) reuse are estimated as M_c/W and M_c/A, respectively. We use arithmetic intensity defined as $\frac{M_c}{W+A}$, as a metric to evaluate the bandwidth requirement of MACs [10].

TABLE 1
Symbols (Fmap = Feature Map, Ops = Operations)

Quantity (symbol)	Unit	Quantity (symbol)	Unit
Energy per pixel (EPP)	Joule	Energy efficiency	MACs/Joule
Average power (P_{avg})	watt	Inference time (I_t)	Millisecond
# Weights (W)	Millions	Throughput	MACs/sec
# MACs (M_c)	Millions	# Activations (A)	Millions
Fmap height (S_o)	-	Fmap width (S_o)	-
Filter height (S_k)	-	Filter width (S_k)	-
# filter-channels (M)	-	# filters (N)	-
Compute to memory ratio (CMR)	OPs/sec/Byte	Group Size (G)	-
Cumulative arithmetic intensity (AI_c)	MACs/Byte	Disparity factor (d_f)	-
Pearson Product Moment correlation coefficient (r_p)	-	Spearman's Rank correlation coefficient (r_s)	-

TABLE 3
Metrics Values Normalized to Standard Convolution (Assuming $N = M = 256$, Feature Map Size ($S_o \times S_o$) = 28×28, Filter Size ($S_k \times S_k$) = 3×3 and Group Size (g) = 4)

Convolution	Arithmeticintensity	M_c	$\frac{M_c}{W}$	$\frac{M_c}{A}$
Standard	1.00	1.000	1	1.000
Pointwise	0.24	0.111	1	0.111
Group	0.45	0.250	1	0.250
Depthwise	0.01	0.004	1	0.004

1. *Standard (Spatial) Convolution:* In standard convolution, filtering (i.e., feature extraction) and combining (i.e., feature aggregation) are performed together. The total number of filter weights and activations (combined ifmaps and ofmaps) involved in standard convolution is $M \times N \times S_k^2$ and $(M + N) \times S_o^2$, respectively. The total number of MAC operations in standard convolution is $M \times N \times S_k^2 \times S_o^2$. Because of combined feature extraction and aggregation, standard convolution incurs high computational complexity.

2. *Pointwise Convolution (PWConv):* In PWConv, the smaller receptive size of filter reduces the number of MACs as well as the number of filter weights involved, which are $M \times N \times S_o^2$ and $M \times N$, respectively. The number of activations is the same as that in standard convolution. Pointwise convolution has been used in NiN [18], inception modules in [3], [19], [20] and in SqueezeNet [5].

3. *Group Convolution:* In group convolution, each group of 2-D filters convolve with only one group (G number) of input feature maps. Compared to standard convolution, in group convolution, the number of MACs and filter weights are reduced by a factor of g ($g = \frac{N}{G}$). The number of activations remains same as that in standard convolution. Group convolution has been used in AlexNet [1] ($g = 2$), ResNext ($g = 32$) [21] and 1.0-G-SqNxt-23 ($g = 2$) [22].

4. *Depthwise Convolution:* Depthwise convolution (DWConv) performs only feature extraction where, one filter convolves with only one input feature map, i.e., one channel of input. Compared to standard convolution, it reduces the number of MACs and number of weights by a factor of N (Table 2). Total number of activations involved in DWConv is $2 \times M \times S_o^2$. Depthwise convolution has been used in MobileNet-V1 [6], MobileNet-V2 [7], and XceptionNet [23]. Note that, DWConv is an extreme case of group convolution where $G = 1$, i.e., $g = N = M$.

Others Layers. The non-Conv layers such as pooling, ReLU, BatchNorm have a negligible number of learnable parameters, and there is no MAC operation involved. However, there are other operations, such as element-wise addition, comparison, and division. These layers have a low arithmetic intensity and high bandwidth requirement. FC layers have a very high number of parameters (weights), and fewer activations, making them memory bound. Hence, in FC layer, both the arithmetic intensity and weight reuse are approximately equal to 1 (as $M_c \approx W$ and $A \ll W$).

2.2 Motivation

Different types of convolutions have been applied to accomplish different design goals and achieve a trade-off between performance and computation/bandwidth overhead. Apart from this, various design heuristics have been used, such as residual connections [4], [24] to facilitate backpropagation in deeper networks, dense connections [25] to enable feature reuse, etc. These design heuristics lead to different computational complexity and degrees of data reuse. Even in standard convolution, the degree of data reuse depends on multiple factors such as filter's dimensions, convolution stride, and dimensions of ifmaps. For the comparison of data reuse and computational complexity in different types of convolution, we assume specific values of variables and show the values of metrics normalized to that for standard convolution in Table 3.

Evidently, the relative number of MACs decreases from standard convolution to DWConv, but the arithmetic intensity also reduces, which increases bandwidth requirement. It is well-known that energy is dominated by data movement rather than computation [9]. Hence, *a decrease in the number of MACs can be dwarfed by the increase in memory accesses, increasing the overall energy consumption.* These observations motivate us to investigate a model that can better incorporate the dynamics of data reuse in DNNs and is a better indicator of the energy efficiency of MACs in DNNs.

2.3 Experimental Setup and Metrics

We perform our experiments using Caffe [26] on two GPUs, viz., Tesla P100 and Quadro P4000 which have significantly different compute and memory resources, as shown in Table 4. Former is a data-center scale GPU while latter is a desktop GPU. We also validate our model on CPU in Section 7.

Power and Inference Time Readings. GPUs have massive compute and memory resources (Table 4); hence, smaller batch sizes can result in resource underutilization, which can lead to an unfair comparison of energy consumption in DNNs with different model size and memory-footprint. In general, larger models have better resource utilization than compact models at smaller B. Hence, for better utilization of GPU compute resources and to enable fair comparison, we take input batch size of four (as used in [8]). For power and inference time measurement, we use `nvidia-smi` utility, which is a high-level utility. The sampling rate of `nvidia-smi` utility depends on the sampling rate of inbuilt power sensors in high-end GPUs, which is quite low [27]. For instance, the

TABLE 4
Configuration of GPUs Used in Our Experiments

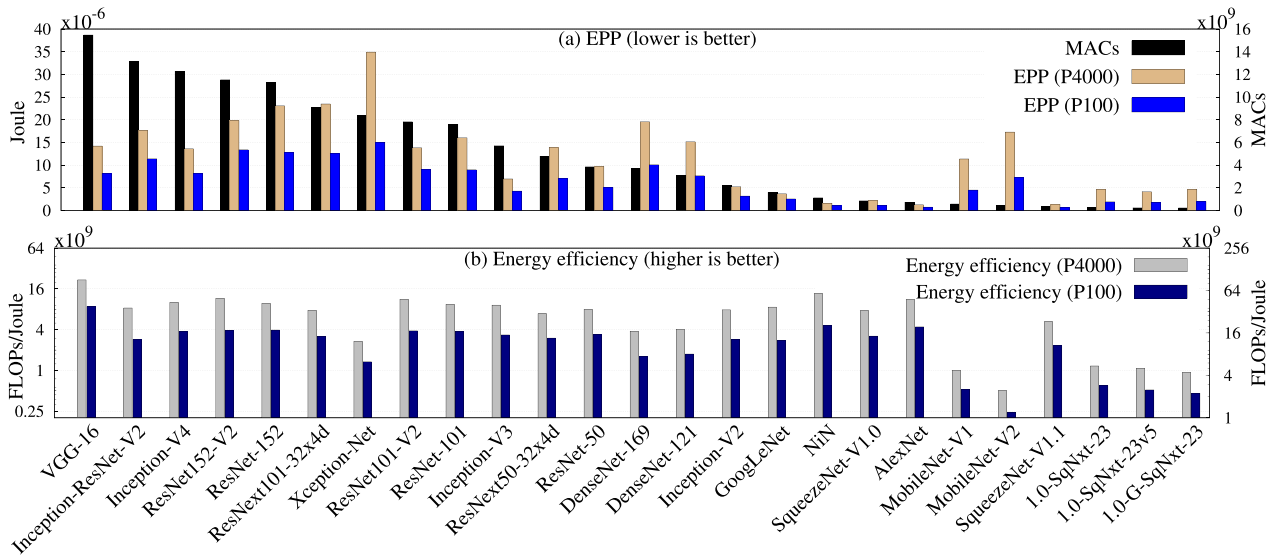| GPU	# core	L2 size	Peak bandwidth	Peak Throughput	CMR
P4000	1792	2 MB	243 GB/s	5.2 TFLOPS	21.4 FLOPs/Byte
P100	3584	4 MB	549 GB/s	9.3 TFLOPS	16.94 FLOPs/Byte

Fig. 2. (a) Energy consumption (EPP) and (b) energy efficiency for 25 DNNs, measured on P4000 and P100 GPUs, illustrated in descending order of number of MACs operations in DNNs. (SqNxt=SqueezeNext).

sampling rate on P100 GPU is $\approx 50 Hz$ (119 samples in 2.36 second), and on P4000 GPU, it is $\approx 1 Hz$ (119 samples in 118.23 sec). Therefore, as suggested in [28], we run the DNN computations for a longer time to see the changes in power reading. In particular, we run 500 iterations for DNNs with very low inference time (AlexNet, NiN, SqueezeNet variants) and 200 iterations for the rest of DNNs shown in Table 7.

