VisualNet: An End-to-End Human Visual System Inspired Framework to Reduce Inference Latency of Deep Neural Networks

Tianchen Wang, *Student Member, IEEE*, Jiawei Zhang, Jinjun Xiong, *Member, IEEE*, Song Bian, *Member, IEEE*, Zheyu Yan, Meiping Huang, Jian Zhuang, Takashi Sato, *Member, IEEE*, Xiaowei Xu, and Yiyu Shi, *Senior Member, IEEE*

Abstract—Acceleration of deep neural network (DNN) inference has gained increasing attention recently with the wide adoption of DNNs for practical applications. For computer vision tasks where inputs are images, existing works mostly focus on improving the throughput of inference for multiple images. However, in many real-time applications, it is critical to reduce the latency of a single image inference, which is more complicated than improving the throughput because of the inherent data dependencies. On the other hand, from human brain's perspective, the complexity in our visual surroundings is first encoded as a pattern of light on a two dimensional array of photoreceptors, with little direct resemblance to the original input or the ultimate percept. Within just a few hundred microns of retinal thickness, this initial signal encoded by our photoreceptors must be transformed into an adequate representation of the entire visual scene. Inspired by how the retina helps human brain incept new information efficiently, we present an end-to-end structured framework built using any existing convolutional neural network (CNN) as the backbone. The proposed framework, called VisualNet, can create task parallelism for the backbone during the inference of a single image. Experiments using a number of neural networks for the ImageNet classification task and the CIFAR-10 classification task on GPUs and CPUs show that the proposed VisualNet reduces the latency of the regular network it builds on by up to 80.6% when both are fully parallelized with state-of-the-art acceleration libraries. At the same time, VisualNet can achieve similar or slightly higher accuracy.

Index Terms—Biologically inspired, computer vision, neural network, human visual system

1 Introduction

In pursuit of higher accuracy, state-of-the-art neural networks (NNs) have become increasingly deeper, resulting in higher computational complexity and longer inference latency [1]. In contrast, many real-time applications, such as autonomous driving call for low inference latency while retaining accuracy [2], [3]. Various techniques have been explored to address this challenge, such as network compression and quantization [4], lightweight networks targeting resource-constrained platforms [5], [6], dynamic computation graphs that provide efficient early exits [7], and, on top of all the above techniques, network parallelization [8], [9], [10].

A large body of works exists on NN parallelization, most of which focus on improving the efficiency of the training process [9], [11], and do not apply to network inference. There are also a few works that explore parallelism [12] to increase the number of images inferred per unit time, i.e., the throughput, by distributing a batch of images to multiple cores. None of these works, however, helps to reduce the single-image inference latency, which is critical for real-time applications. Existing techniques for reducing the latency of single-image inference include operator parallelism [8], [9] and model parallelism [10]. The former technique explores concurrency in operators such as convolution, and the latter distributes kernels of convolutional layers across multiple cores. Due to the inherent data dependency of DNNs, these approaches do not offer good scalability and usually cannot fully utilize a large number of cores available in modern high-performance computing platforms.

On the other hand, the huge success of DNNs largely relies on its emulation of how human brain process information [13]. Let us have a more detailed understanding of how human visual system works. As shown in Fig. 1, the first stage of processing an image in the brain starts with projecting the two-dimensional scene on the retina layer through the optical lens to perform the first stages of image processing [14]. After the retina layer, which is made of

- *Tianchen Wang, Zheyu Yan, and Yiyu Shi are with the University of Notre Dame, Notre Dame, IN 46556 USA. E-mail: {twang9, zyan2, yshi4}@nd.edu.*
- *Jiawei Zhang, Meiping Huang, Jian Zhuang, and Xiaowei Xu are with Guangdong Academy of Medical Sciences, Guangzhou, Guangdong 510080, China. E-mail: 17110240008@fudan.edu.cn, huangmeiping@126.com, zhuangjian5413@tom.com, xiao.wei.xu@foxmail.com.*
- *Jinjun Xiong is with the State University of New York at Buffalo, Buffalo, NY 14260 USA. E-mail: jinjun@buffalo.edu.*
- *Song Bian and Takashi Sato are with Kyoto University, Kyoto 606-8501, Japan. E-mail: sbian@easter.kuee.kyoto-u.ac.jp, takashi@i.kyoto-u.ac.jp.*

Manuscript received 1 November 2021; revised 4 May 2022; accepted 23 June 2022. Date of publication 4 July 2022; date of current version 10 October 2022.
This work was supported in part by the National key Research and Development Program of China under Grant 2018YFC1002600, in part by the Science and Technology Planning Project of Guangdong Province, China under Grant 2018B090944002, and 2019B020230003, in part by Guangdong Peak Project under Grants DFJH201802, and in part by the National Natural Science Foundation of China under Grant 62006050.
(Corresponding author: Xiaowei Xu.)
Digital Object Identifier no. 10.1109/TC.2022.3188211

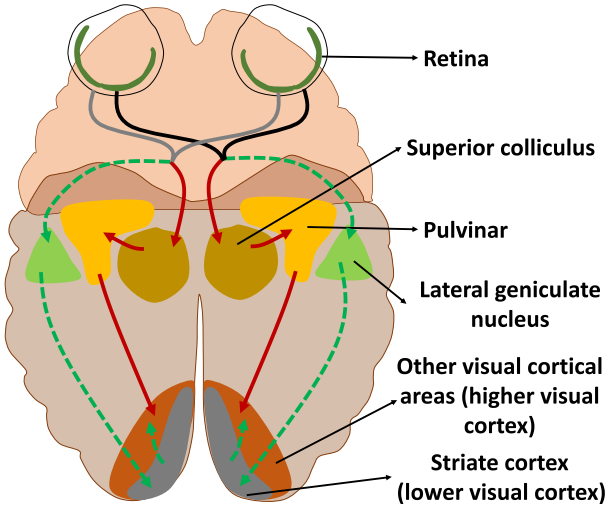


Fig. 1. Illustration of the human visual system which includes two paths, with red solid lines as tectopulvinar path and green dash lines as geniculostriate path, to pass the visual information to the higher visual cortex.

both the sensory neurons that respond to light and intricate neural circuits, the electrical message further travels down the optical nerves towards the visual cortex (aka, the brain) in two pathways as the geniculostriate pathway and the tectopulvinar pathway, to which we call internal info path and spatial info path, respectively. In the internal info path through the lateral geniculate nucleus (LGN), the various internal information of the image, such as motion, color, texture, and pattern, is detected by the multiple layers within LGN (four magnocellular layers and two parvocellular layers), and the signals are sent to the cortex thereafter. In the spatial info path through the superior colliculus (SC), the signals provide the viewer with more information on objects' absolute spatial information but not sensitive to fine details [15], [16]. Moreover, the SC receives the feedback signal resulting in eye movement for further visual details. A conception illustration of the above process is shown in Fig. 2a.

Some recent approaches mimic the above details with improved performance [17], [18], [19]. For example, Perceptnet [17] performed a series of perceptual operations in turn simulating the retina-LGN-V1 cortex pathway. Inspired by brain function, InterpoNet [18] investigated optical flow with sparse-to-dense interpolation and lateral dependency regulations. CNN-F [19] introduced a generative feedback with latent variables to existing CNN architectures, where consistent predictions are made through alternating maximum a posteriori inference under a Bayesian framework. However, all these methods just mimic parts of the overall system, but not the whole picture. Thus, a natural question occurs to us: is it possible to build a framework to mimic the structure and function of the human visual system that leads to a more efficient and faster approach to deal with visual tasks?

