Machine Intelligence Research
www.mi-research.net

19(5), October 2022, 366-411
DOI: 10.1007/s11633-022-1340-5

Efficient Visual Recognition: A Survey on Recent Advances and Brain-inspired Methodologies

Yang Wu[1] Ding-Heng Wang[2] Xiao-Tong Lu[3] Fan Yang[4] Man Yao[2,5]
Wei-Sheng Dong[3] Jian-Bo Shi[6] Guo-Qi Li[7,8]

[1] Applied Research Center Laboratory, Tencent Platform and Content Group, Shenzhen 518057, China

[2] School of Automation Science and Engineering, Faculty of Electronic and Information Engineering, Xi′an Jiaotong University, Xi′an 710049, China

[3] School of Artificial Intelligence, Xidian University, Xi′an 710071, China

[4] Division of Information Science, Nara Institute of Science and Technology, Nara 6300192, Japan

[5] Peng Cheng Laboratory, Shenzhen 518000, China

[6] Department of Computer and Information Science, University of Pennsylvania, Philadelphia PA 19104-6389, USA

[7] Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

[8] University of Chinese Academy of Sciences, Beijing 100190, China

Abstract: Visual recognition is currently one of the most important and active research areas in computer vision, pattern recognition, and even the general field of artificial intelligence. It has great fundamental importance and strong industrial needs, particularly the modern deep neural networks (DNNs) and some brain-inspired methodologies, have largely boosted the recognition performance on many concrete tasks, with the help of large amounts of training data and new powerful computation resources. Although recognition accuracy is usually the first concern for new progresses, efficiency is actually rather important and sometimes critical for both academic research and industrial applications. Moreover, insightful views on the opportunities and challenges of efficiency are also highly required for the entire community. While general surveys on the efficiency issue have been done from various perspectives, as far as we are aware, scarcely any of them focused on visual recognition systematically, and thus it is unclear which progresses are applicable to it and what else should be concerned. In this survey, we present the review of recent advances with our suggestions on the new possible directions towards improving the efficiency of DNN-related and brain-inspired visual recognition approaches, including efficient network compression and dynamic brain-inspired networks. We investigate not only from the model but also from the data point of view (which is not the case in existing surveys) and focus on four typical data types (images, video, points, and events). This survey attempts to provide a systematic summary via a comprehensive survey that can serve as a valuable reference and inspire both researchers and practitioners working on visual recognition problems.

Keywords: Visual recognition, deep neural networks (DNNS), brain-inspired methodologies, network compression, dynamic inference, survey.

Citation: Y. Wu, D. H. Wang, X. T. Lu, F. Yang, M. Yao, W. S. Dong, J. B. Shi, G. Q. Li. Efficient visual recognition: A survey on recent advances and brain-inspired methodologies. *Machine Intelligence Research*, vol.19, no.5, pp.366–411, 2022. http://doi.org/10.1007/s11633-022-1340-5

1 Introduction

Deep neural networks (DNNs) have achieved great success in many visual recognition tasks. They have largely improved the performance of long-lasting problems such as handwritten digit recognition[1], face recognition[2], image categorization[3], etc. They are also enabling the exploration of new boundaries, including studies on image and video captioning[4–6], body pose estimation[7], and many others. However, such successes are generally conditioned on huge amounts of high-quality hand labelled training data and the recently greatly advanced computational resources. Obviously, these two conditions are usually too expensive to be satisfied in most cost-sensitive applications. Even when people do have enough high-quality training data, due to the massive efforts of many annotators, it is usually a great challenge to figure out how to train an effective model with limited resources and within an acceptable time. Assuming that somehow the model can be properly trained (no matter how much effort it takes), it is still not easy to have the

Review
Special Issue on Brain-inspired Machine Learning
Manuscript received April 7, 2022; accepted May 26, 2022; published online August 18, 2022
Recommended by Associate Editor Chun-Hua Shen
Colored figures are available in the online version at https://link.springer.com/journal/11633

model properly deployed for real applications on the end users′ side, as the run-time inference has to fit the available or affordable resources, and the running speed has to meet the actual needs that can be real-time or even more than that. Therefore, besides accuracy, which is usually the biggest concern in academia, efficiency is another important issue and, in most cases, an indispensable demand for real applications.

