ADDERIC: TOWARDS LOW COMPUTATION COST IMAGE COMPRESSION

Bowen Li[1], *Yao Xin*[2], *Chao Li*[1], *Youneng Bao*[1], *Fanyang Meng*[2], *Yongsheng Liang*[1*]

[1]Harbin Institute of Technology, Shenzhen, China [2]Peng Cheng Laboratory, Shenzhen, China

ABSTRACT

Recently, learned image compression methods have shown their outstanding rate-distortion performance when compared to traditional frameworks. Although numerous progress has been made in learned image compression, the computation cost is still at a high level. To address this problem, we propose AdderIC, which utilizes adder neural networks (AdderNet) to construct an image compression framework. According to the characteristics of image compression, we introduce several strategies to improve the performance of AdderNet in this field. Specifically, Haar Wavelet Transform is adopted to make AdderIC learn high-frequency information efficiently. In addition, implicit deconvolution with the kernel size of 1 is applied after each adder layer to reduce spatial redundancies. Moreover, we develop a novel Adder-ID-PixelShuffle cascade upsampling structure to remove checkerboard artifacts. Experiments demonstrate that our AdderIC model can largely outperform conventional AdderNet when applied in image compression and achieve comparable rate-distortion performance to that of its CNN baseline with about 80% multiplication FLOPs and 30% energy consumption reduction.

Index Terms— Image compression, AdderNet, energy consumption

1. INTRODUCTION

Image compression is one of the most essential tasks in both signal processing and computer vision fields. Recently, with the rapid development of deep learning, learned image compression methods [1, 2, 3, 4] have drawn much attention and shown their excellent performance when compared to traditional frameworks [5, 6, 7]. Although numerous progress has been made in learned image compression, the recently proposed models still maintain high computation complexity, being not conducive to deployment on devices with limited resources, e.g., mobile phones and embedded devices that have extensive needs of image compression.

To address this problem, several methods have been developed. One of the most popular approaches is pruning [8], which aims to remove redundant weights to accelerate the original networks. Another important approach is knowledge distillation [9], which transfers knowledge from a large model to a smaller model without loss of validity. However, these compressed models still contain massive multiplication operations and consume enormous computation resources. Although some model quantization methods [10, 11, 12] can save massive computation resources, they cannot maintain the comparable performance of their original networks. Recently, Chen *et al.* [13] pioneered AdderNet, which utilizes $L1$ norm to calculate the similarities between inputs and filters, to replace CNN. In other words, it avoids the multiplication operations and largely reduces the computation cost without largely accuracy drop. Thus, AdderNet is suitable for deployment to computing-constrained devices.

At present, AdderNet has been successfully applied to image classification [13, 14] and single image super-resolution [15], which shows performance comparable to their CNN baselines. However, there is no work to construct an image compression network using AdderNet due to the following reasons. First, image compression aims to learn different frequency information according to different bitrates, but the adder layer cannot learn high-frequency efficiently [13]. Second, the batch normalization (BN) layers [16] used in conventional AdderNet are not beneficial to remove spatial redundancies [1]. Moreover, it is much easier to cause checkerboard artifacts [17] in the decoder because of the large output covariance [13] of AdderNet. To this end, we proposed AdderIC and develop three schemes to address

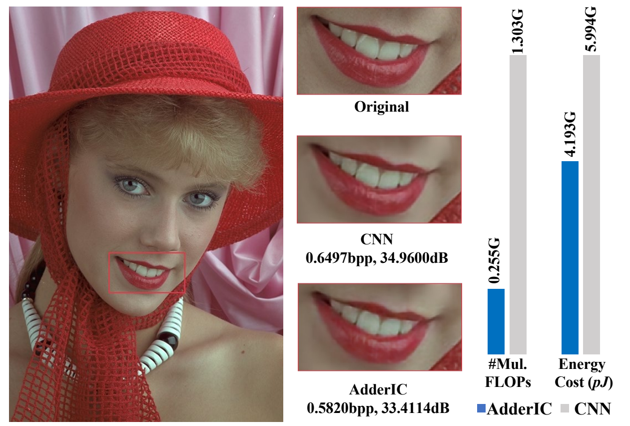


Fig. 1. Visualization of reconstruction quality and corresponding multiplication FLOPs and energy cost budget.