Inspired by the concept described above, we propose a generic efficient framework, named VisualNet (shown in Fig. 2b), that resembles the function of the human visual system and reduce inference latency. The analogies of the retina layer, the visual pathways, and the cortex in the visual system are explicitly incorporated in our VisualNet. Specifically, through a lightweight NN-based retina module, an input image is decomposed and split into two feature representations with smaller sizes, which are forwarded to the internal info path and spatial info path, respectively. The internal info path is designed as the main feature extractor that accounts for most of the computation, and we use a regular NN as the backbone in the path. The spatial info path is expected to generate the spatial feature guiding for more information towards better prediction, where we use independence and sparsity as the expected properties to regulate the learning. An additional intrinsic connection between two paths towards prediction accuracy is added as well. Then, the two output features from the paths are forwarded to the NN-based visual cortex module, where the features are mixed to produce the inference results.

The additional advantage of VisualNet is that it can be applied on top of all existing parallelization techniques to enhance the scalability further and reduce the an NN's inference latency. With the regulated independence in spatial info path and the connection with internal info path, the internal features can be processed in parallel by invoking multiple backbone instances. Meanwhile, as input to the backbone, the internal features are much smaller in dimension than the original input image. Accordingly, latency reduction can be achieved. Experimental results on ImageNet/CIFAR-10 classficiation tasks using various VisualNet-based NNs show that VisualNet provides up to 80.6% latency reduction while achieving similar or slightly higher accuracy.

2 Related Work

Various aspects targeting different levels of DNN inference are devised to reduce the inference latency of DNNs. The existing approaches can be classified into three large categories: 1). DNN structure, including novel components in

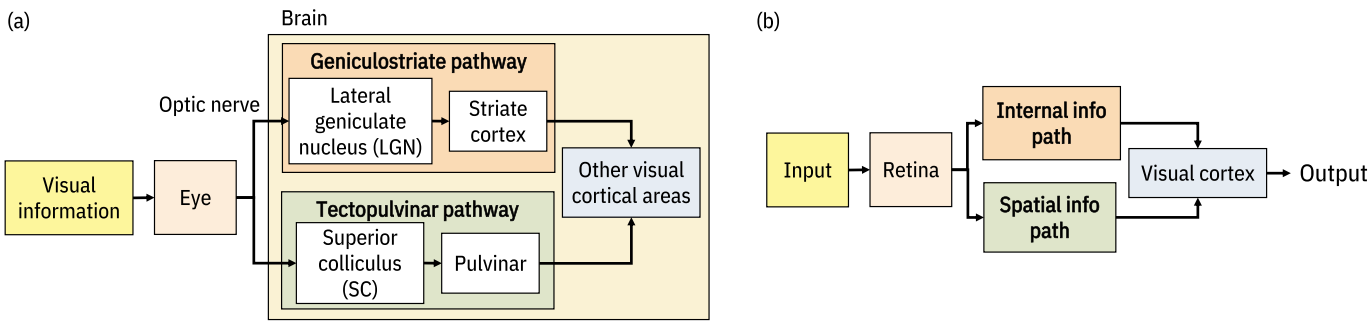


Fig. 2. Conceptual illustration of (a) the human visual system (drawn from the image [16]) and (b) VisualNet. Our VisualNet works in a similar manner as the human visual system.

DNN, novel DNN architecture search, and knowledge distillation; 2). DNN optimization, including computation operator optimization, parameter factorization, network pruning, and network quantization; 3). Hardware acceleration, including acceleration on various platforms of CPU/GPU/ASIC/FPGA, etc, and hardware level optimization such as building lookup table, reusing computation, and optimizing memory. In the rest of this section, we will go through the first category and the interested readers are recommended to go to [20], [21] for more details of DNN optimization and DNN hardware acceleration.

2.1 Efficient DNN Components

By designing an efficient DNN component, it means that by replacing or appending to the component or layer in classical DNN, such as VGG [22] which is made by a sequence of convolution-ReLU modules, the DNN can achieve higher performance and/or lower inference latency. We briefly describe the previously proposed efficient DNN components in the following.

Batch normalization [23] is a technique for training DNN that standardizes the inputs to a layer for each mini-batch of input data. One of the difficulties of training DNN is the distribution of the inputs to layers deep in the network may change after each mini-batch when the weights are updated, which causes the optimizer to chase a moving target forever. Batch normalization was introduced to address this change of distribution, which is also called the internal covariate shift. In modern DNNs, the batch normalization layers are often appended after the convolution layer (before or after the non-linear activation layer).

Separable convolution [24] mainly consists of two main types: spatially separable convolutions, and depthwise separable convolutions. The spatially separable convolutions refer to the ones that the convolutional kernels are decomposed into smaller kernels across their spatial axes, which resulting in fewer multiplications. However, the biggest limitation of the spatially separable convolutions is that only a minority of kernels is spatially separable, and training a DNN with spatially separable convolutions would limit the performance significantly. For depthwise separable convolutions, the two operations, depthwise convolutions and pointwise convolutions, are performed sequentially. In depthwise operation, the convolution is applied to a single channel at a time, which is different from the standard convolution in which the convolution is done for all the input channels. In pointwise operation, the 1×1 convolution operation is applied for all the input channels, which is usually used as channel-wise alignment among the layers.

Inception block [25] was proposed as a way of reducing computational expense. The most simplified version works by performing a convolution on an input with not one, but three different sizes of filters (1x1, 3x3, 5x5) as well as a max pooling operation. Then, the resulting outputs are concatenated and sent to the next layer. By structuring the CNN to perform its convolutions on the same level, the network gets progressively wider rather than deeper. An extra 1x1 convolution before 3x3 or 5x5 layers can be added as further reducing the required computation. By doing so, the number of input channels is limited and 1x1 convolutions are far cheaper than 5x5 convolutions. It is important to note, however, that the 1x1 convolution is added after the max-pooling layer, rather than before.

Residual block [26] is a stack of layers set where the output of a layer is taken and added to another layer deeper in the block, which was introduced as part of the ResNet architecture. The intuition is that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. Ideally, if the optimal case is an identity mapping, it would be easier to force the residual to zero than to fit an identity mapping by a stack of nonlinear layers. Accordingly, the skip connections allow the network to learn the identity mapping much easier.

MixConv [27] proposed a new mixed depthwise convolution (MixConv), which naturally mixes up multiple kernel sizes in a single convolution. The module acts as a simple drop-in replacement of vanilla depthwise convolution, which improves the accuracy and efficiency.

2.2 Lightweight CNNs

As the computation of convolution layers is expensive which limits the deployment of large CNNs with better accuracies on mobile platforms, many studies proposed new lightweight, mobile-friendly CNN structures/design guidelines in various aspects.

MobileNet series (V1 [28] V2 [6] V3 [29]) is a lightweight network family. Basically, MobileNet V1 proposed the streamlined architecture that uses depth wise separable convolutions, along with two global hyperparameters for latency/accuracy trade-off. MobileNet V2 proposed linear bottlenecks between the layers and the short connections between the bottlenecks to boost the performance. MobileNet V3 incorporated hardware-aware network architecture search (NAS) to search for the structure of each block targeting the best performance.

MNASNet [30] proposed an automated mobile neural architecture search approach, which explicitly incorporate model latency into the main objective so that the search can identify a model that achieves a good trade-off between accuracy and latency.

ShuffleNet V1 [5] V2 [31] is another lightweight network family. In ShuffleNet V1 two new operations, pointwise group convolution and channel shuffle, are proposed to greatly reduce computation cost while maintaining accuracy. ShuffleNet V2 further improved the architecture with minor modifications such as introducing channel split operation and moving channel shuffle to the end before the concatenation in both units.