Though most of the research on using DNNs for visual recognition tasks focuses on accuracy, there are still many encouraging progresses on the efficiency side, especially in the recent few years. For example, some survey papers have been published on efficiency issues for DNNs, as detailed in the following Section 1.1. However, none of them pays a major attention to visual recognition tasks, especially lacking coverage of special efforts to efficiently deal with visual data, which has its own properties, and the so-called third generation of efficient neural network models, which are inspired by human brains, i.e., spiking neural networks (SNNs)[8], are also lacking in discussions. In practice, efficient visual recognition has to be a systematic solution that takes into account not only compact/compressed networks, efficient dynamic inference, and hardware acceleration, but also proper handling of visual data, which may be of various types (such as images, videos, points, and brain-inspired events) with quite different properties. That might be an important reason for the lack of a survey on this topic. Therefore, as far as we know, this survey provides the first survey on efficient visual recognition algorithms with DNNs, particularly brain-inspired methodologies, including event data and SNNs. It targets a systematic overview of recent advances and trends from various aspects, based on our expertise and experiences with major types of visual data, their various recognition models, network compression algorithms, and efficient inference.

1.1 Related surveys

There are some related surveys published recently, but their scopes and contents are significantly different from ours.

Task-specific DNN models. A few surveys focus on the progresses of specific tasks, such as 3D data representation[9], texture representation[10], generic object detection[11], brain-inspired event-based vision[12]. Though they have conducted comprehensive reviews on the existing models for such specific tasks, which is very valuable for understanding the progresses on the model development side, the efficiency issue is unfortunately not their focus and thus lacks sufficient coverage and in-depth analysis.

General introduction of DNNs and efficiency strategies for model compression. Zhang et al.[13] have a much narrower coverage which can be good for the detailed directions it focuses on, but its distinctive categorization may confuse a certain audience. Deng et al.[14] are comprehensive and professional in both DNN compression and hardware design. In contrast to the above surveys, another line of model compression focuses on only the algorithmic part. For example, Cheng et al.[15] cover all major aspects of efficient DNNs but lacks advanced content, Lebedev and Lempitsky[16] focus mainly on convolutional neural network (CNN)-based models, and Elsken et al.[17] focus on the specific area of automatic network architecture search (NAS).

Efficient inference. An emerging research topic of dynamic inference has received extensive attention, which is dedicated to obtaining efficient inference by executing data-dependent adaptively dynamic computational graphs and parameters at the inference stage. Han et al.[18] focus on the dynamic neural network comprehensively in analog-activated DNNs with three main categories: sample-wise, spatial-wise, and temporal-wise. However, this survey lacks the analysis of brain-inspired spike-activated DNNs, such as SNNs[8], which are natural users of efficient dynamic computational graphs and parameters at the inference stage.

1.2 Contributions and organization

Compared with other surveys, this survey mainly focuses on the global efficiency of the production line from the raw visual data to the final recognition results, and it is expected to help the readers who are interested in the modern visual recognition tasks and their efficient DNN-based and brain-inspired solutions. This survey contributes in to following aspects, which are also novelties, to the best of our knowledge.

1) A systematic survey of existing advances on efficient visual recognition approaches with modern DNNs and brain-inspired SNNs, which is the first of its kind, as far as we are aware.

2) The first summary of data-related issues for efficient visual recognition, including data compression, data selection, and data representation.

3) A new investigation of network compression models from the perspective of benefiting visual recognition tasks.

4) A review of acceleration approaches for run-time inference in the scope of efficient visual recognition, particularly dynamic networks.

5) Insightful discussions on challenges, opportunities, and new directions in efficient visual recognition.

For clarity, the pipeline of this survey is shown in Fig. 1 as the blueprint of the structure of this paper. Specifically, in Section 2, we introduce the four main data types commonly concerned with visual recognition problems and discuss their properties and their challenges. Section 3 reviews the efforts on three aspects before the actual recognition part: data compression, data selection, and data representation. Section 4 briefly introduces and analyzes the widely studied directions for network compression within the scope of visual recognition. Section 5

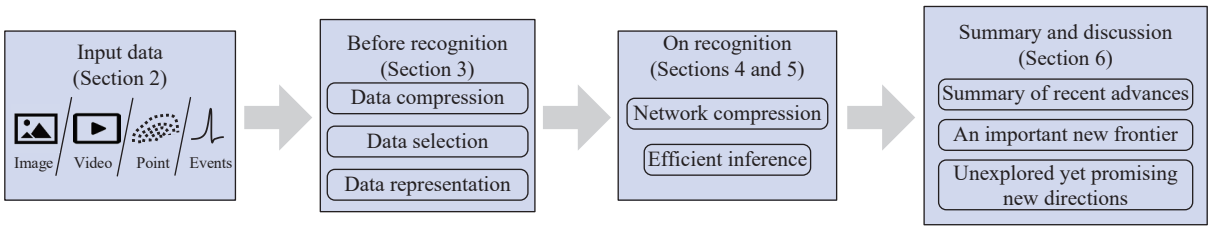


Fig. 1 Pipeline of this survey

provides a summary of recent progress in efficient model inference in the testing phase, which is very important for the real deployment of DNN-based and brain-inspired visual recognition systems. Finally, Section 6 outlines all efforts to generate a clear overall mapping and discusses some important uncovered aspects and new research directions.