*Corresponding author, email: liangys@hit.edu.cn. This research was supported by the National Natural Science Foundation of China (Grant No.61871154No. 62031013), by the Youth Program of National Natural Science Foundation of China (61906103,61906124), by the Basic and applied basic research fund of Guangdong Province (2019A1515011307).

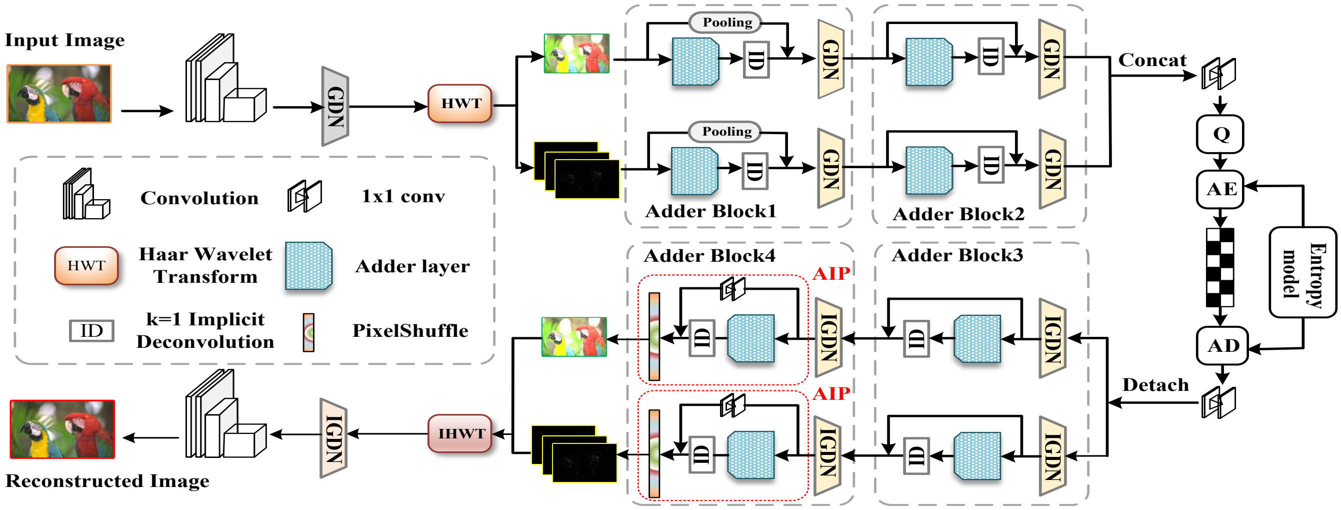


Fig. 2. The proposed AdderIC. AE and AD denote arithmetic encoder and arithmetic decoder, while Q is quantization.

above problems and our main contributions are as follows:

(1) To our knowledge, it is the first image compression framework based on adder neural networks, namely AdderIC.

(2) We introduce Haar Wavelet Transform (HWT) to enable AdderIC to learn high-frequency information efficiently. In addition, $k = 1$ implicit deconvolution is adopted to reduce spatial redundancies. Furthermore, we develop a novel Adder-ID-PixelShuffle (AIP) upsampling structure to remove checkerboard artifacts in the decoder.

(3) Experiments demonstrate that the proposed AdderIC model can greatly outperform traditional AdderNet when applied in image compression and achieve comparable performance to that of its CNN baseline with about 80% multiplication FLOPs and 30% energy consumption reduction.