MobileViT [32] is a light-weight and general-purpose vision transformer for mobile devices, which presents a different perspective for the global processing of information with transformers. MobileViT block replaces local processing in convolutions with global processing using transformers. It allows MobileViT block to have CNN- and ViT-like properties, which helps it learn better representations with fewer parameters and simple training recipes.

2.3 Knowledge Distillation

The goal of knowledge distillation is to build and train a DNN with simpler structure and less complexity from a large model, meanwhile achieving the similar performance.

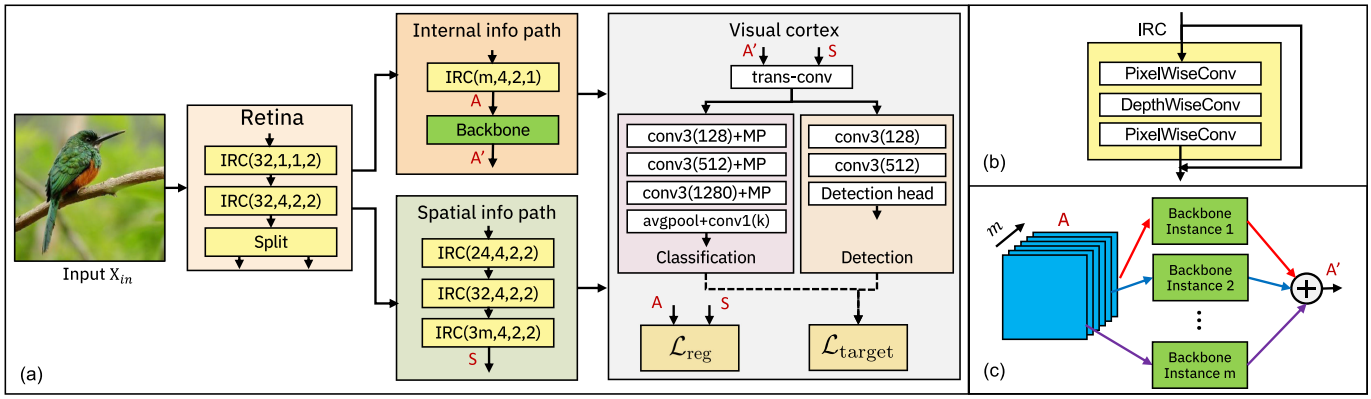


Fig. 3. (a) The illustration of VisualNet architecture. IRC(c, t, n, s) denotes the inverted residual cell with output channel c, expansion factor t, repeat times n, and stride s. k denotes the number of classes for classification. conv3/1(c) denotes the convolution cell (conv+bn+relu) with the kernel size 3x3/1x1 with output channel c and MP is size 2 max-pooling. m is the number of independent instances. (b) The structure of IRC. (c) The illustration of parallel propagation task parallelism by invoking multiple instances of the VisualNet backbone. The $\bigoplus$ denotes the concatenation operation.

Typically in knowledge distillation, a small student network is optimized to imitate a large teacher network. The existing studies mainly focus on perspectives of knowledge categories, training schemes, teacher-student architecture, and distillation algorithms.

Soft targets [33] was proposed for image classification task for the response-based knowledge, which usually refers to the neural response of the last output layer of the teacher model. As stated in [33], the soft targets are the probabilities that the input belongs to the classes, which contain the informative dark knowledge from the teacher model.

Online distillation strategies [34] were proposed as to overcome the limitation of the offline, static distillation [33], to further improve the performance of the student model, especially when a large-capacity high performance teacher model is not available. In online distillation, both models (teacher and student) are updated at the same time, and the whole framework is end-to-end trainable.

In this work, we will explore a method that extracts independent features from an input image, which creates task parallelism with little data dependency. Since task parallelism is orthogonal to operator or model parallelism, the proposed method can be combined with the existing techniques further to reduce the inference latency of a single image.

3 VisualNet

3.1 Intuition

We start with the intuition of VisualNet. As illustrated in Section 1 about the human visual system, after the retina layer (in human eyes) there are two pathways connected to it for further processing. The spatial info path through SC is used for extracting the absolute spatial information with less detail, where the SC also guides the eye movement for further information. The internal info path through LGN with multiple layers focuses more information such as motion, texture, color, etc. The signals from two pathways intersect at the visual cortex in the human brain for the final decision.

With the modules' functions and structures of the human visual system illustrated above, we have an interpretation of the visual system which could be connected to NN learning. Specifically, the retina module could be considered the initial information processing module to extract the low-level features for two pathways, where the two output features serve different purposes. The features for the spatial info path are more related to spatial information of the input image whose property we would like to enforce, which could be achieved by a lightweight NN with additional regulations. The features for the internal info path are more related to learning tasks and could be further fine-tuned through an NN backbone. As in the human visual system, the signals after spatial info path are for pursuing further information resulting in a better decision, thus in our design, the features extracted after the spatial path are expected to be high-level with the properties of independence and sparsity as proper guidance. On the other hand, we would like to build the intrinsic connection between the spatial and internal info paths, which could be achieved by enforcing reconstruction accuracy. With the enforcement of independence, sparsity, and reconstruction accuracy added, the features after spatial/internal info paths are then forwarded to the visual cortex, where the final output for various tasks can be obtained.

3.2 Architecture and Analysis

Based on the intuition above, we propose VisualNet to resemble the functions of the human visual system with the illustration shown in Fig. 3a. Our VisualNet comprises four modules: the retina module, the spatial path, the internal path, and the visual cortex module.

3.2.1 Structure of VisualNet

For the retina module, since it is designed as a lightweight NN to extract the initial representation and output two separate features for further processing, we construct it with two inverted residual cells (IRC) [6] (with two downsamplings) and a channel-wise split at the end of the module. IRC is adopted when constructing the retina layer is because compared with the normal residual block, IRC has a faster inference without compromising accuracy, which aligns with the need of retina layer of fast feature extraction. The two output features are forwarded to the internal info path and spatial info path, respectively.

In VisualNet, the spatial info path is designed to further extract the high-level spatial features after the retina layer, that we adopt three IRCs to construct the path (with three downsamplings). We denote the output feature of spatial info path as spatial feature S, which is then forwarded to the visual cortex module for further regulations and processing.

Since the internal info path is more task-specific which serves as extensive learning for representation extraction, we design the internal info path in VisualNet as the main learner in the framework. In other words, we apply a regular NN as the backbone of VisualNet in the internal info path along with an IRC cell before the backbone to align the dimension. We denote the input and the output of the backbone as the internal feature A and the learned internal feature A' respectively, and both are forwarded to the visual cortex module. Note that by using a regular network as VisualNet backbone, we only take the stacked convolutional cells (such as conv2 to conv5 layers in ResNets) since they act as the main feature extractor and account for the majority of the computation/latency. We use V-{backbone name} to denote the name of the framework implemented with the specific backbone.

After A, A' and S are obtained from the two paths, we design the visual cortex to proceed with two sub-tasks: 1). regulating A and S towards the properties of independence, sparsity, and accuracy; 2). mixing A' and S as the mixed feature and propagating it to get the prediction for the specific task. For the first sub-task, the mixing of S and the learned application-specific internal feature A' is achieved by the transposed convolution between the two features. The mixing operation could be interpreted as decoding the learned feature A' by S. Then the decoded mixed feature is propagated through the task-specific module. For classification, the features go through a sequence of convolutional cells (conv3+max-pooling). For object detection, the features go through two convolutional cells without downsampling and then propagate into the detection head (such as Faster R-CNN [35] or SSD [36]).