2 Visual recognition: Data types, challenges, and preliminaries

Visual recognition covers several data types and a large number of detailed tasks. Meanwhile, the recognition methods also include variant-specific approaches. This section first introduces four commonly concerned visual data types, i.e., image, video, point, and brain-inspired event, with their properties and recognition challenges. Then the related efforts before and on the recognition discussed in the paper are also listed as brief preliminaries for the readers.

2.1 Data types

Images are the most studied visual data, probably due to their wide existence and relative simplicity of acquisition, storage, transmission, and processing. In many cases, they are both efficient and sufficient for sharing information visually. In contrast, as both the capturing devices and other related infrastructures and devices are greatly advanced and popularized, videos seem to be the most informative media. They have increased dramatically and will probably be even able to replace images in most scenarios soon. Videos appear very natural to humans, so they generally contain huge redundant information in the spatio-temporal domain. Points are usually sparse compared with images, but one dimension of them is higher and may have very large ranges. There are mainly two types of visual point data: point clouds and 2D/3D skeletons. The most valuable advantage of points is that they contain geometry information, so the shape is the main information for recognition tasks. Events or event streams are relatively special since brain-inspired dynamic vision sensor (DVS) cameras appeared late[19, 20]. To record only valid vision information and avoid motion blur, DVS encodes the time, location, and polarity of the brightness changes for each pixel at an extremely high event rate (1 M to 1 G events per second), just as in the biological neural system. The spatial sparseness and high temporal resolution of event streams have unique advantages in low latency and efficiency.

2.2 Challenges

It is clear that different data types have different characteristics, e.g., images: spatial information; videos: abundant informative and spatio-temporal information; points: spatial sparse, high dimension, and shape information; event streams: spatial sparse and spatio-temporal information. Thus, the challenges they must deal with in recognition tasks should also be discussed.

2.2.1 Images: Larger scale and deeper understanding

There are two clear trends in image-based recognition with DNNs. The first is that the scale of processed data increases quickly. As shown in Fig. 2, there is a clear history and trend that the benchmark dataset for developing new models has been shifting to larger scales and wilder contents (from MNIST[1] to CIFAR-10/CIFAR-100[21], and ImageNet[22]). With larger and more diverse training data, the trained DNN models can do more challenging recognition tasks, but it also brings greater challenges in efficient computation. The second is that the recognition is going toward deeper understanding and richer results. Traditionally, classification or categorization is most commonly concerned, but recently a lot of efforts and progresses have gone far beyond that, spreading to many tasks including detection, attribute extraction, keypoint/pose estimation, semantic segmentation, image captioning, visual question answering, and even to the visual genome extraction, as shown in Fig. 3. Such a trend greatly extends the research area and has attracted wider and stronger interests from both academia and industry. While new performance records are made on different tasks in a much shorter time, the demand for exploring proper acceleration approaches has become greater than ever, especially from the industry side, which is eager to apply the latest models to various real scenarios.

2.2.2 Videos: Redundant spatio-temporal information

Information redundant is the natural challenge of videos compared to images. In contrast, some of the visual challenges for images, such as occlusions and static background clutters, may get alleviated in videos when the motion information in the videos is properly treated. However, videos have their own particular challenges. A