2. PROPOSED METHOD

2.1. Learning High-Frequency Information Effectively

Image compression is required to learn different frequency information. However, it is hard for AdderNet to learn high-frequency information effectively [15]. To this end, our solution is to take advantage of Wavelet Transform to decompose an image into different frequency information and force AdderNet to learn. Furthermore, since this work aims at reducing computation cost, we choose Haar Wavelet Transform (HWT) [18], which is a lightweight wavelet that does not involve multiplication operations, into our AdderIC model. HWT is based on two functions as follows:

$$\varphi(t) = \begin{cases} 1 & 0 \le t < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

$$\psi(t) = \begin{cases} 1 & 0 \le t < \frac{1}{2}, \\ -1 & \frac{1}{2} \le t < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where $\varphi(t)$ and $\psi(t)$ denote Haar basis function and Haar wavelet function, respectively. Then, we define two vector spaces V^j and W^j, which are spanned by $\varphi(t)$ and $\psi(t)$ respectively, as follows:

$$V^j = \text{span}\left\{2^{\frac{j}{2}}\varphi\left(2^j t - i\right), i = 0, 1, \ldots, 2^j - 1\right\} \tag{3}$$

$$W^j = \text{span}\left\{2^{\frac{j}{2}}\psi\left(2^j t - i\right), i = 0, 1, \ldots, 2^j - 1\right\} \tag{4}$$

where i and j represent shift and scale factors, respectively. The relationship between the two vector spaces is derived:

$$V^{j+1} = V^j \oplus W^j \tag{5}$$

which means that each space V^{j+1} can be decomposed into lower-order space V^j and W^j. Moreover, V^j is an orthogonal complement of W^j and V^{j+1} can be further decomposed:

$$V^{j+1} = V^0 \oplus W^0 \oplus W^1 \oplus \ldots \oplus W^j \tag{6}$$

Thus, V^{j+1} is represented by a Haar basis function in V^0 and a series of Haar wavelet functions in W^i $(i = 0, 1, \ldots, j)$. In other words, HWT can decompose function values into one approximation coefficient and several detail coefficients, which refer to low-frequency and high-frequency information respectively. In AdderIC, first-level HWT is employed to decompose the image into A, H, V, D, where A is the approximation coefficient, and H, V, D are horizontal, vertical, and diagonal detail coefficients. Adder layers follow each coefficient to learn different frequency information.

2.2. Implicit Deconvolution

Image compression aims at reducing pixel-wise redundancies in images to improve performance. In the CNN-based frameworks [1, 2, 3], the convolutional layers can be used to reduce spatial redundancies. However, the BN layer in conventional AdderNet cannot meet this requirement and would result in performance degradation. Instead, inspired by Ye *et al.* [19], implicit deconvolution (ID) with kernel size 1 ($k = 1$) is adopted after each adder layer to reduce pixel-wise redundancies. The core of this scheme is to calculate the covariance matrix Cov of input X as follows:

$$Cov = \frac{1}{N}(X-\mu)^T(X-\mu) \tag{7}$$

where N denotes the number of samples and μ is the mean value of X. Then, the network deconvolution operation D can be represented as the inverse square root of Cov:

$$D = Cov^{-\frac{1}{2}} \tag{8}$$

Note that coupled Newton-Schulz iteration approach is used to reduce the computation complexity of deconvolution. Once deconvolution matrix D is obtained, we can multiply each centered input matrix $(X-\mu)$ by it to reduce spatial redundancies. Furthermore, the expression of ID can be derived:

$$y = (X-\mu)\cdot D\cdot W = X\cdot(D\cdot W) - \mu\cdot D\cdot W \tag{9}$$

where W is ID weight and y is ID output. Although ID is adopted to replace the Convolution-BN (convolutional layer and BN layer cascade) structure in many computer vision tasks and has shown excellent performance, it cannot be directly used to replace the Adder-BN structure in AdderNet due to their different calculation paradigms. However, we can use $k=1$ ID to merely replace the BN layers in our network, which would not only prevent the gradients explosion but also reduce spatial redundancies. Although its computation complexity is a little higher than the BN layer, it is significantly lower than that of regular $k>1$ convolution layers.