For the second sub-task, we use three loss terms to regulate the learning. The first term measures independence among the extracted spatial feature S, where we adopt the approximation of negentropy due to its robustness and simple computation [37]. The second term is to guide the sparsity of the spatial feature S. The third term computes the reconstruction error between the mixing (reconstruction) result and the input image, which ideally should be identical. Mathematically, the loss function of the additional regulation for S and A with the three loss terms can be expressed as

$$\mathcal{L}_{\text{reg}} = \lambda_i J(\mathsf{S}) + \lambda_s g(\mathsf{S}) + \lambda_r \parallel \mathsf{A} \otimes \mathsf{S} - \mathsf{X}_{in} \parallel_2^2, \tag{1}$$

where λ_i, λ_s and λ_r are the coefficients of the three loss terms which are all set to 1.0 in our experiments; $\otimes$ is a transposed convolution operation, i.e., the data is reconstructed as $\mathsf{A} \otimes \mathsf{S}$ so that the dimension of the mixing results matches that of the input data; $g(\cdot)$ denotes the sparsity loss where L_1 loss is adapted; $J(\cdot)$ denotes the independence loss (approximation of negentropy to measure nongaussianity [38]), which can be computed as

$$J(\mathsf{S}) = \text{avg}(-0.75\log\cosh(\mathsf{S}/\alpha)), \tag{2}$$

where $\text{avg}(\cdot)$ denotes element-wise average of an entire tensor.

With all modules and regulations proposed above, we can achieve end-to-end training by combining the loss obtained from Equation (1) with the target task's loss function. Specifically, the loss function $\mathcal{L}$ can be expressed as

$$\mathcal{L} = \mathcal{L}_{\text{target}} + \lambda_{\text{e}}\mathcal{L}_{\text{reg}}, \tag{3}$$

where $\mathcal{L}_{\text{target}}$ is the loss function defined by the target task (such as image classification error or object detection error), which we denote as target loss; λ_{e} is the loss weight of the additional regulations, which is set to 0.1 in our experiments.

3.2.2 Acceleration Analysis

Compared with a regular NN, VisualNet constructed using it as backbones can achieve lower latency, mainly because the mixing tensors are drastically smaller in size than the original image and thus can be handled much faster. As constructed in VisualNet, the internal info path is related to the learning task and the spatial info is enforced to be independent, which allows us to explore task parallelism. In other words, we treat the internal feature A as independent tasks corresponding to independent spatial feature S. Thus the internal feature in the VisualNet backbone can be treated as independent batches and handled in parallel (i.e., new task parallelism) by invoking multiple instances of the backbone network, as shown in Fig. 3c. Note that the processing of each internal feature can still utilize any existing parallelization techniques by applying them to the backbone.

3.3 The Connection Between VisualNet and Regular Neural Network

The VisualNet architecture seems to be built on intuitions so far. In this section, we will establish a theorem that explains the underlying connection between a regular neural network and VisualNet with it as the backbone. We will show that using a regular neural network to learn the transform of the mixing tensors only for the target application, with bases unchanged, can indeed yield an accuracy comparable with that of the regular neural network.

Theorem. Denote a regular neural network with input X_{in} and output $\mathsf{X}_{out} = f(\mathsf{X}_{in})$. In the ideal scenario that the Visual-encoder/decoder are lossless (i.e., the transposed convolution results in a perfect reconstruction $\mathsf{X}_{in} = \mathsf{A}_{in} \otimes \mathsf{S}$), under first order approximation X_{out} can be expressed as $\mathsf{X}_{out} = \mathsf{A}_{out} \otimes \mathsf{S}$, where A_{out} only depends on the network f and the mixing tensors A_{in}.

Proof. Proof by construction. Since the transposed convolution is a completely linear operation, with the assumption that the Visual-encoder/decoder are lossless, each element (pixel) P in X can be reconstructed as

$$P_i = A_{i,1}S_1 + A_{i,2}S_2 + \cdots + A_{i,m}S_m, \tag{4}$$

where S_q $(1 \le q \le m)$ are the realization of the i^{th} basis tensors S_i put in a vector form, and $A_{i,q}$ $(1 \le q \le m)$ are the corresponding mixing vectors that can be obtained from the mixing tensors A. Here we call the data in the form of Equation (4) as *canonical form*.

Any neural network can be viewed as a function that operates on the elements (pixels) in the input image, through operations such as *scaling*, *sum*, *max*, etc. When the input elements are in the canonical forms, we note that these operations can be done with first-order moments (w.r.t. the independent basis) preserved, as detailed below. Specifically, given any pair of data D_i, D_j in the form as shown in (4), their scaling D_{scale}, sum D_{sum} and max $D_{\max}$ are respectively shown as

$$D_{\text{scale}} = w \times D_i = \sum_{q=1}^{m} A_{\text{scale},q} X_q, \tag{5}$$

$$D_{\text{sum}} = sum(D_i, D_j) = \sum_{q=1}^{m} A_{\text{sum},q} X_q, \tag{6}$$

$$D_{\max} = max(D_i, D_j) = \sum_{q=1}^{m} A_{\max,q} X_q, \tag{7}$$

where w is the scaling ratio.

The key of the above operations is that the results are put back into the same canonical form as shown in Equation (4), which allows the same operations to be carried out repeatedly throughout the entire network during forward propagation. The mixing vectors ($A_{\text{scale},q}$, $A_{\text{sum},q}$, and $A_{\max,q}$) are obtained by

$$A_{\text{scale},q} = w \times A_{i,q}, \tag{8}$$

$$A_{\text{sum},q} = A_{i,q} + A_{j,q}, \tag{9}$$

$$A_{\max,q} = \Phi(\beta) A_{i,q} + \Phi(-\beta) A_{j,q},$$
$$\beta = (\mu_{P_i} - \mu_{P_j}) / (\sigma_{P_i}^2 + \sigma_{P_j}^2 - 2\sigma_{P_i}\sigma_{P_j})^{1/2}, \tag{10}$$

where $\Phi(\cdot)$ is the Cumulative Distribution Function (CDF) of a standardized normal distribution; μ_{P_i}, μ_{P_j} and $\sigma_{P_i}^2, \sigma_{P_j}^2$ are the mean values and variances of P_i and P_j, respectively, which can be calculated as

$$\mu_P = \frac{1}{m}\sum_{i=1}^{m} A_i, \sigma_P = \frac{1}{m}\sum_{i=1}^{m} (A_i - \mu_P)^2. \tag{11}$$

The results of *scaling* and *sum* are self-evident, while that of the *max* operation is based on [39] which proves that the first-order moments are preserved. As such, we can propagate the mixing vectors in the canonical forms all the way to the output and pack them back in the tensor form as A_{out}; as S always remain the same throughout the propagation, $\mathsf{X}_{out} = \mathsf{A}_{out} \otimes \mathsf{S}$. □

A closer look at the proof above reveals that the nonlinear operations in a neural network, such as *max*, cause the computation to be coupled basis-wise. Specifically, computing the basis coefficient of the *max* output requires calculating the means $(P_{i,\mu}, P_{j,\mu})$ and variances $(\sigma_{P_i}^2, \sigma_{P_j}^2)$ of both inputs first, which involve mixing tensors of all the basis tensors in the two input canonical forms. If we further relax these nonlinear operations to be basis-wise, e.g.,

$$P_{\max,\text{ new}} = \sum_{q=1}^{m} A_{\max,q} S_q = \sum_{q=1}^{m} \mathbf{max}(A_{i,q}, A_{j,q}) S_q, \tag{12}$$

then we can fully decouple the propagation of the mixing vectors for different S_i and allow them to be done in parallel. If we pack these mixing vectors in the tensor form, then this is exactly the approach of the VisualNet, which allows the mixing tensors of each basis to be handled through an instance of the regular neural network backbone in training and inference.

The above discussion demonstrates the relationship between VisualNet and its regular network backbone and supports the intuitive approach's feasibility.