For power and inference time readings, we used a similar methodology as used in [29]. The idle power consumption on P100 and P4000 GPUs are 31 Watts and 5 Watts respectively. We sample the power readings at every 100ms, and once the power consumption becomes stable, we take average power reading as the power consumption of DNN computation. Similarly, we use the average forward pass time (over 200/500 iterations) reported in the Caffe deep learning framework as the inference time of DNN. Further, to mitigate the effect of noise and increase the robustness of power and inference time readings, we repeat the steps mentioned above three times and take an average over these iterations. Similarly, we use `pcm-power` utility provided by Intel to measure the power on CPU (Section 7).

Energy Metrics. "Energy per pixel (EPP)" (Eq. (1)) measures the energy consumed in processing of one input pixel over entire DNN, whereas "energy efficiency" (Eq. (2)) shows the number of MAC operations performed per unit of energy. We have used EPP to remove the bias due to differences in input image size used by different DNNs. For example, InceptionV3, InceptionV4, and XceptionNet work with input image size 299×299 while most of DNNs work with input size 224×224 (Table 7).

$$\text{Energy per pixel (EPP)} = \frac{P_{avg} \times I_t}{\#\text{Pixels in input frame}} \quad (1)$$

$$\text{Energy Efficiency} = \frac{(\text{batch size}) \times (\# \text{MACs})}{P_{avg} \times I_t}. \quad (2)$$

Correlation Coefficients. We use the Pearson product-moment correlation coefficient (PPMCC) along with Spearman's rank correlation coefficient (SRCC) to examine the relationship

between two variables. PPMCC measures the strength of a linear relationship between two variables using the absolute value of data. By contrast, SRCC a nonparametric test that evaluates the monotonicity between two variables using the rank of data, without any presumption about the data distribution [30], [31]. A value close to +1/-1 for both PPMCC (r_p) and SRCC (r_s) indicates strong positive/negative correlation whereas a value 0 indicates no correlation between two variables.

Importance of SRCC. When we use PPMCC in conjunction with SRCC, we get more insights about the relationship between two variables. For example, since SRCC uses the rank of data instead of absolute values, the effect of outliers is quite meager as compared to that on PPMCC. Therefore, when there is a significant difference between PPMCC and SRCC, either the sample size is insufficient or there exists a group of data points for which a linear/monotonic relationship does not exist. In this paper, we perform experiments on a sufficiently large sample size of 25 DNNs (Section 6), and the results with PPMCC and SRCC help find the outliers. Furthermore, the comparative study between PPMCC and SRCC helps analyze the batch size sensitivity of our proposed model.

Concurrent Activations. This includes all the data that needed for the execution of an operation such as convolution [17]. For example, in forward pass, concurrent activation for a convolution operation consist of ifmaps and ofmaps of the current layer, and outputs of previous layers if feed-forward (residual or skip) connections are present in the network. The size of maximum concurrent activation depends on types of operation, e.g., different types of convolution (Table 2), ReLU operation, weight update in back-propagation, etc., and the network topology (linear/non-linear architecture [32]). Hence, through the inter-layer dependency, which can be obtained from the network's computational graph and the execution order, the size of maximum concurrent data can be estimated.

3 ENERGY-EFFICIENCY OF DNNs

Fig. 2a shows EPP and MAC operations, whereas Fig. 2b shows energy efficiency of DNNs. EPP and energy efficiency
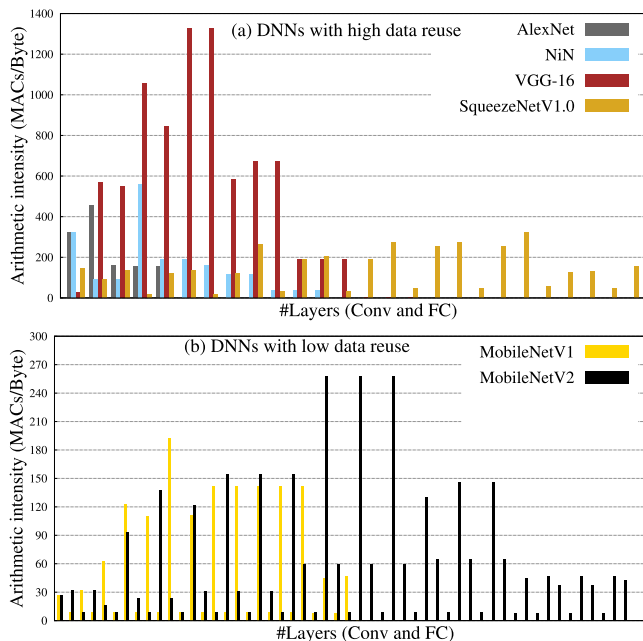
Fig. 3. Layer-wise arithmetic intensity in DNNs with high data reuse (top) and DNNs with low data reuse (bottom).

values are measured experimentally on P4000 and P100 GPUs. To make the comparison easier, on X-axis, DNNs are arranged in decreasing order of MAC operations.

Variation Due to Depthwise Separable Convolution. The energy efficiency of MobileNet variants (i.e., MobileNet-V1 and MobileNet-V2) and XceptionNet is significantly lower than that of neighboring DNNs with a higher number of MACs. Hence, their energy per pixel is significantly higher than the neighboring DNNs with a higher number of MACs. In MobileNet variants and XceptionNet, depthwise separable convolution (DWConv, followed by PWConv) has been used to reduce the number of MACs. Both the DWConv and PWConv have a significantly lower degree of data reuse (Table 2), which results in costlier memory access and makes MACs energy inefficient.

Variations Due to Feed-Forward/Skip Connections. Even though the number of MACs in MobileNet-V2 is lower than that in MobileNet-V1 and both these networks perform depthwise separable convolution (DWSConv), EPP of the former is significantly higher than that of the latter. This is because MobileNet-V2 has feed-forward connections that increase the concurrent activations and further reduce the already-low data reusability of DWSConv. This translates to low energy efficiency and high EPP of MobileNet-V2. Similarly, the feed-forward connections in XceptionNet exacerbate the low activation reusability in DWSConv and lower the energy efficiency. Also, XceptionNet has a relatively higher number of MACs, which, in conjunction with low data reusability arising from DWSConv and feed-forward

connections, results in the highest EPP among 25 DNNs. In DenseNet models presence of skip connections result in a concatenation of fmaps from previous layers [16] and increases the number of activations at runtime. Since the number of activations primarily drives the memory-footprint [33], skip connections results in higher memory-footprint, which in turn increases memory access hence reduces energy efficiency. For the same reason, DenseNet variants also have quite low energy efficiency than their neighboring DNNs, which leads to higher EPP compared to neighboring DNNs with relatively higher MACs.

Variation Due to Low Activation Reuse. Compared to SqueezeNet-V1.1, the variants of SqueezeNext (1.0-G-SqNxt-23, 1.0-SqNxt-23, and 1.0-SqNxt-23v5) have a lower number of MAC operations; however, the EPP of SqueezeNext variants is much higher than that of SqueezeNet-V1.1. The activation reuse in SqueezeNext variants is substantially low compared to that in SqueezeNet-V1.1, and hence, the latter's energy efficiency is substantially higher than the variants of SqueezeNext.

In summary, energy per pixel of a DNN depends on the total number of MAC operations and the energy consumed by each MAC operation. Former can easily be estimated at the design time, but estimating the latter is difficult due to varying degrees of data reuse in different layers of DNNs.

4 CONVENTIONAL APPROACHES

We discuss two possible approaches for estimating the data reuse in DNNs and also show their limitations. These approaches are (1) layer-wise arithmetic intensity (Section 4.1) and (2) cumulative arithmetic intensity (AI_c) (Section 4.2).

4.1 Layer-Wise Arithmetic Intensity

Fig. 3 shows the arithmetic intensity of a layer (Conv and FC) defined as the ratio of "number of MACs performed in that layer" to "the sum of the total number of weights and activations in that layer". We divide DNNs in two categories based on their degree of data reuse: those with higher data reuse, e.g., AlexNet, VGG-16, NiN, and SqueezeNetV1.0 (Fig. 3a) and those with lower data reuse, e.g., MobileNet-V1 and MobileNet-V2 (Fig. 3b). From Fig. 3a, we observe that arithmetic intensity of nearly all layers of VGG-16 and NiN are higher and lower (respectively) than that of other DNNs. Similarly, from Fig. 3b, the layer-wise arithmetic intensity in MobileNet-V2 is comparable or higher than that of the MobileNet-V1.