2.3. AIP Upsampling Structure

In CNN-based frameworks [1, 2, 3], transposed convolution, which can be treated as a special case of regular convolution, is the most widely used operation to reconstruct the image. For Adder-based networks, although we could implement transposed adder layer according to the relationship between regular and transposed convolution, it is prone to cause artifacts as illustrated by the variance of adder layer output:

$$Var\left[Y_{Adder}\right] = C_1(Var[X] + Var[F]) \tag{10}$$

where C_1 and F denote the simplified coefficient and adder weight. As stated in [13], the output variance of AdderNet is much larger than that of conventional CNNs, which could result in checkerboard artifacts. To this end, we propose a novel Adder-ID-PixelShuffle (AIP) upsampling structure to replace transposed convolution in the decoder. In this architecture, we employ $k=1$ ID after each adder layer with stride 1 and increase the number of output channels to the square of the upsampling factor. At last, the pixelshuffle layer is utilized to upsample the feature maps. Consequently, the output variance of the proposed AIP structure can be derived:

$$Var\left[Y_{AIP}\right] = C_2\{(Var[X] + Var[F])\, Var[W]\}\, Var[P] \tag{11}$$

where P and C_2 represent pixelshuffle and another simplified coefficient, respectively. Note that $Var[P]$ can be treated as one. If the variance of the weight W in ID is small, the overall variance will be reduced to a level similar to that of CNN, which means that it can help reduce the checkerboard artifacts. Thus, the proposed AIP structure is effective for AdderNet and can be used in other vision fields.

(a) Convolution (b) Adder-BN (c) Adder-ID

Fig. 3. Visualization of latent codes and redundancies.

2.4. Overall Structure

As shown in Fig. 2, the AdderIC is based on a classical CNN-based architecture [1], which consists of analysis transform and synthesis transform. Following the setting of traditional AdderNet, we maintain the first convolution layer and the last transposed convolution layer, and employ $k=1$ ID instead of the BN after each adder layer. The HWT after convolution decomposes the feature maps into four branches, and each branch can extract features in distinct frequency. Then the feature maps from four branches are concatenated in channel dimension and then do $k=1$ convolution to reduce the number of the channel to 128. After quantization and entropy coding, the feature maps are restored to 512 channels with another $k=1$ convolution and then resolved to four branches each followed by adder blocks. Moreover, different kinds of residual connections are supplemented in different adder blocks to enable AdderIC to learn identity mapping [15].

3. EXPERIMENTS

3.1. Experiment Setup

The AdderIC model is trained and optimized for MSE by using different λ values (*i.e.* 64, 128, 256, 512, 1024, 2048) on the CLIC training dataset [20] which consists of 61894 images with 256×256 pixels, and evaluated on the standard Kodak dataset [21] with 24 images of 512×768 or 768×512 pixels and CLIC test dataset [20] with 15160 images of 256×256 pixels. Specifically, we first train the models with $\lambda = 2048$ for 400 epochs with a batch size of 16 using Adam optimizer, and the initial learning rates of the adder layer and the convolution layer are 1×10^{-3} and 1×10^{-4}, respectively. After 336 epochs, the learning rates decay to one-tenth of their original value. For other bitrates, we adopt the model trained on high bitrate ($\lambda = 2048$) as a pre-trained model and then