4 Results

In this section, we first present the ablation studies demonstrating the efficacy of various modules in VisualNet. We then evaluate how the proposed VisualNet reduces the inference latency of various NNs in image classification task by using these NNs as backbones. We further show the visualization of the two paths in the VisualNet process. Lastly, we evaluate VisualNet with object detection task. Note that VisualNet is not a new NN architecture but rather a technique that resembles the functions of the human visual system to reduce the inference latency of existing CNNs.

4.1 Experimental Setup

The main focus of VisualNet is to decompose existing CNN architectures for the acceleration of single-image inference via task parallelism, rather than accuracy improvement. As such, the goal of VisualNet is not to beat the accuracy or latency of the state-of-the-art which mostly comes from neural architecture search [40] for dedicated task/dataset. Instead, we chose to demonstrate its potential to reduce the inference latency of some of the most widely used NN architectures, including two lightweight networks ResNet18 (V-ResNet18), MobileNet v2 (V-MobileNet), and three large networks ResNet50 (V-ResNet50), VGG16 (V-VGG16), and DenseNet121 (V-DenseNet121).

We used PyTorch to implement all models and evaluate them on both GPUs and CPUs. For GPUs, we used a cluster of Nvidia P100. For CPUs, we used Intel Xeon multi-core processors (48 cores with 256 G memory), similar to those used in many real-time deep learning applications [41], [42]. As discussed earlier, VisualNet provides task parallelism, which can be combined with existing operator and model parallelism techniques that accelerate single image inference. To demonstrate this, we used the state-of-the-art Intel MKL-DNN[1] for CPU and CUDA/cuDNN for GPU to fully accelerate all the networks, including VisualNets and the NNs they build on (regular network counterparts).

The training configurations such as learning rate schedule, weight initialization/decay, optimizer, and input dimension follow corresponding regular networks' approaches. All modules in the models are trained from scratch. We evaluated the effect of the independent instance number in Section 4.3.3, and it was shown that a larger one leads to higher accuracy. Therefore, unless otherwise specified, m is set to 24 for all the VisualNets.

Although VisualNet can reduce the latency in both training and inference, in the experiments we focus on the latter. This is because in training the computation resources can always be fully utilized through instance parallelism (i.e.,

1. https://github.com/intel/mkl-dnn

TABLE 1
Ablation Studies of VisualNet Using Top-1 Accuracy (%) on CIFAR-10 Classification

Discussion item	Configuration	Acc. (%)
Backbone	ResNet50	93.9
	VGG16	93.1
	Adaptive average pooling	86.4
Visual cortex	w/ visual cortex	93.9
	w/o visual cortex	91.7
Loss term	w/ all regulation losses	93.9
	w/o independence loss	93.0
	w/o sparse loss	90.9
	w/o reconstruction loss	92.2
	w/o all regulation losses	91.4

processing multiple input images in parallel), and reducing the latency of a single image is not important. For real-time inference, however, the latency of a single image is critical.

4.2 Ablation Studies

We performed ablation studies to investigate the effects of different body parts of VisualNet on the CIFAR-10 classification task. Note that the structure of the VisualNet is adjusted to fit CIFAR-10. We remove two downsampling modules (one in the retina layer and one in the spatial info path), and for the visual cortex, we remove the third convolutional modules (conv3(1280)+MP) and let $k = 10$. The results are shown in Table 1.

4.2.1 Effects of VisualNet Backbone

We conducted experiments to study the effect of VisualNet backbones built with regular NNs to learn the transformation of the internal feature A for target applications. V-ResNet50 and V-VGG16 achieve a Top-1 accuracy of 93.9% and 93.1% on CIFAR-10, respectively. We then replaced the VisualNet backbones shown in Fig. 3 with an adaptive average pooling operation, and with the same settings of hyperparameters and optimizer, the accuracy drops to 86.4%, showing signs of underfitting. This convincingly shows that VisualNet backbones play critical roles in learning the transform effectively by increasing the model's representation power.

4.2.2 Effects of Visual Cortex

We conducted experiments to study the effect of the visual cortex module which performs feature mixing and regulation on the outputs from the previous two paths, and producing the output. We experiment with V-ResNet50 and replace the lightweight NN (convolution cells) in the visual cortex with an average pooling operation and a linear classifier. The test accuracy drops to 91.7%, which is 2.2% lower than the one with the visual cortex. This shows that feature fusion with the lightweight NN impacts accuracy and is necessary to the visual cortex design.

4.2.3 Effects of Loss Terms

In Equation (1), the regulation losses (independence, sparsity, accuracy) are introduced to regulate the training process, with four hyperparameters λ_i, λ_s, λ_r, and λ_e. These parameters for losses calculation will only affect the accuracy but not the latency and throughputs. We conducted experiments with V-ResNet50 to investigate their effects on accuracy. Specifically, we verified the following cases: (i.) without the independence loss ($\lambda_i = 0$); (ii.) without the sparsity loss ($\lambda_s = 0$); (iii.) without the reconstruction loss ($\lambda_r = 0$); and (iv.) without all the regulation losses ($\lambda_e = 0$). The validation accuracies for these four cases are 93.0%, 90.9%, 92.2%, and 91.4%, respectively. These convincingly show that the regulation losses are necessary for better performance, and the sparsity loss matters the most among all the loss terms.

4.3 Image Classification

We conducted the experiments on image classification with the ILSVRC2012 ImageNet classification dataset [43], and our VisualNet is constructed following Fig. 3a. We use Top-1 accuracy to evaluate the center crop ($256^2 \times 3$) from the images ($292^2 \times 3$) in the validation set. Note that the regular networks are trained and evaluated with the new dimension ($256^2 \times 3$) rather than the conventional dimension ($224^2 \times 3$).

4.3.1 Accuracy

The Top-1 classification accuracies of VisualNets and their regular network counterparts on ImageNet are shown in the first data column of Table 2. It can be noticed that except for V-VGG16 which has a 0.4% lower accuracy than the regular VGG16, all other VisualNets achieve higher accuracies, ranging from 0.2% to 0.8%, than their regular network counterparts. The slightly increased accuracy comes from the fact that VisualNet can learn the feature from the input with its special structures more effectively.

4.3.2 Latency

Before we look at the latency, the FLOP count (FLOPs) and the parameter size (#params) of each model are reported in the rightmost two columns in Table 2. The smaller input size of the backbone reduces the amount of computation of VisualNets, leading to a smaller total FLOP count than their regular network counterparts. On the other hand, the difference in parameter sizes between VisualNets and regular networks is caused by two reasons: (i.) the additional parameters brought by the retina module, spatial path, and visual cortex module, (ii.) VisualNet only takes stacked convolutional cells in the regular networks as its backbone. For example, the conv1 layer in ResNets and the multiple fully connected layers in VGG16 are not included. For VGG16, the fully connected layers account for the overwhelming majority of the parameters (3 layers take up 123.7 M out of the total 138.4 M parameters) and they are used for final dimension reduction and classification.