Table 5 shows the median and variance of the arithmetic intensity of layers in these DNNs. The median value of layer-wise arithmetic intensity in VGG-16 is significantly higher than the median value of layer-wise arithmetic intensity in other DNNs with higher data reuse. Also, NiN has the lowest median among the DNNs with higher data reuse.

TABLE 5
Median and Variance of Layer-Wise Arithmetic Intensities in DNNs (SqNet=SqueezeNetV1.0)

Metric	AlexNet	VGG-16	NiN	SqNet	MobileNet-V1	MobileNet-V2
Median	154	560	117	134	18	32
Variance	2.69E+04	2.09E+05	2.24E+04	8.30E+03	3.70E+03	4.52E+03

Intuitively, the energy efficiency of VGG-16 should be higher compared to other DNNs with high data reuse, and also, the energy efficiency of NiN should be lowest in the same group. Surprisingly, on both P4000 and P100 GPUs, *the energy efficiency of NiN is highest, and that of the VGG-16 is lowest among the DNNs with higher data reuse*.

Similarly, MobileNet-V2 has lower energy efficiency than MobileNet-V1, which is counter-intuitive. By observing the variance of the layer-wise arithmetic intensity of DNNs in Table 5, one may argue that since the variance values of VGG-16 and MobileNet-V2 are higher than the other DNNs in their respective groups, hence their energy efficiency is lowest, despite having higher median. This argument is flawed because it cannot explain why NiN has higher energy efficiency than SqueezeNetV1.0, even when NiN has a significantly higher variance than SqueezeNetV1.0. The above discussion proves that *"layer-wise arithmetic intensity" is not a good indicator of energy efficiency of DNNs* with both high and low data reuse. Besides this, finding layer-wise arithmetic intensity in DNNs such as Inception-V3, Inception-V4, and Inception-ResNet-v2, is tedious because these networks have thousands of Conv layers and many layers have several branches leading to irregular computation and reuse patterns.

4.2 Cumulative Arithmetic Intensity

For each DNN, a single metric, viz., arithmetic intensity ($AI_c = \frac{M_c}{W+A}$) is defined as the ratio of "total number of MAC operations performed by network in one forward pass" to "the sum of total number of weights and activations that network has". We have plotted the roofline models with AI_c for 25 DNNs (Table 7) on two GPUs P4000 (Fig. 5a) and P100 (Fig. 5b). We also measure the energy efficiency of 25 DNNs on P4000 and P100 and plot in Figs. 5e and 5f, respectively.

In the roofline models (with AI_c) on both GPUs (Figs. 5a and 5b), MobileNet-V1, DenseNet and XceptionNet are predicted as compute-bound whereas AlexNet is predicted as bandwidth-bound. It is well known that due to the costlier off-chip accesses, bandwidth-bound operations are energy inefficient compared to compute-bound operations. Despite this, AlexNet has substantially higher energy efficiency than MobileNet-V1, DenseNet, and XceptionNet on both the GPUs (refer Figs. 5e and 5f). Also, XceptionNet and DenseNet are predicted as compute-bound, but their energy efficiency is lower than that of the AlexNet. In summary, *data reuse predicted by AI_c is not correlated with the energy efficiency of DNNs*.

To understand the reason behind limitations of AI_c, we study the architecture of XceptionNet and DenseNet. We found that DenseNet has many skip connections which results in concatenation of fmaps from previous layers. This concatenation of fmaps increases the number of concurrent activations [16], [17] and decreases the effective data reuse. Similarly, XceptionNet uses DWSconv which has very low data reuse (Table 3). It also has skip connections which increase the concurrent activation data and further reduce the data reuse. AI_c ($= \frac{M_c/W}{1+A/W} = \frac{M_c/A}{1+W/A}$) gives equal importance to weight reuse (M_c/W) and activation reuse (M_c/A) and hence, it is unable to capture the runtime change in data reuse.

5 PROPOSED MODEL

In this section, we first discuss the need to give unequal importance to the reuse of different data types, specifically weights and activations (Section 5.1). We then propose a model that more accurately incorporates the dynamics of data reuse in DNNs (Section 5.2). We highlight the effectiveness of our model (Section 5.4) and compare it with AI_c to get more insights and explain why AI_c fails to predict the nature (memory-bound/compute-bound) of some DNNs.

5.1 Reuse of Different Data Types has Unequal Importance

As we know, the number of weights (W) in a DNN does not change at runtime. However, the number of concurrent activations can change at run time and grows in proportion to the number of feed-forward connections (as explained in Section 2.3). DNNs such as DenseNet have a relatively higher number of skip connections, which leads to a substantial increase in concurrent activations. Further, as shown in Table 7, the ratio of the total number of activations to the total number of weights, i.e., $\frac{A}{W}$ varies significantly across different DNNs, ranging from 0.03 in AlexNet to 32.80 in 1.0-G-SqNxt. This imbalance between W and A creates an imbalance between $\frac{M_c}{W}$ and $\frac{M_c}{A}$, for example, compact DNNs such as MobileNet and SqueezeNext have very low $\frac{M_c}{A}$ (Table 7). To account for the imbalance between the weight and activation reuse and also to model the runtime change in effective data reuse, our model decouples the weight and activation reuse.

Does a metric that decouples weight and activation reuse inherit the properties of arithmetic intensity? As shown in Table 2, the arithmetic intensity of commonly used convolution types are different. On decoupling the data reuse in terms of weight and activation reuse, we find that in all convolution types, the weight reuse is the same (i.e., S_o^2), whereas activation reuse is different. *This is quite interesting because it shows that the variation in activation reuse governs the variation in arithmetic intensity for different convolution types.* Since lower arithmetic intensity leads to higher bandwidth pressure, lower activation reuse leads to a higher number of memory accesses, making DNN energy inefficient. Based on these insights, activation reuse should be given more importance than weight reuse for computing the overall data reuse. As shown in Table 3, the activation reuse decreases from standard convolution to DWConv in the same order in which the relative value of arithmetic intensity is decreasing (but with different magnitudes). This confirms that *a metric that decouples weight and activation reuse inherits the properties of arithmetic intensity.*

5.2 Decoupling the Weight and Activation Reuse

We now decouple weight, and activation reuse from the formulation of AI_c and establish a relation between arithmetic intensity (AI_c) and weight and activation reuse. Since arithmetic mean is never less than harmonic mean, we have

$$\frac{W+A}{2} \geqslant \frac{2 \times W \times A}{W+A} \Rightarrow \frac{2 \times M_c}{W+A} \leqslant \frac{M_c \times (W+A)}{2 \times W \times A}$$

$$\Rightarrow \frac{M_c}{W+A} \leqslant \frac{1}{4} \times \left[\frac{M_c}{A} + \frac{M_c}{W} \right]$$
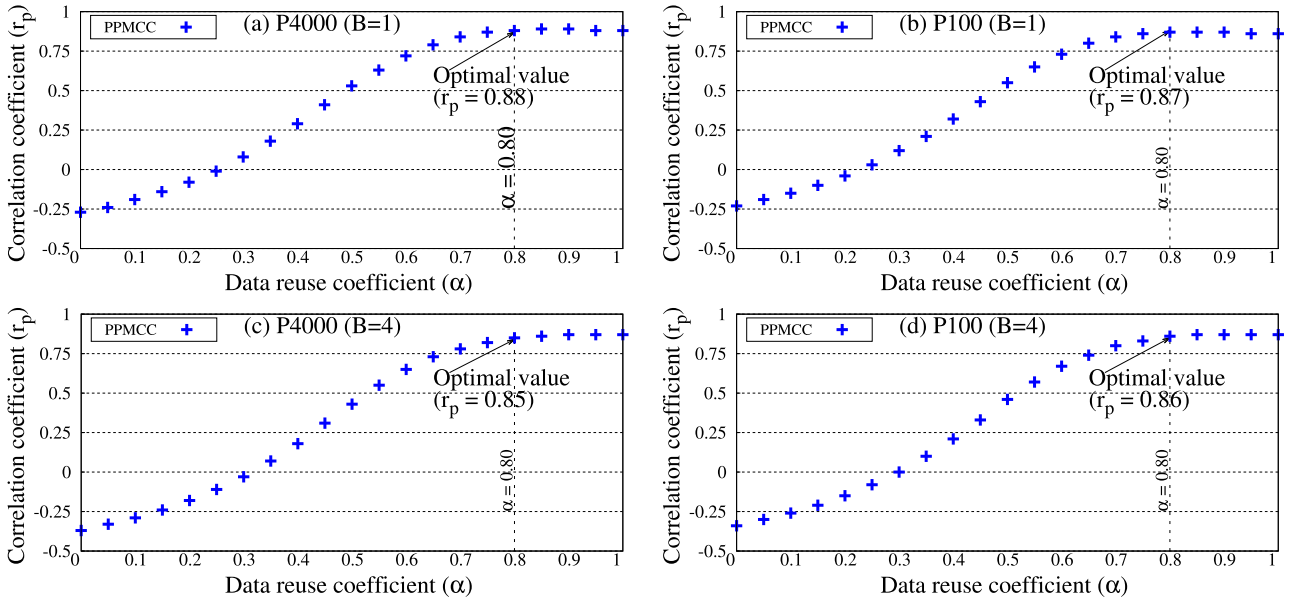
Fig. 4. Variations in the PPMCC calculated between weighted arithmetic intensity at different values of α, and energy efficiency measured for 25 DNNs on P4000 GPU with B=1 (a), and $B = 4$ (c). (b) and (d): these results on P100 GPU.