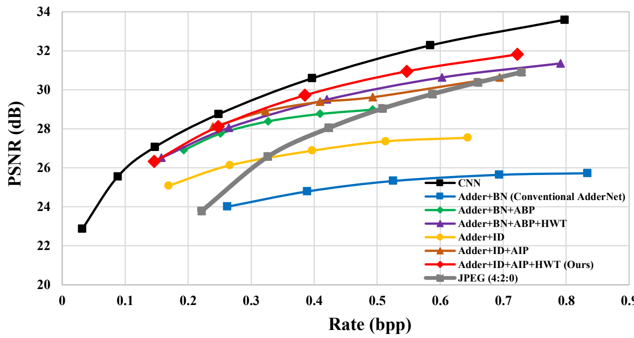


Fig. 4. PSNR on Kodak Dataset

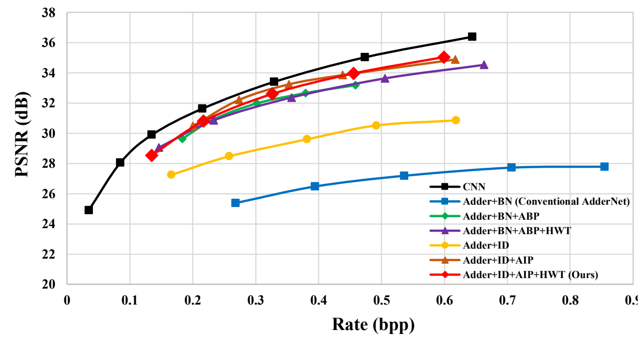


Fig. 5. PSNR on CLIC Test Dataset

Table 1. #Mul. FLOPs and energy cost of different networks.

Model	CNN	AdderIC	Reduction
Multiplication FLOPs	1.303G	0.255G	80.43%
Energy Cost (pJ)	5.994G	4.193G	30.05%

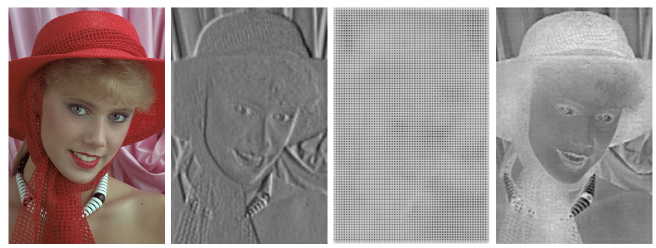
fine-tune other models for 100 epochs with a learning rate of 1×10^{-4}. Other settings are similar to the CNN counterpart [1]. The performance metrics for the following evaluations include efficiency in reducing redundancies, reconstruction quality, average bit-per-pixel (bpp), average peak signal-to-noise (PSNR), computational cost and energy consumption.

3.2. Performance Comparison

To verify the effectiveness of ID for reducing spatial redundancies, we visualize the latent codes and required bits of different architectures, including convolution, Adder-BN and the proposed Adder-ID. From Fig. 3, it is obvious that the conventional Adder-BN cannot effectively reduce spatial redundancies, while the proposed Adder-ID is able to efficiently capture useful information and significantly reduce pixel-wise redundancies, which is similar to the convolution layer.

To confirm the efficiency of AIP structure, we visualize the output feature maps of different upsampling structures in Fig. 6. The results show that the transposed adder layer can easily cause checkerboard artifacts, which would lead to poor reconstruction quality. By contrast, the proposed AIP structure as well as the transposed convolution can avoid them and maintain the details of the original images effectively.

At last, we compare the proposed AdderIC with its CNN counterpart in aspects of rate-distortion performance, along with computation cost and energy consumption. From Fig. 1, Fig. 4 and Fig. 5, we can see that our AdderIC model shows comparable performance to CNN-based architecture, and is better than other AdderNet structures. Besides, we omit the ID and $k = 1$ convolution due to their low computation cost, and compare the multiplication FLOPs and energy cost between AdderIC and its CNN counterpart in Table 1. The energy cost is computed according to literature [22]. The results show that AdderIC reduces the multiplication FLOPs by approximately 80% and energy consumption by 30%.



(a) (b) (c) (d)

Fig. 6. Visualization for checkerboard artifacts comparison. (a) is the input image, while (b), (c) and (d) denote output feature maps of the transposed convolution layer, the transposed adder layer and the proposed AIP structure, respectively.

3.3. Ablation Studies

To verify the effectiveness of the proposed three schemes in Sec. 2, several ablation experiments are conducted as shown in Fig. 4 and Fig. 5. The results show that directly replacing the convolution layer with the traditional Adder-BN structure will get a serious performance degradation, while the Adder-ID can efficiently remove spatial redundancies to improve the performance. Besides, using the proposed AIP structure instead of the transposed adder can reduce checkerboard artifacts, which helps to enhance the reconstruction quality and further improve the performance. In addition, AdderIC can effectively learn high-frequency information by introducing HWT, which is beneficial to preserve the details in images. In summation, the combination of these three methods can bring satisfactory rate-distortion performance to AdderIC.