The latency of these networks on GPUs and CPUs is shown in the second and third data columns in Table 2, where both VisualNets and their regular network counterparts are fully parallelized. When implementing VisualNets on GPUs, the latency can be reduced by up to 57.7%. On CPUs, the latency can be reduced by up to 80.6%. From the data, we can see that as the total FLOP count in the regular network gets larger, the latency speedup becomes more significant. This is because the overhead induced by the extra

TABLE 2
Top-1 Accuracy (%) Without and With Element-Wise Noise Added, Latency on GPU/CPU, Total FLOP Count, and Parameter Size of VisualNets and Their Regular Network Counterparts on ImageNet Classification

Networks	w/o noise Acc. (%)	w/ noise Acc. (%)	Latency (ms)		FLOPs (G)	#params (M)
			GPU	CPU		
MobileNet	71.2	63.4±0.09	11.5	31.9	0.39	3.50
V-MobileNet	**71.4**	**68.8±0.07**	10.3(-10.4%)	27.3(-14.4%)	0.11	3.77
DenseNet121	75.2	72.3±0.12	24.4	87.7	3.70	7.98
V-DenseNet121	**75.8**	**73.1±0.11**	20.9(-14.3%)	56.4(-35.7%)	0.97	8.48
ResNet18	69.6	62.9±0.10	5.6	45.6	2.37	11.69
V-ResNet18	**70.4**	**66.3±0.05**	5.2(-7.1%)	31.5(-30.9%)	0.64	12.67
ResNet50	76.8	69.9±0.08	10.6	92.3	5.34	25.56
V-ResNet50	**77.2**	**73.7±0.10**	9.1(-14.2%)	46.2(-49.9%)	1.38	25.10
VGG16	**70.5**	64.3±0.08	13.7	156.0	20.17	138.37
V-VGG16	70.1	**66.3±0.06**	5.8(-57.7%)	30.3(-80.6%)	1.33	16.23

For latency, all the networks are fully parallelized using state-of-the-art acceleration libraries as described in Section 4.1.

modules in VisualNets becomes less significant than the benefit of task parallelism brought by design. Even for the compact MobileNet, V-MobileNet can reduce its latency by 10.4% and 14.4% on GPUs and CPUs.

4.3.3 Effect of the Number of Independent Instances

We further study the effect of number of independent on the accuracy of VisualNet (it has little impact on latency since the corresponding mixing tensors are handled in parallel). We use the same tasks and the same networks used above. The results are shown in Table 3. From the table we can see that larger number of independent instances leads to higher accuracy. This is because with larger basis dimension, more useful features can be extracted by both internal info path and spatial info path, and learned by the following backbone. The results of backbone networks ResNet18 and ResNet50 with two different attention modules [44], [45] are also included for reference. We can see that without implementing module-level modification on backbone network, our VisualNets still are on par with the state-of-the-art attention models.

4.3.4 Robustness of VisualNet

We conducted the experiments to study the effect of noisy input for VisualNet, where the additional regulations are added towards independence, sparsity, and accuracy. We use the classification task with ImageNet as an example to prove the robustness of our VisualNets for noisy input. Both VisualNets and the regular networks are trained with the original images without noise and validated with the noise added images. The noise for each image is randomly selected from the Gaussian/Laplace/Poisson noises, with a random scale between 0.01 and 0.1 per channel. The validation with noisy input is performed 5 times, and the results with deviation are shown in Table 2. Compared with the regular network counterparts, our VisualNets are much less affected by the noisy input images and achieved up to 5.4% higher accuracy than the regular counterparts.

TABLE 3
Comparison of Top-1 Accuracy (%) on ImageNet Classification Between ResNet18 (R18), ResNet50 (R50), DenseNet121 (D121), and VGG16 (V16) and Their VisualNets Implementations

Networks	R18	R50	D121	V16
Regular	69.6	76.8	75.2	**70.5**
Regular+SE [44]	70.6	76.9	-	-
Regular+CBAM [45]	**70.7**	**77.3**	-	-
VisualNets ($m = 8$)	69.7	75.7	71.6	66.4
VisualNets ($m = 16$)	69.9	76.9	74.4	68.9
VisualNets ($m = 24$)	70.4	77.2	**75.8**	70.1

For each VisualNet three different number of independent instances are set as $m = 8, 16, 24$. The modifications based on attention modules, squeeze-and-excitation (SE) [44] and convolutional block attention (CBAM) [45], are also included for reference.

4.3.5 Visualization of VisualNet Process

We use some images from ImageNet as examples to visualize the process in the V-ResNet50 network. Some of the corresponding spatial features (S), internal features (A) and the mixed features between the learned internal feature and spatial feature ($\mathsf{A}' \otimes \mathsf{S}$) are illustrated in Fig. 4. We can see that the mixing features ($\mathsf{A}' \otimes \mathsf{S}$) indeed contain spatial information, which justifies using regular CNNs as backbones to transform them for target applications.

4.4 Object Detection

We evaluate the performance of VisualNets on object detection task. Based on MMDetection deteciton framework, we adopt two configurations: Faster R-CNN [35] with ResNet50 and SSD300 [36] with VGG16 [22] for evaluation. We only use GPUs for evaluation since the CPU inference is not supported by the framework. The models are trained based on the union of VOC 2007 trainval and VOC 2012 trainval ("07 +12") tested on VOC 2007 testset.

The object detection accuracy (mAP@0.5), the latency and the total FLOP count of the VisualNets and their regular network counterparts are shown in Table 4. The table shows that VisualNets can achieve a similar accuracy compared with the regular network counterpart, with 0.4% higher in Faster R-CNN+ResNet50 and 0.6% lower in SSD300 +VGG16. Meanwhile, due to reduced computation needed for VisualNets, the latency and the total FLOP count for