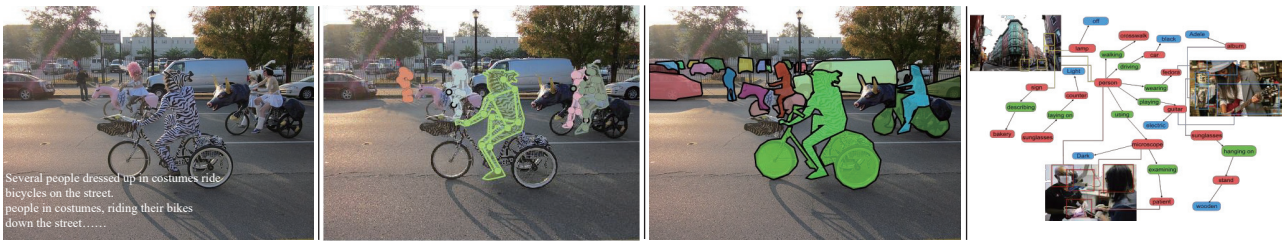
(a) MNIST: 70 000 handwritten digits

(b) CIFAR 10: 60 000 images in 10 classes with 6 000 images per class

(c) CIFAR 100: 60 000 images in 100 classes with 600 images per class

(d) ImageNet: 14 197 122 images in 21 841 non-empty synsets (From which about 1.4 M images of 1 000 synsets are commonly used for image classification.)

Fig. 2 Scale of images for recognition has increased greatly.



(a) COCO dataset, captions task

(b) COCO dataset, keypoints task

(c) COCO dataset, detection task

(d) Visual genome dataset, connecting language and vision

Fig. 3 Expected image recognition results are becoming deeper and richer, going far beyond simple classification.

major one is that the desired information is unevenly distributed in both spatial and temporal dimensions, as shown in Fig. 4. How to extract such sparse information from a large amount of data without getting confused is a great challenge. Meanwhile, the bigger data size (compared to images) and, in many cases, the need for real-time or even faster processing have made the efficiency issue even more important. Note that most of the new recognition tasks for images also exist for videos, which require extra effort for specific strategies on acceleration.

2.2.3 Points: Geometry and high dimension

Though not as popular as images and videos, point data also play an important role in visual recognition. Except for the point clouds obtained from radar or depth cameras (depth images can be easily turned into point clouds) for 3D object recognition[23, 24], and 2D/3D skelet-

Sampled video frames from a "Parallel bars" video of activitynet dataset

Fig. 4 Video recognition tasks usually target extracting few high-level semantics (e.g., category) from a large number of video frames, which are likely to have much redundant/irrelevant information.

ons used for action recognition[25–29], 3D CAD models have also been used for visual recognition, such as the ShapeNet dataset[30]. In a general sense, they can also be regarded as point data, as the CAD polygonal models can be turned into voxels and point clouds when needed. Therefore, they are also included in the data samples as shown in Fig. 5. Since the geometry information in points is usually very informative for recognition, maintaining such 3D geometry information while improving the efficiency of models is critical and is not easy, as the dimensional range might be very large.

2.2.4 Event streams: Spatial sparse and very high-rate temporal information

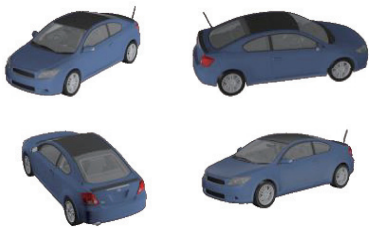
Although DVS cameras have become commercially available only since 2008, they pose a new paradigm shift by using sparse and asynchronous events (events can be seen as binary signals with position) to represent visual information[12]. Unlike conventional cameras, which produce fixed low-rate synchronized frames, DVS cameras exhibit advantages mainly in three aspects[31]. Firstly, DVS cameras require fewer resources, as the events are sparse and only triggered when the intensity changes. Secondly, the μs temporal resolution of DVS can avoid motion blur by producing high-rate events. Thirdly, DVS cameras have a high dynamic range (140 dB versus 60 dB of conventional cameras) for various challenging illumination conditions. These characteristics bring advantages over conventional cameras when orienting to visual tasks that require low latency, low power consumption, and stability for variant illumination, which have been used in high-speed object tracking[31], autonomous driving[32], simultaneous localization and mapping (SLAM)[33], low-latency interaction[34], etc. Event streams only have event-based data (0 or 1), so they are similar to the points data at the level of each sampling point, as shown in Fig. 6. The most critical trait of events is that their high temporal resolution acts as the main role, which is nonexistent in traditional points and other data types. Therefore, how to effectively and efficiently extract information from sparse, non-uniform, and high-rate event streams is a great challenge.