$$\Rightarrow AI_c \leqslant \frac{1}{4} \times [\text{ActivationReuse} + \text{WeightReuse}]. \quad (3)$$

To give unequal importance, we introduce a *data reuse coefficient* (α) in Eq. (3), where $0 \leq \alpha \leq 1$. Different values of α would give different weightage to both types of reuse. After introducing α in Eq. (3), we refer resultant metric as "data type aware weighted arithmetic intensity" (DI).

$$DI = \frac{[\alpha \times \text{ActivationReuse} + (1 - \alpha) \times \text{WeightReuse}]}{4}. \quad (4)$$

To find the value of α such that DI would become a more accurate indicator of energy efficiency, we find the Pearson correlation coefficient (r_p) between DI and energy efficiency for α values between 0 to 1 with a step size of 0.05 (Fig. 4). We experimentally measure the energy efficiency of 25 state-of-the-art DNNs on both the GPUs P4000 and P100. Figs. 4 and 6 show the results. We observe that on both GPUs, with increasing value of α, the correlation (r_p) continues to increase till it reaches a plateau at $\alpha \approx 0.80$ (Fig. 4). However, the correlation (r_s) does not saturate at α=0.8 and keeps increasing even with the higher values of α (Fig. 6). This difference between the saturation points indicates that there exist few DNNs for which the linear relationship between the weighted arithmetic intensity and energy-efficiency does not hold at higher α. Hence, we take $\alpha = 0.80$ and substitute this value in Eq. (4) that gives the final expression for DI. Note that the trends in the increase in both the r_p and r_s with the higher values of α are consistent on both GPUs, which have significantly different compute power and CMR (Table 4). Thus, *the proposed metric DI is platform-agnostic*. Also, at α=0.8 the absolute value of r_p is higher than that of the r_s. Hence, even if there are outliers in the data (which affect r_p more than r_s), the linear association between DI and energy-efficiency exists for those outliers too.

5.3 Batch Size Sensitivity of Our Proposed Model

To study the effect of batch size on our proposed model (DI with $\alpha = 0.80$) we plot the r_p (Fig. 4) and r_s (Fig. 6) with

batch size one and four. The trend in variation of r_p and r_s with α is similar across both B=1 and B=4. Also, there is negligible change in absolute values of r_p and r_s at α=0.8. Thus, even though changing B alters the energy/power consumption of different DNNs differently (Table 7), the relationship between weighted arithmetic intensity and energy-efficiency of DNNs remains linear irrespective of B.

Increasing B increases the data level parallelism and arithmetic intensity [34]. More precisely, it increases the weight reuse; however, the activation reuse remains constant with a change in B. The variation in data reuse in a DNN is either due to different layer types (Conv, FC, non-Conv) or layers' dimensions. Therefore, when activation reuse is very low due to the presence of a particular type of convolution, then energy-efficiency seldom improves with a higher B. For example, both DWConv and FC layers are bandwidth-bound because the former has very low activation reuse (and higher weight reuse), whereas the latter has low weight reuse (and higher activation reuse) [15]. However, FC layers become compute-intensive at higher B [34]; whereas energy-efficiency of the DWConv layer does not improve with increasing B. Thus, if our proposed model would have been sensitive to B, the importance of weight reuse would have reduced further, and the saturation points in Fig. 4 would have shifted towards the right (i.e., α would exceed 0.8). In reality, saturation points remain the same across both B=1 and B=4. *Evidently, our model remains valid for different values of B.*

In Table 6, we compare DI with AI_c using r_p and r_s. Evidently, compared to AI_c, DI has a stronger correlation (r_p) with energy efficiency as measured for 25 DNNs on both P4000 and P100 GPUs. However, the difference between r_s for AI_c and DI is quite small as compared to the difference between r_p for AI_c and DI (Table 6). For example, the difference between r_s is only 0.04, whereas that between r_p is 0.26. This further strengthens our claim that there is a group of DNNs for which the conventional metric AI_c does not have a linear relationship with energy-efficiency, but DI does

TABLE 6
Correlation (r_p and r_s) of AI_c and DI With Energy
Efficiency of 25 DNNs Measured on P4000
and P100 GPUs With $B=1$ and $B=4$

Batch size	GPU	Correlation	Metric	CorrelCoeff.
$B=1$	P4000	PPMCC (r_p)	AI_c	0.62
			DI (ours)	0.88
		SRCC (r_s)	AI_c	0.72
			DI (ours)	0.76
	P100	PPMCC (r_p)	AI_c	0.63
			DI (ours)	0.87
		SRCC (r_s)	AI_c	0.68
			DI (ours)	0.73
$B=4$	P4000	PPMCC (r_p)	AI_c	0.70
			DI (ours)	0.85
		SRCC (r_s)	AI_c	0.72
			DI (ours)	0.79
	P100	PPMCC (r_p)	AI_c	0.66
			DI (ours)	0.86
		SRCC (r_s)	AI_c	0.59
			DI (ours)	0.73

have a linear relationship. Hence, DI is a better indicator of the energy efficiency of DNNs. From Table 2, we conclude that *its the activation reuse which causes variation in arithmetic intensity, not the weight reuse*. In our model, we obtained $\alpha = 0.80$, which indicates that activation reuse has a much higher impact on DI than weight reuse. Evidently, our model confirms the intuition (Section 5.1) and highlights the importance of activation reuse.

5.4 Salient Features of Our Model

We first discuss how modeling of activation and weight reuse captures all types of reuses across different layers in DNNs. Based on the computation and data reuse patterns, layers in DNNs can be broadly categorized as Conv, non-Conv, and FC layers. In Conv layers, there is feature map reuse, filter reuse, and filter weight reuse [14], whereas in FC layers, except filter-weight reuse, all these types of reuses are possible. In other words, all the possible data reuse in both Conv and FC layers can be expressed as weight and activation reuse. However, in non-Conv layers, only ifmap and ofmap activations are processed, and their reuse can be described as activation reuse.

As shown in Table 2, the degree of data reuse in different types of convolutions is different due to the activation reuse, whereas weight reuse remains constant across all types of convolutions. Hence, energy efficiency of MACs for different types of convolutions can be expressed in terms of activation reuse. Since non-Conv layers do not have learn-able parameters and process only feature maps, their energy metrics are governed by activation reuse. However, in FC layers, $M_c \approx W$ and $A \ll W$, thus, arithmetic intensity and weight reuse are approximately equal to 1. Hence, the energy efficiency of MACs in FC layers is governed by weight reuse.

In summary, except FC layer, all the layers' energy efficiency can be expressed in terms of activation reuse. The higher value of α implies higher importance of activation reuse. Hence, FC layers have a lower impact on the energy efficiency of DNNs. In fact, deeper networks such as Inception-V4, Inception-ResNet-V2 have hundreds of Conv and

non-Conv layers, but very few FC layers and some networks such as NiN have no FC layers at all.

We now discuss how our model addresses the shortcomings of AI_c. As discussed in Section 4.2, AI_c predicts Alex-Net as memory-bound and MobileNet-V1 as compute-bound (shown in roofline models in Figs. 5a and 5b) but the energy-efficiency of AlexNet is quite high and that of the MobileNet-V1 is quite low (Figs. 5e and 5f). This is counter-intuitive because operations (MACs) of memory-bound workload are energy-inefficient due to the higher number of memory-accesses. The reason for these irregularities are better explained by our model. As shown in Table 7, Alex-Net has very low $\frac{M_c}{W}$ but significantly high $\frac{M_c}{A}$, whereas MobileNet-V1 has high $\frac{M_c}{W}$ but significantly low $\frac{M_c}{A}$. By virtue of giving higher importance to $\frac{M_c}{A}$, our model is able to accurately predict AlexNet as compute-bound and Mobile-Net-V1 as memory-bound (Figs. 5c and 5d)

5.5 When and Why Does AI_c Fail?

We define relative disparity (d_f) between AI_c and DI as

$$d_f = \left(\frac{AI_c - DI}{AI_c} \right) \times 100 = 75 - 6.25 \times \left[\frac{A}{W} + 3 \times \frac{W}{A} \right].$$
(5)

Equation (5) shows that, $\frac{A}{W}$ has less impact on relative disparity (d_f) than $\frac{W}{A}$. Since $\frac{A}{W}$ is same as weight reuse to activation reuse ratio, Eq. (5) implies that weight reuse has less impact on d_f. Table 7 shows weight reuse ($\frac{M_c}{W}$), activation reuse ($\frac{M_c}{A}$), AI_c and DI, activation to parameter ratio ($\frac{A}{W}$) and d_f value for 25 state-of-the-art DNNs. For gaining more insights, we study three cases which are shown in Table 8.