4. CONCLUSION

In this paper, we developed three improvement schemes for the proposed AdderIC, which is the first image compression framework based on AdderNet. Experiments demonstrate that AdderIC can largely outperform conventional AdderNet when applied in image compression and achieve comparable performance to that of its CNN baseline with about 80% multiplication FLOPs and 30% energy consumption reduction. Future works include better training strategies to further improve the performance of AdderIC.

5. REFERENCES

[1] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli, "End-to-end optimized image compression," in *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[2] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston, "Variational image compression with a scale hyperprior," *arXiv preprint arXiv:1802.01436*, 2018.

[3] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 7936–7945, 2020.

[4] George Toderici, Sean M. O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar, "Variable rate image compression with recurrent neural networks," in *4th International Conference on Learning Representations, ICLR 2016*, 2016.

[5] Michael W. Marcellin, Ali Bilgin, Michael J. Gormish, and Martin P. Boliek, "An overview of jpeg-2000," in *Proceedings of the Conference on Data Compression*, USA, 2000, p. 523.

[6] Majid Rabbani and Rajan Joshi, "An overview of the JPEG 2000 still image compression standard," *Signal Processing: Image Communication*, vol. 17, no. 1, pp. 3–48, jan 2002.

[7] F. Bellard, "BPG image format," Apr. 2018. [Online]. Available: https://bellard.org/bpg/.

[8] Huizi Mao, Song Han, Jeff Pool, Wenshuo Li, Xingyu Liu, Yu Wang, and William J Dally, "Exploring the granularity of sparsity in convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 13–20.

[9] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the Knowledge in a Neural Network," *arXiv preprint arXiv:1503.02531*, 2015.

[10] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio, "Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1," *arXiv preprint arXiv:1602.02830*, 2016.

[11] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi, "XNOR-net: Imagenet classification using binary convolutional neural networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9908 LNCS, pp. 525–542, 2016.

[12] Fengfu Li, Bo Zhang, and Bin Liu, "Ternary weight networks," *arXiv preprint arXiv:1605.04711*, 2016.

[13] Hanting Chen, Yunhe Wang, Chunjing Xu, Boxin Shi, Chao Xu, Qi Tian, and Chang Xu, "Addernet: Do we really need multiplications in deep learning?," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1468–1477.

[14] Yixing Xu, Chang Xu, Xinghao Chen, Wei Zhang, Chunjing Xu, and Yunhe Wang, "Kernel based progressive distillation for adder neural networks," *arXiv preprint arXiv:2009.13044*, 2020.

[15] Dehua Song, Yunhe Wang, Hanting Chen, Chang Xu, Chunjing Xu, and Da Cheng Tao, "AdderSR: Towards energy efficient image super-resolution," *arXiv preprint arXiv:2009.08891*, 2020.

[16] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[17] Augustus Odena, Vincent Dumoulin, and Chris Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, pp. e3, 2016.

[18] Radomir S. Stanković and Bogdan J. Falkowski, "The haar wavelet transform: its status and achievements," *Computers Electrical Engineering*, vol. 29, no. 1, pp. 25–44, 2003.

[19] Chengxi Ye, Matthew Evanusa, Hua He, Anton Mitrokhin, Tom Goldstein, James A. Yorke, Cornelia Fermüller, and Yiannis Aloimonos, "Network Deconvolution," *arXiv preprint arXiv:1905.11926*, 2019.

[20] Toderici George, Shi Wenzhe, Timofte Radu, Theis Lucas, Balle Johannes, Agustsson Eirikur, Nick Johnston, and Mentzer Fabian, "Workshop and challenge on learned image compression (clic2020)," 2020.

[21] Rich Franzen, "Kodak lossless true color image suite (photocd pcd0992)," http://r0k.us/graphics/kodak/, 1999.

[22] William Dally, "High-performance hardware for machine learning," *NIPS Tutorial*, 2015.