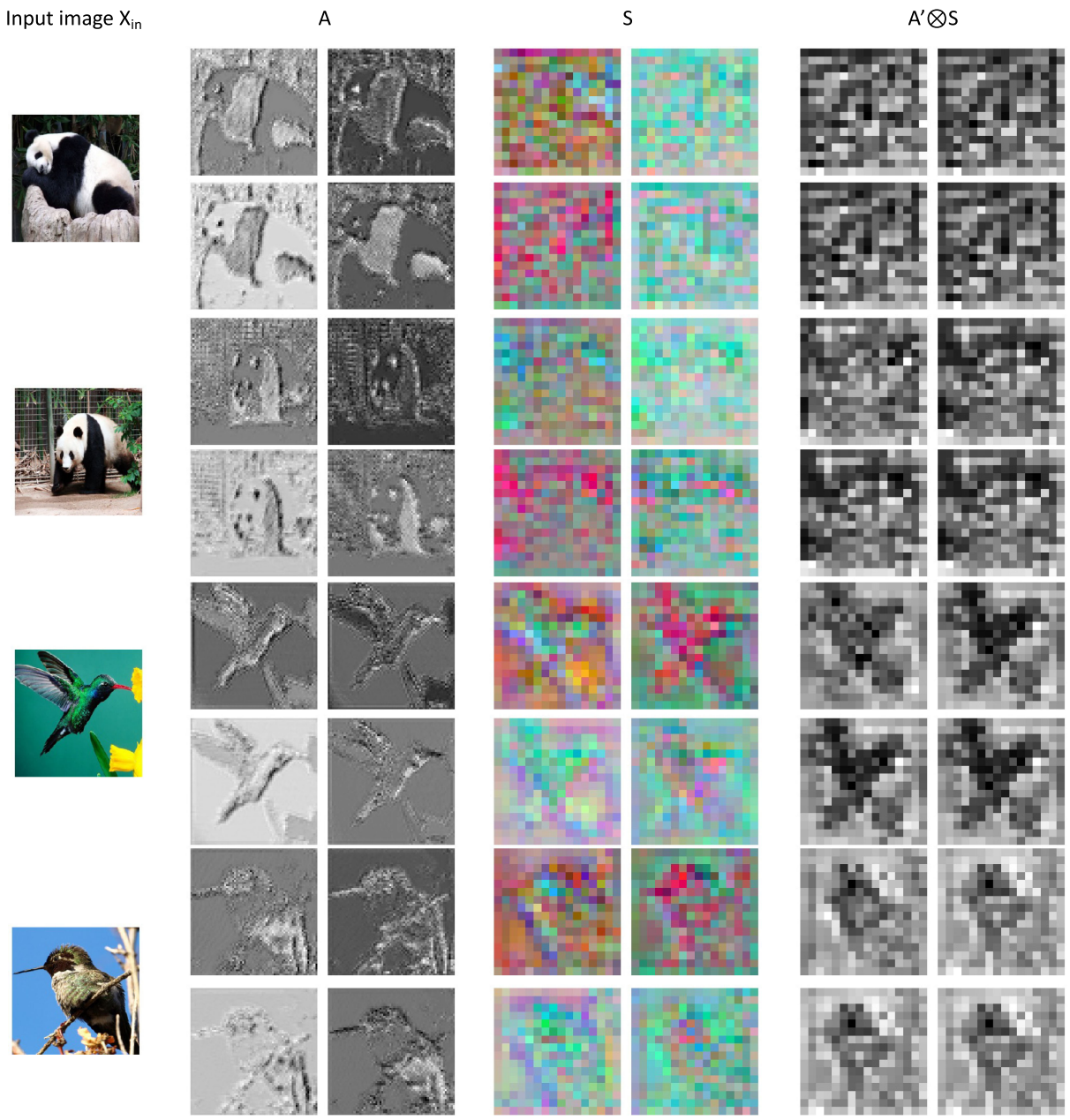


Fig. 4. Visualization of the spatial feature (S), the internal feature (A), and the mixed features between the learned internal feature A′ and the spatial feature S (A′ ⊗ S) for some example input images (X_{in}). Note that for each image only some of the features are shown. They are of different sizes and are scaled differently for compact display.

TABLE 4
The mAP@0.5 (%), Latency on GPU, Total FLOP Count, and Parameter Size of VisualNets and Their Regular Network Counterparts on PASCAL VOC 2007 Object Detection

Configuration	Network	mAP@0.5 (%)	Latency(ms)	FLOPs(G)	#params (M)
Faster R-CNN+ResNet50	Regular	74.1	73.5	31.63	41.53
	VisualNet	**74.5**	69.8	27.65	42.47
SSD300+VGG16	Regular	**77.2**	38.8	34.42	34.31
	VisualNet	76.6	30.9	15.51	35.26

For latency, all the networks are fully parallelized using state-of-the-art acceleration libraries as described in Section 4.1.

both detection frameworks are reduced. For SSD300+VGG16, VisualNet can reduce the latency by 20.4%. It achieves less significant speedup on Faster R-CNN+ResNet50 due to the heavy detection head in Faster R-CNN, which is not part of the backbone and thus cannot be accelerated by VisualNet.

5 Conclusion

In this work, we propose a general framework (VisualNet) that mimics the human visual system's function and structure, leading to an enhancement of the scalability and reduction of the single image inference latency in existing CNNs. Specifically, through a lightweight retina layer, an input image is decomposed and split into two for spatial and internal info paths, respectively. The spatial feature is obtained after the spatial path, and the internal feature can be transformed through learning for different target applications, such as image classification and object detection using a regular NN as the backbone. The spatial feature in the visual cortex decodes the learned internal feature to get the inference results. Experimental results on ImageNet classification and CIFAR-10 classification using various NNs show that VisualNet provides up to 80.6% latency reduction while achieving similar or slightly higher accuracy.

References

[1] X. Xu *et al.*, "Scaling for edge inference of deep neural networks," *Nature Electron.*, vol. 1, no. 4, pp. 216–222, 2018.

[2] K. Muhammad, A. Ullah, J. Lloret, J. Del Ser, and V. H. C. de Albuquerque, "Deep learning for safe autonomous driving: Current challenges and future directions," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4316–4336, Jul. 2021.

[3] T. Wang, J. Xiong, X. Xu, and Y. Shi, "SCNN: A general distribution based statistical convolutional neural network with application to video object detection," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 5321–5328.

[4] Y. Choi, M. El-Khamy, and J. Lee, "Universal deep neural network compression," 2018, *arXiv:1802.02271*.

[5] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6848–6856.

[6] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.

[7] L. Liu and J. Deng, "Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3675–3682.

[8] U. A. Muller and A. Gunzinger, "Neural net simulation on parallel computers," in *Proc. IEEE Int. Conf. Neural Netw.*, 1994, pp. 3961–3966.

[9] A. Coates, B. Huval, T. Wang, D. Wu, B. Catanzaro, and N. Andrew, "Deep learning with COTS HPC systems," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1337–1345.

[10] T. Ben-Nun and T. Hoefler, "Demystifying parallel and distributed deep learning: An in-depth concurrency analysis," 2018, *arXiv:1802.09941*.

[11] Y. Zhu *et al.*, "Statistical training for neuromorphic computing using memristor-based crossbars considering process variations and noise," in *Proc. Des. Automat. Test Europe Conf. Exhib.*, 2020, pp. 1590–1593.

[12] J. Jiang, B. Cui, C. Zhang, and L. Yu, "Heterogeneity-aware distributed parameter servers," in *Proc. ACM Int. Conf. Manage. Data*, 2017, pp. 463–478.

[13] N. Kriegeskorte, "Deep neural networks: A new framework for modeling biological vision and brain information processing," *Annu. Rev. Vis. Sci.*, vol. 1, pp. 417–446, 2015.

[14] H. Kolb, "How the retina works: Much of the construction of an image takes place in the retina itself through the use of specialized neural circuits," *Amer. Scientist*, vol. 91, no. 1, pp. 28–35, 2003.

[15] M. T. Banich and R. J. Compton, *Cognitive Neuroscience*, Cambridge, U.K.: Cambridge Univ. Press, 2018.

[16] B. Kolb, I. Q. Whishaw, and G. C. Teskey, *An Introduction to Brain and Behavior*, New York, NY, USA: Worth, 2001.

[17] A. Hepburn, V. Laparra, J. Malo, R. McConville, and R. Santos-Rodriguez, "Perceptnet: A human visual system inspired neural network for estimating perceptual distance," in *Proc. IEEE Int. Conf. Image Process.*, 2020, pp. 121–125.

[18] S. Zweig and L. Wolf, "InterpoNet, a brain inspired neural network for optical flow dense interpolation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4563–4572.

[19] Y. Huang *et al.*, "Neural networks with recurrent generative feedback," in *Proc. 34th Int. Conf. Neural Inf. Process. Syst.*, 2020, Art. no. 46.

[20] K. Abdelouahab, M. Pelcat, J. Serot, and F. Berry, "Accelerating CNN inference on FPGAs: A survey," 2018, *arXiv:1806.01683*.

[21] R. Reed, "Pruning algorithms-a survey," *IEEE Trans. Neural Netw.*, vol. 4, no. 5, pp. 740–747, Sep. 1993.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[23] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

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

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

[26] 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.

[27] M. Tan and Q. V. Le, "MixConv: Mixed depthwise convolutional kernels," 2019, *arXiv:1907.09595*.

[28] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[29] A. Howard *et al.*, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1314–1324.

[30] M. Tan *et al.*, "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2820–2828.

[31] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 116–131.

[32] S. Mehta and M. Rastegari, "MobileViT: Light-weight, general-purpose, and mobile-friendly vision transformer," 2021, *arXiv:2110.02178*.

[33] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[34] D. Chen, J.-P. Mei, C. Wang, Y. Feng, and C. Chen, "Online knowledge distillation with diverse peers," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 3430–3437.

[35] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[36] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.

[37] A. Hyvarinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Trans. Neural Netw.*, vol. 10, no. 3, pp. 626–634, May 1999.

[38] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, vol. 46, Hoboken, NJ, USA: Wiley, 2004.

[39] C. E. Clark, "The greatest of a finite set of random variables," *Operations Res.*, vol. 9, no. 2, pp. 145–162, 1961.