As shown in Table 8, in *case 1*, activation reuse ($\frac{M_c}{A}$) dominates total data reuse ($\frac{M_c}{W} + \frac{M_c}{A}$), but AI_c is nearly equal to weight reuse ($\frac{M_c}{W}$). This leads to huge disparity between AI_c and DI. For example, AlexNet has $30\times$ higher activation reuse than weight reuse and hence, its relative disparity is highest among all the 25 DNNs (Table 7). In *case 3*, weight reuse dominates the total data reuse, however, disparity is noticeable but not as large as in case 1 because weight reuse has less impact compared to activation reuse (refer Eq. (4)). In *case 2*, the d_f is lower and AI_c would be able captures the dynamics of data reuse in DNNs. In summary, when either $A \approx W$, for example in variants of InceptionNet, ResNet and ResNext (Table 7), or when A is significantly higher than W, for example MobileNet-V2 and variants of Squeeze-Next (Table 7), AI_c would be able to capture the data reuse patterns in DNNs. *However, AI_c fails to estimate the data reuse when the $\frac{A}{W}$ ratio is very low (e.g., AlexNet) and also, when $\frac{A}{W}$ ratio is moderately high (e.g., MobileNet-V1).*

6 GENERALITY AND USE CASES OF DI

6.1 Generality of Proposed Model

For the generality test, we use confidence intervals for the population correlation coefficient (ρ), which measures the linear correlation between two variables over the entire population. Note that the sample (Pearson) correlation coefficient (r_p) is a measure of the correlation between the two variables over a sample (ϕ) taken randomly from the

(a) Roofline model (with AI_c) on P4000 GPU

(b) Roofline model (with AI_c) on P100 GPU

(c) Roofline model (with DI) on P4000 GPU

(d) Roofline model (with DI) on P100 GPU

(e) Throughput and energy-efficiency on P4000 GPU
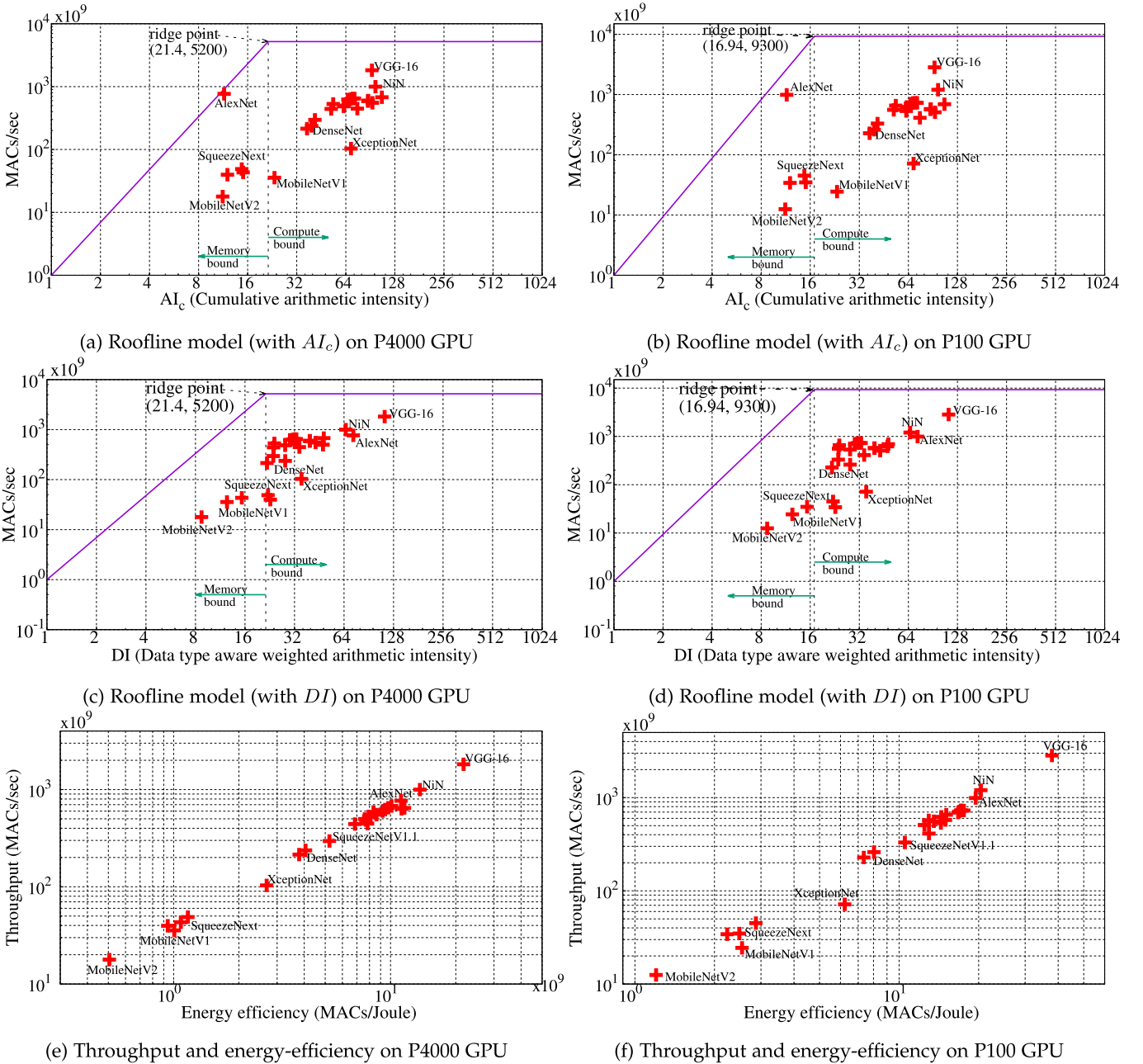
(f) Throughput and energy-efficiency on P100 GPU

Fig. 5. Roofline model with (a) conventional metric (AI_c), and (c) our proposed metric (DI) on x-axis; and (e) energy efficiency measured on P4000 GPU for 25 DNNs. (b), (d), (f): these results on P100 GPU. All measured values are with B=4.

population. We perform the following steps to compute the confidence intervals for ρ at a given r_p and ϕ.

Step 1: "Central limit theorem" can be applied when the data follow normal distribution or the sample size is large. In our experiment, ϕ is 25 which is large enough (refer Eq. (6)) to apply "central limit theorem". To get normal distribution, we transform r_p using "Fisher's Z transform" as $Z_r = \frac{1}{2} \times \log_e\left(\frac{1+r_p}{1-r_p}\right)$ [35]. Then, we calculate standard error (S_e) which is approximated as $\frac{1}{\sqrt{\phi-3}}$ [35].

Step 2: For 95 percent confidence, the upper limit (U_{r_p}) and lower limit (L_{r_p}) of confidence intervals are $U_{r_p} = Z_{r_p} + (1.96 \times S_e)$ and $L_{r_p} = Z_{r_p} - (1.96 \times S_e)$ respectively. Similarly, for 99 percent confidence, $U_{r_p} = Z_{r_p} + (2.58 \times S_e)$ and $L_{r_p} = Z_r - (2.58 \times S_e)$ respectively [35]. Note that these confidence intervals are corresponding to Z_{r_p}.

Step 3: We compute inverse Fisher transform [35] to calculate the upper (U) and lower (L) limit of confidence intervals corresponding to r_p, which are $U = \frac{e^{2 \times U_{r_p}} - 1}{e^{2 \times U_{r_p}} + 1}$ and $L = \frac{e^{2 \times L_{r_p}} - 1}{e^{2 \times L_{r_p}} + 1}$.

Observations. Table 9 shows the confidence interval for both 95 and 99 percent confidence. For a better approximation of ρ using r_p, window size (Δ) for a confidence interval, should be as narrow as possible. Smaller window size implies a lesser deviation in ρ and ensures that the same correlation would hold in other sets of samples taken from a large population. For AI_c on P4000 GPU, at 99 percent confidence, L=0.31 and Δ=0.58. On P100 GPU, these values are $L = 0.27$ and Δ=0.63. This shows that, AI_c can have very poor correlation with energy efficiency in some cases, e.g., for AlexNet, relative disparity (d_f) is very high (Table 7). By comparison, with DI at 99 percent confidence, $L = 0.61$ and
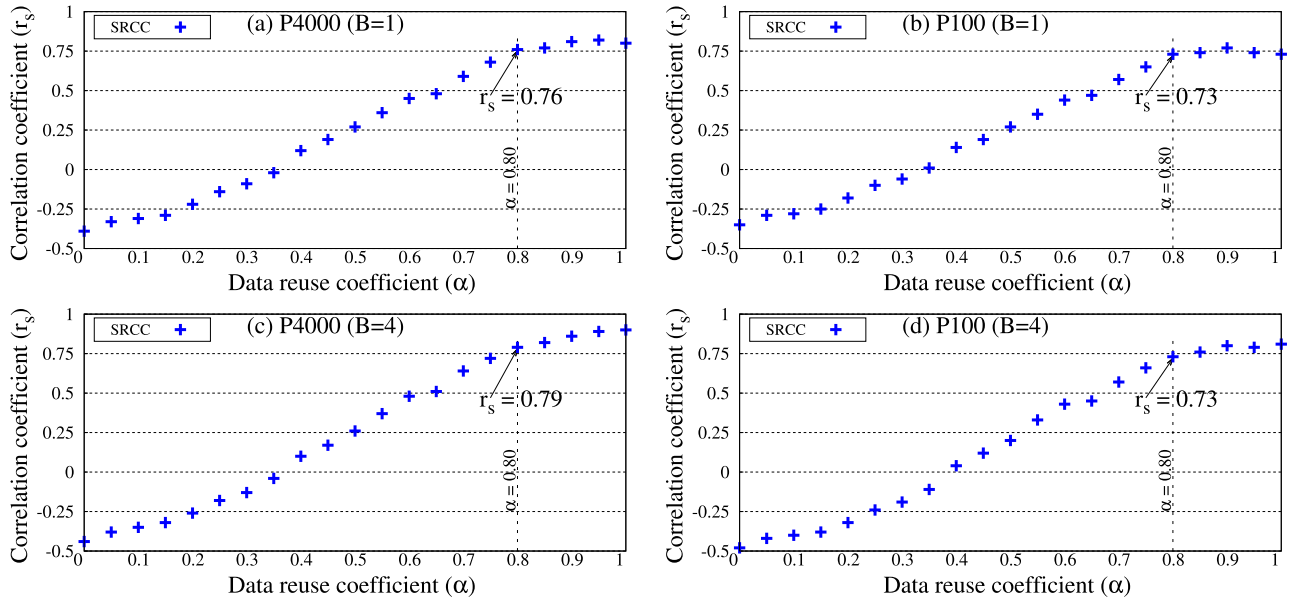
Fig. 6. Variations in the SRCC calculated between weighted arithmetic intensity at different values of α, and energy efficiency measured for 25 DNNs on P4000 GPU with B=1 (a), and $B = 4$ (c). (b) and (d): these results on P100 GPU.

$\Delta = 0.34$ on P4000 GPU, whereas $L = 0.63$ and $\Delta = 0.32$ on P100 GPU. Thus, it can be said with 99 percent confidence that ρ lies between 0.61 to 0.95 on P4000 and between 0.63 to 0.95 on P100 GPU. Clearly, *our model exhibits better positive correlation with energy efficiency in any population of DNNs on both GPUs.*

Minimum ϕ Required for Generality Test. Unlike Δ, window size corresponding to Z_{r_p}, i.e., $\Delta_r = U_{r_p} - L_{r_p}$, is independent of r_p. We find minimum sample size (ϕ) such that $\Delta_{r_p} \leqslant 1$ at 95 percent confidence.

$$\Delta_{r_p} \leqslant 1 \Rightarrow 2 \times 1.96 \times \frac{1}{\sqrt{\phi - 3}} \leqslant 1 \Rightarrow \phi \geqslant 18.37. \quad (6)$$

The minimum number of DNNs required for the generality test is 19. We take ϕ as 25 for which $\Delta_{r_p} = 0.836$.

6.2 Use Cases for Proposed Model

The design process of a DNN (e.g., pruning, quantization) requires massive human efforts to fine-tune the design hyper-parameters [36]. In DNN pruning techniques, deciding the compression ratio for each layer is a daunting task, especially for a deeper network such as ResNet-152 [24] and Inception-V4 [20]. To save these human efforts, there is a growing trend for automating the design of machine learning models, which is termed as AutoML. For example, He *et al.* [36] automate the task of pruning using reinforcement

TABLE 7
Comparison of Data Reuse in Terms of Weight ($\frac{M_c}{W}$) and Activation Reuse ($\frac{M_c}{A}$);
Conventional Metric (AI_c) and **Proposed Metric** (DI) for 25 DNNs

| Model Name | Image size | $\frac{M_c}{W}$ | $\frac{M_c}{A}$ | AI_c | DI | d_f | $\frac{A}{W}$ | P100 | | | | P4000 | | | |
| | | | | | | | | B=1 | | B=4 | | B=1 | | B=4 | |
								I_t	P_{avg}	I_t	P_{avg}	I_t	P_{avg}	I_t	P_{avg}
AlexNet [1]	224×224	11.85	361.50	11.48	72.89	**-535.16**	0.03	2.21	37.0	2.92	50.8	2.40	60.5	3.77	68.6
VGG-16 [2]	224×224	111.81	537.15	92.55	113.02	-22.12	0.21	9.19	58.3	21.83	74.8	10.50	76.0	34.00	83.8
NiN [18]	224×224	146.05	291.34	97.28	65.57	32.60	0.50	2.88	35.0	3.69	59.0	2.30	59.0	4.45	73.0
GoogLeNet [19]	224×224	227.14	158.05	93.20	42.97	53.90	1.44	10.64	20.5	12.50	40.8	8.36	44.2	11.50	64.2
Inception-V2 [20]	231×231	196.43	122.22	75.34	34.27	54.52	1.61	18.54	17.3	21.40	31.7	14.50	41.6	19.70	57.0
Inception-V3 [20]	299×299	240.34	138.40	87.82	39.70	54.80	1.74	29.40	21.5	39.80	38.4	24.91	48.1	38.12	65.2
Inception-V4 [3]	299×299	287.49	169.10	106.47	48.19	54.74	1.70	55.83	24.0	71.00	41.3	41.45	51.2	72.50	67.0
ResNet-50 [24]	224×224	151.41	82.83	53.54	24.14	54.92	1.83	15.47	27.3	23.60	43.5	14.56	53.1	29.20	66.8
ResNet-101 [24]	224×224	170.37	107.51	65.91	30.02	54.46	1.58	32.06	26.0	42.83	42.0	26.50	52.6	48.70	66.0
ResNet-152 [24]	224×224	187.74	112.88	70.49	31.96	54.66	1.66	46.42	25.5	62.10	41.4	38.00	52.2	69.60	66.5
ResNet101-V2 [4]	224×224	175.57	112.20	68.45	31.22	54.40	1.56	32.26	26.5	43.00	42.5	26.00	53.0	49.44	56.3
ResNet152-V2 [4]	224×224	191.56	116.23	72.34	32.82	54.62	1.65	44.13	26.9	63.13	42.5	38.38	53.5	71.30	55.8
Inception-ResNet-V2 [3]	299×299	235.85	139.04	87.47	39.60	54.73	1.70	71.10	24.7	92.48	44.0	56.30	50.3	89.64	70.4
ResNext50-32x4d [21]	224×224	190.97	71.59	52.07	23.87	54.16	2.67	23.50	25.8	34.27	41.2	22.24	50.0	43.25	64.5
ResNext101-32x4d [21]	224×224	205.52	89.00	62.11	28.08	54.79	2.31	53.56	22.8	68.00	37.1	43.84	47.7	74.80	63.0
DenseNet-121 [25]	224×224	385.96	44.01	39.50	28.10	28.87	8.77	35.82	16.6	47.32	32.4	30.00	42.0	52.02	58.4
DenseNet-169 [25]	224×224	262.90	43.08	37.01	21.76	41.21	6.10	49.70	16.3	65.30	31.0	42.00	41.8	69.45	56.5
SqueezeNet-V1.0 [5]	224×224	678.08	68.91	62.55	47.69	23.77	9.84	3.69	22.2	5.45	43.1	3.40	46.1	6.80	65.5
SqueezeNet-V1.1 [5]	224×224	281.57	48.49	41.37	23.78	42.52	5.81	3.42	17.3	4.23	31.4	2.96	41.1	4.72	56.6
1.0-SqNxt-23 [22]	224×224	380.50	15.32	14.73	22.09	-50.00	24.84	22.35	9.4	24.22	15.7	18.41	33.1	22.34	42.0
1.0-SqNxt-23v5 [22]	224×224	242.04	16.01	15.02	15.30	-1.91	15.12	24.56	8.3	25.88	14.0	18.00	32.4	20.85	40.0
1.0-G-SqNxt-23 [22]	224×224	406.35	12.39	12.02	22.80	-89.61	32.80	24.27	9.3	25.81	15.3	18.80	33.4	22.22	42.3
MobileNet-V1 [6]	224×224	135.65	28.24	23.37	12.43	46.82	4.80	27.71	9.5	93.97	9.6	19.10	35.0	64.40	35.4
MobileNet-V2 [7]	224×224	124.80	12.36	11.24	8.71	22.52	10.10	42.90	9.5	140.11	10.4	30.30	34.7	98.23	35.3
XceptionNet [23]	299×299	366.58	84.81	68.87	35.29	48.76	4.32	124.92	11.4	468.16	11.5	87.20	38.3	324.36	38.5

Also, the disparity factor (d_f), activation to parameter ratio ($\frac{A}{W}$), along with average power (P_{avg} in watts) and inference time (I_t in milliseconds) measured for B=1 and B=4 on P4000 and P100 GPUs.