[40] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," 2018, *arXiv:1806.09055*.

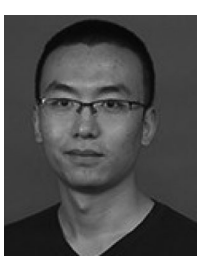
[41] G. Raina, H. Corporaal, P. Cuijpers, M. Peemen, and G. Rauwerda, "Deep convolutional network evaluation on the Intel Xeon Phi: Where subword parallelism meets many-core," Master's thesis, Eindhoven Univ. Technol., Eindhoven, Netherlands, 2016. [Online]. Available: https://pure.tue.nl/ws/files/46932913/844256-1.pdf

[42] C. Zhuang, S. Zhang, X. Zhu, Z. Lei, J. Wang, and S. Z. Li, "FLDet: A CPU real-time joint face and landmark detector," in *Proc. IAPR Int. Conf. Biometrics*, 2019, pp. 1–8.

[43] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

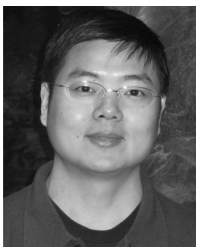
[44] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[45] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 3–19.



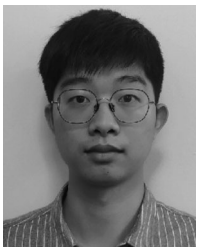
Tianchen Wang (Student Member, IEEE) received the PhD degree from the University of Notre Dame, Notre Dame, Indiana, in 2020. He is currently a senior research scientist with Comcast AI, responsible for develop computer vision and deep learning edge solutions. He has published more than 20 technical papers in conferences and journals. His current research focuses on the application and implementation of deep learning and computer vision.

Jiawei Zhang received the bachelor's degree in information and computing science from Xidian university, in 2017. He is currently working toward the PhD degree in computer science in the Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University, Shanghai, China. He majors in biomedical image analysis.



Jinjun Xiong (Member, IEEE) received the PhD degree from the University of California at Los Angeles, Los Angeles, California, in 2006. He is currently the empire innovation professor with the Department of Computer Science and Engineering, University at Buffalo (UB). Previously, he was program director and a research staff member with IBM Thomas J. Watson Research Center, Yorktown, New York, responsible for building world-class programs on cognitive computing systems research. He has published more than 100 technical papers in refereed international conferences and journals. His current research interests include future computing systems, cognitive computing, Big Data analytics, smarter energy, and very large-scale integrated circuit designs. He was a recipient of numerous best paper awards, the Best Paper Award nominations, and the Outstanding PhD Award from the University of California at Los Angeles.

Song Bian (Member, IEEE) received the BS degree with highest distinction from the University of Wisconsin-Madison, in 2014, and the MS and PhD degrees from Kyoto University, in 2017 and 2019, respectively. He is currently an associate professor with Beihang University. His main areas of interest include applied cryptography, secure multi-party computation, and domain-specific hardware accelerators. He was a research fellow of the Japan Society for the Promotion of Science from 2017 to 2019. He served as technical committee members/reviewers for top international conferences/journals across different fields of studies, including AAAI, ASP-DAC, *Journal of Cryptology*, and *ACM Transactions on Design Automation of Electronic Systems*. He is a member of the IPSJ.



Zheyu Yan received the BS degree from Zhejiang University, in 2019. He is currently working toward the PhD degree in the Department of Computer Science and Engineering, University of Notre Dame. His general research direction is hardware/software co-design for neuromorphic computing. His current interests are uncertainty modeling of non-volatile emerging device-based neural accelerators and achieving efficient DNN inference via co-design efforts.

Meiping Huang received the BS degree in clinical medicine from Sun Yet-sen University, Guangzhou, China, and the MS degree in medical imaging from Jinan University. She is currently a chief physician with the Interventional Catheterization Lab, Guangdong Provincal People's Hospital, Guangzhou, China. She is members of the National Imaging Group of the Chinese Society of Cardiology, the National Pediatrics Group of the Chinese Society of Radiology, the National Radiology Group of the Chinese Pediatrics Society, and a variety of other provincial societies. She has presided four Science and Technology Planning Project of Guangdong Province, and participated in more than 10 national-level and scientific research projects. Her research interests include imaging diagnosis of cardiovascular disease, surgical navigation of structural heart disease based on multimodal imaging technology, application of Big Data technology and artificial intelligence in cardiovascular medicine, 3D printing, and visualization technology research and development.

Jian Zhuang received the BS degree in clinical medicine from First Military Medical University, Guangzhou, China, in 1984, and the MS and PhD degrees in cardiovascular surgery form Guangdong Cardiovascular Institute, Guangzhou, China. He is the director of Guangdong Cardiovascular Disease Center, Guangdong Provincial People's Hospital. He is also the director of WHO Collaborating Centre for Research and Training in Cardiovascular Diseases, and the chief expert of Heart Surgery at Guangdong Provincial People's Hospital. He was the chairman of ninth committee of the Chinese Society of Thoracic and Cardiovascular Surgery. Since 1989, he worked with Guangdong Provincial People's Hospital and engaged in the prevention, diagnosis and treatment of congenital heart disease, covering patients in all age. He is also interested at applications of artificial intelligence and three-dimensional technologies in structural heart diseases to increase the precision and intelligence of clinical practice.

Takashi Sato (Member, IEEE) received the BE and ME degrees from Waseda University, Tokyo, Japan, and the PhD degree from Kyoto University, Kyoto, Japan. He was with Hitachi, Ltd., Tokyo, Japan, from 1991 to 2003; with Renesas Technology Corp., Tokyo, Japan, from 2003 to 2006; and with the Tokyo Institute of Technology, Yokohama, Japan. In 2009, he joined the Graduate School of Informatics, Kyoto University, Kyoto, Japan, where he is currently a professor. He was a visiting industrial fellow with the University of California, Berkeley, from 1998 to 1999. His research interests include CAD for nanometer-scale LSI design, fabrication-aware design methodology, and performance optimization for variation tolerance. He is a member of the Institute of Electronics, Information and Communication Engineers (IEICE). He received the Beatrice Winner Award at ISSCC 2000, and the Best Paper Award at ISQED 2003.

Xiaowei Xu received the BS and PhD degrees in electronic science and technology from the Huazhong University of Science and Technology, Wuhan, China, in 2011 and 2016, respectively. He is currently an assistant professor with Guangdong Cardiovascular Institute, Guangdong Provincial People's Hospital, Guangzhou, China. He worked as a post-doc researcher with the University of Notre Dame, IN, USA from 2016 to 2019. His research interests include deep learning, and medical image segmentation. He was a recipient of DAC system design contest special service recognition reward in 2018 and outstanding contribution in reviewing, Integration, the VLSI journal in 2017. He has served as TPC members in ICCD, ICCAD, ISVLSI and ISQED.

Yiyu Shi (Senior Member, IEEE) received the BS degree in electronic engineering from Tsinghua University, Beijing, China in 2005, and the MS and PhD degrees in electrical engineering from the University of California, Los Angeles, in 2007 and 2009, respectively. He is currently a professor with the Department of Computer Science and Engineering, University of Notre Dame, the site director of NSF I/UCRC Alternative and Sustainable Intelligent Computing, and the director of the Sustainable Computing Lab (SCL). His current research interests focus on hardware intelligence and biomedical applications. In recognition of his research, many of his papers have been nominated for the best paper awards in top conferences. He was also the recipient of NSF CAREER Award, IEEE Region 5 Outstanding Individual Achievement Award, IEEE TCVLSI Mid-Career Research Award, among others. He is the education chair of ACM SIGDA, deputy editor-in-chief of the IEEE VLSI CAS Newsletter.