TABLE 8
Disparity Between AI_c and DI for Different Cases

	Case 1: $A \ll W$	Case 2: $A \approx W$	Case 3: $A \gg W$
M_c/A	larger	comparable	smaller
M_c/W	smaller	comparable	larger
AI_c	$\approx M_c/W$	$\approx 0.5 \times M_c/A$	$\approx M_c/A$
DI	$\approx 0.2 \times M_c/A$	$\approx 0.25 \times M_c/A$	$\approx 0.06 \times M_c/W$
d_f	$\approx 75 - 18.75 \times \frac{W}{A}$	≈ 50	$\approx 75 - 6.25 \times \frac{A}{W}$

learning. Similarly, Wang *et al.* [37] automate the process of layer-wise quantization using reinforcement learning. Furthermore, there is an increasing trend in automating the design of compact and high-performance DNNs [38].

The works mentioned above on automating the design process optimize the DNN architecture to reduce the number of MAC operations. Since these metrics are only a proxy for energy consumption, optimizing them does not necessarily optimize the energy efficiency of DNNs. To design energy-efficient DNNs, a design-time metric that is a better representative of energy-efficiency of DNNs is required. Such a metric can be used in the objective function of AutoML tasks. Since data movements primarily drive energy consumption, data reuse can be used as the approximation for memory accesses ($\frac{1}{DI}$). Hence, instead of using #MACs for the energy-efficient design of DNNs, the following metric can be used in the objective functions in the AutoML tasks.

$$\text{Optimized metric} \propto M_c \times \left(\frac{1}{DI}\right)^k s.t.\ k \in (0,1). \quad (7)$$

Since the cost of memory access is orders of magnitude higher than arithmetic operation, k is used to normalize the memory cost with respect to computational cost in Eq. (7). Consequently, the above optimized metric balances the number of computations and the number of memory access to optimize the network's energy efficiency. Evidently, DI will be a valuable tool for DNN designers.

7 VALIDITY OF PROPOSED MODEL

Our model estimates the data reuse available in DNNs with the assumption that underlying platforms have sufficient compute/memory resources to exploit the available data reuse in DNNs. However, different hardware platforms are optimized for contrasting design goals and have dissimilar memory-hierarchy with a non-identical number of layers and capacity. We now discuss whether our model applies to general-purpose hardware such as GPU and CPU, or do

TABLE 9
Confidence Intervals for Population Correlation (ρ)

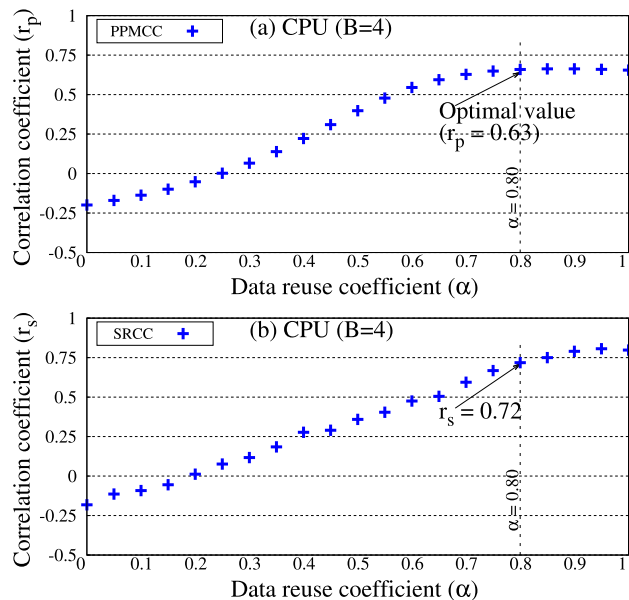| GPU	Metric	95% confidence			99% confidence		
		L	U	Δ	L	U	Δ
P4000	AI_c	0.42	0.86	0.44	0.31	0.89	0.58
	DI (**Ours**)	0.68	0.93	**0.25**	0.61	0.95	**0.34**
P100	AI_c	0.36	0.84	0.48	0.24	0.87	0.63
	DI (**Ours**)	0.70	0.94	**0.24**	0.63	0.95	**0.32**

Fig. 7. PPMCC (a) and SRCC (b) calculated between weighted arithmetic intensity at different values of α, and energy efficiency measured for 25 DNNs on CPU with B=4.

we need to re-calibrate the value of α on them? We also discuss the effect of memory-hierarchy on our model.

GPU. Both P100 and P4000 GPUs have substantially different memory and compute capability, which is also manifested by different CMR values (Table 4). Hence, the memory-access pattern and their cost would vary significantly for both the GPU. In Fig. 2a, it is shown that EPP values of DNNs are higher on P4000 GPU compared to that on P100 GPU. Our extensive experiments validate the proposed model and hence do not require re-calibration of α for both the GPUs, even though they have different compute and memory resources. This indicates the applicability of our model to different GPUs regardless of their memory-hierarchy and CMR values. However, one may need to re-calibrate the α to use DI as a representative of the energy efficiency of other DNN models that are not used in our experiments.

CPU. CPUs have hardware managed cache hierarchy along with sophisticated techniques for cache miss management and cache coherence. Also, CPUs have a higher amount of off-chip memory than GPUs, which can accommodate a larger model with larger batch size. Furthermore, CPUs have a much lower amount of parallelism than GPUs. We investigate whether the above-mentioned differences affect our model. We perform our experiments on "Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60 GHz" which has 12 cores, 64 KB L1 cache, 256 KB L2 cache, 15 MB L3 cache, and 32 GB primary memory. The correlation results are shown in Fig. 7. Similar to the results on P100 and P4000 GPU, both r_p and r_s increase with rising value of α and r_p saturates at 0.8. This substantiates the higher importance of activation reuse for estimating the available data reuse. Therefore, the correlation trends would be similar (PPMCC first increases with α and later saturates at higher α) even for the DNNs not used in our experiments. However, the value of α may need to be re-calibrated for the precise use of DI as a representative of the energy-efficiency of other DNN models that are not used in our experiments, on the CPU.
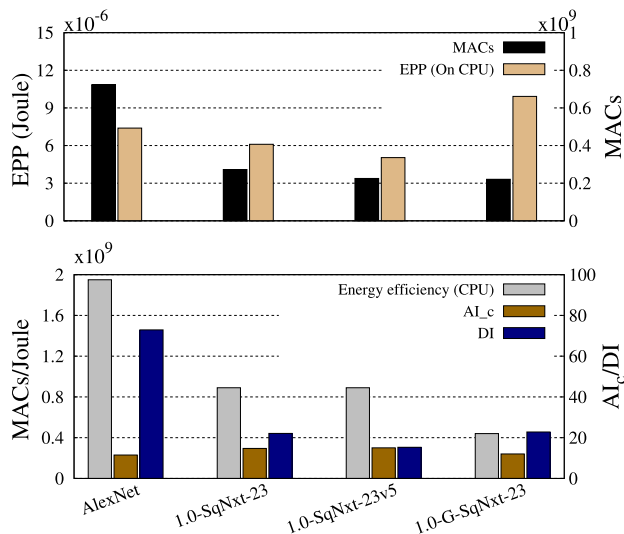
Fig. 8. Energy consumption (EPP) and energy-efficiency of AlexNet and SqueezeNext variants measured on CPU.

For a better comparison of our model with AI on CPU, we select AlexNet and three SqueezeNext variants, which have a very high disparity between A and W (Table 7). The EPP, energy efficiency, AI_c, and DI are shown in Fig. 8. Also, the calculated r_p and r_s with AI_c and DI are shown in Table 10. The negative correlations (r_p and r_s) with AI show the ineffectiveness of AI as a representative of energy-efficiency on CPU, whereas, high positive correlations with DI indicate DI as an appropriate representative of energy-efficiency on CPU.

8 RELATED WORK

Yang et al. [14] proposed an energy estimation tool which takes the layer's dimensions and sparsity in DNN as inputs and estimates the energy consumption of a DNN. They validated their energy model on a systolic array-based processor (Eyeriss [39]) with very few DNNs (AlexNet, VGG-16, GoogLeNet, and SqueezeNet). Chen et al. [40] employed a model named "Eyexam", which takes design decisions of both the DNN model and systolic accelerator as input and predicts their effects on the performance. Kwon et al. [41] proposed an analytical cost model named "MAESTRO", which takes DNN's layer dimensions and the employed dataflow as inputs and provides detailed performance analysis of DNNs on systolic accelerators. Cai et al. [42] build a model that predicts the latency for latency-aware DNN design using an architecture search. Li et al. [43] performed a detailed study on the energy-efficiency of DNNs on CPU and GPU and provides insights for the energy-aware design of DNNs. The works mentioned above demonstrate their effectiveness for very few DNNs, far from comprehensive, on a limited set of hardware platforms. They do not include state-of-the-art design methodologies, such as dense connections in DenseNets, where the number of activations varies at runtime. Hence, their models' applicability is quite limited and cannot be generalized over a wide range of DNNs. Nevertheless, Binaco et al. [44] have done extensive experimentation and presented the detailed comparison of the inference time of the SOTA DNNs on a multitude of

TABLE 10
Correlation (r_p, r_s) of AI_c and DI With Energy Efficiency of AlexNet and SqueezeNext Variants on CPU

CPU	X	Y	correlation (X, Y)
PPMCC (r_p)	AI_c	Energy efficiency	-0.39
	DI (**Ours**)	Energy efficiency	**0.91**
SRCC (r_s)	AI_c	Energy efficiency	-0.37
	DI(**Ours**)	Energy efficiency	**0.76**

GPUs. However, due to the missing energy comparison and the lack of an appropriate metric and/or mathematical model that can predict the inference time and/or energy efficiency of DNNs at the design time, the insights for data reuse in the representative DNNs is missing.

9 CONCLUSION AND FUTURE WORK

We show that the conventional metric (AI) does not always accurately estimate the degree of data reuse in DNNs. We propose a novel model that decouples the weight and activation reuse and accounts for their unequal importance. We show that our model applies to a diverse set of DNNs and is a better representative of energy efficiency in DNNs. In future work, we will evaluate our model on other accelerators to show its applicability over a broad range of hardware platforms.

ACKNOWLEDGMENTS

REFERENCES

[1] A. Krizhevsky et al., "ImageNet classification with deep convolutional neural networks," in Proc. 25th Int. Conf. Neural Inf. Process. Syst., 2012, pp. 1097–1105.
[2] K. Simonyan et al., "Very deep convolutional networks for large-scale image recognition," 2014, arXiv:1409.1556.
[3] C. Szegedy et al., "Inception-v4, inception-ResNet and the impact of residual connections on learning," in Proc. 31st AAAI Conf. Artif. Intell., 2017, pp. 4278–4284.
[4] K. He et al., "Identity mappings in deep residual networks," in Proc. Eur. Conf. Comput. Vis., 2016, pp. 630–645.
[5] F. N. Iandola et al., "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5mb model size," 2016, arXiv:1602.07360.
[6] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, arXiv:1704.04861.
[7] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 4510–4520.
[8] N. Ma et al., "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in Proc. Eur. Conf. Comput. Vis., 2018, pp. 122–138.
[9] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in Proc. IEEE Int. Solid-State Circuits Conf. Digest Tech. Papers, 2014, pp. 10–14.
[10] M. Harris, "Mapping computational concepts to GPUs," in Proc. ACM SIGGRAPH Courses, 2005, pp. 50–es.
[11] S. Williams et al., "Roofline: An insightful visual performance model for multicore architectures," Commun. ACM, vol. 52, no. 4, pp. 65–76, 2009.
[12] J. W. Choi, D. Bedard, R. Fowler, and R. Vuduc, "A roofline model of energy," in Proc. IEEE 27th Int. Symp. Parallel Distrib. Process., 2013, pp. 661–672.
[13] M. Ghane et al., "Power and energy-efficiency roofline model for GPUs," 2018, arXiv:1809.09206.

[14] T.-J. Yang, Y.-H. Chen, J. Emer, and V. Sze, "A method to estimate the energy consumption of deep neural networks," in *Proc. 51st Asilomar Conf. Signals Syst. Comput.*, 2017, pp. 1916–1920.

[15] J. Park *et al.*, "Deep learning inference in Facebook data centers: Characterization, performance optimizations and hardware implications," 2018, *arXiv: 1811.09886*.

[16] B. H. Ahn *et al.*, "Ordering chaos: Memory-aware scheduling of irregularly wired neural networks for edge devices," in *Proc. Mach. Learn. Syst.*, 2020, pp. 44–57.

[17] L. Lai *et al.*, "Not all ops are created equal!" *SysML*, 2018, *arXiv:1801.04326*.

[18] M. Lin *et al.*, "Network in network," 2013, *arXiv:1312.4400*.

[19] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.

[21] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5987–5995.

[22] A. Gholami *et al.*, "SqueezeNext: Hardware-aware neural network design," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 1719–171909.

[23] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1800–1807.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[25] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.

[26] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.

[27] R. A. Bridges *et al.*, "Understanding GPU power: A survey of profiling, modeling, and simulation methods," *ACM Comput. Surv.*, vol. 49, no. 3, pp. 1–27, 2016.

[28] K. Kasichayanula, D. Terpstra, P. Luszczek, S. Tomov, S. Moore, and G. D. Peterson, "Power aware computing on GPUs," in *Proc. Symp. Appl. Accelerators High Perform. Comput.*, 2012, pp. 64–73.

[29] B. Wu, A. Wan, F. Iandola, P. H. Jin, and K. Keutzer, "SqueezeDet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2017, pp. 129–137.

[30] M. M. Mukaka, "A guide to appropriate use of correlation coefficient in medical research," *Malawi Med. J.*, vol. 24, no. 3, pp. 69–71, 2012.

[31] J. Hauke *et al.*, "Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data," *Quaestiones Geographicae*, vol. 30, no. 2, pp. 87–93, 2011.

[32] L. Wang *et al.*, "SuperNeurons: Dynamic GPU memory management for training deep neural networks," in *Proc. Symp. Princ. Pract. Parallel Program.*, 2018, pp. 41–53.

[33] N. K. Jha, S. Mittal, and G. Mattela, "The ramifications of making deep neural networks compact," in *Proc. 32nd Int. Conf. VLSI Des. and 18th Int. Conf. Embedded Syst.*, 2019, pp. 215–220.

[34] Y. Wang *et al.*, "A systematic methodology for analysis of deep learning hardware and software platforms," in *Proc. Mach. Learn. Syst.*, vol. 2020, pp. 30–43, 2020.

[35] V. Bewick *et al.*, "Statistics review 7: Correlation and regressions," *Crit. Care*, vol. 7, pp. 451–459, 2003.

[36] Y. He *et al.*, "AMC: AutoML for model compression and acceleration on mobile devices," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 815–832.

[37] K. Wang, Z. Liu, Y. Lin, J. Lin, and S. Han, "HAQ: Hardware-aware automated quantization with mixed precision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8604–8612.

[38] M. Tan *et al.*, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[39] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 53, no. 1, pp. 127–138, Jan. 2017.

[40] Y.-H. Chen, T.-J. Yang, J. Emer, and V. Sze, "Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 2, pp. 292–308, Jun. 2019.

[41] H. Kwon, P. Chatarasi, M. Pellauer, A. Parashar, V. Sarkar, and T. Krishna, "Understanding reuse, performance, and hardware cost of DNN dataflow: A data-centric approach," in *Proc. 52nd Annu. IEEE/ACM Int. Symp. Microarchitecture*, 2019, pp. 754–768.

[42] H. Cai *et al.*, "ProxylessNAS: Direct neural architecture search on target task and hardware," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=HylVB3AqYm

[43] D. Li, X. Chen, M. Becchi, and Z. Zong, "Evaluating the energy efficiency of deep convolutional neural networks on CPUs and GPUs," in *Proc. IEEE Int. Conf. Big Data Cloud Comput., Soc. Comput. Netw., Sustain. Comput. Commun.*, 2016, pp. 477–484.

[44] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark analysis of representative deep neural network architectures," *IEEE Access*, vol. 6, pp. 64 270–64 277, 2018.

Nandan Kumar Jha received the BTech degree in ECE from NIT, Surat, India, in 2013. He is working toward the MTech (Research Assistant) degree in the CSE at IIT, Hyderabad, India. He has worked as an electrical design engineer in Seagate Technology, India and as a project research assistant at IIT, Bombay, India. He research interests are deep learning and computer architecture.

Sparsh Mittal received the BTech degree in ECE from IIT, Roorkee, India and the PhD degree in computer engineering from Iowa State University, Ames, Iowa. He has worked as a post-doctoral research associate at Oak Ridge National Lab, Oak Ridge, Tennessee and as an assistant professor at IIT Hyderabad, India. He is currently working as an assistant professor at IIT Roorkee, India. He has published nearly 90 papers in top conferences and journals.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.