2.3 Preliminaries

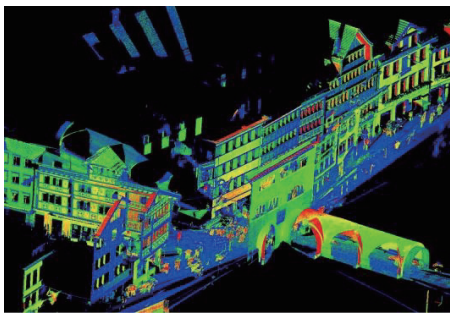
In view of the variety of methods that will be introduced and discussed in the following content of this survey, it is necessary to make brief preliminaries to show most of the representative methods with their characteristics. Hence, Table 1 is designed to give a clear view from data processing (Section 3) to real deployment (Section 5), and the location of each method is also given. Please note that some trivial practices cannot be exhibited here due to space limitations.

3 Before recognition: Efforts on the data

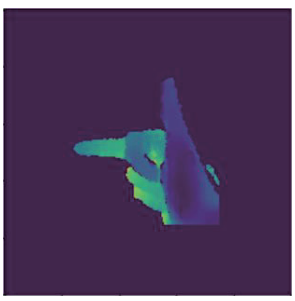
Visual data have their own properties, which can be made use of when efficient recognition models are designed. Many efforts can be made to change/map the



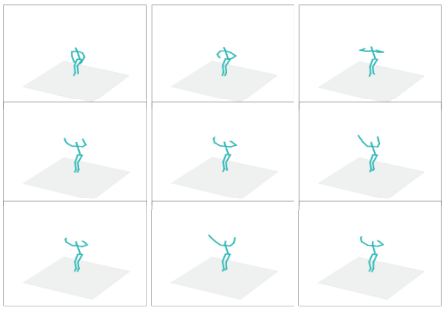
(a) 3D CAD polygonal models from ShapeNet dataset



(b) Point cloud from SEMANTIC3D.NET benchmark



(c) 3D depth data from BigHand2.2M dataset



(d) 3D Skeleton data from NYU-RGB+D dataset

Fig. 5 Examples of point data, in its general sense

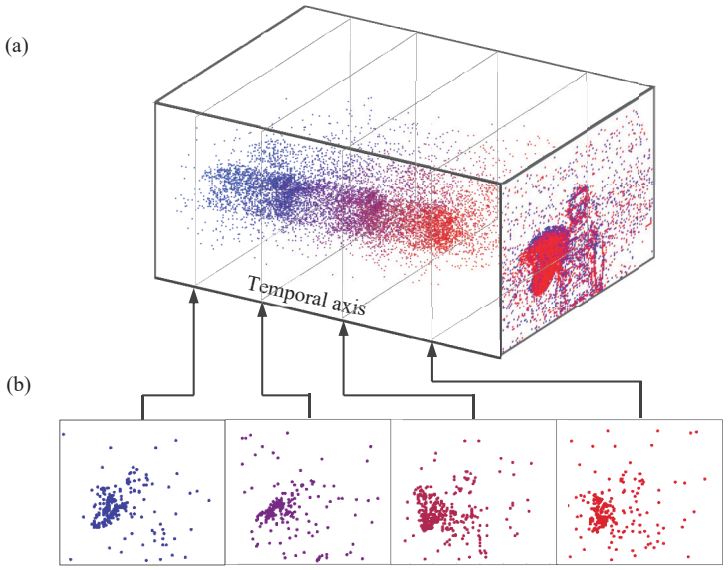


Fig. 6 An example of DVS-captured dataset named DVS gesture: (a) Spike pattern recorded by DVS; (b) Slices of spike events at different timesteps. The blue and red colors denote the On and Off channels, respectively.

Table 1 Preliminaries of the efficient methods mentioned in this survey

Task			Method	Characteristic	Location
Efforts on data	Data compression	Images	Scale-invariant feature transform (SIFT)[35], discrete cosine transform (DCT) domain[36], Wavelet transform[37]	Greatly optimized and easy to be used directly	Section 3.1.1
			Encoder-decoder networks[38–41]	Ensure both the relevant information and speed	Section 3.1.1
		Videos	H.265[42] and H.264[43]	Can not be optimized end-to-end	Section 3.1.2
			Coding by CNN[44], Kalman filtering network[45], frames reconstructed by CNN+long short term memory (LSTM)[46]	Must restore the compressed data to a raw video format and bring lots of extra cost	Section 3.1.2
			Deep feature flow[47], direct training[48]	End-to-end and real-time process	Section 3.1.2
		Points	Auto-encoder-based geometry codec[49–51]	Aiming at geometric characteristics	Section 3.1.3
		Events	Transforming the events as groups with multiple styles[52–54]	Necessary to yield signal-to-noise ratio	Section 3.1.4
	Data selection	Images	Subsampling by active learning[55, 56], gradient method[57], or clustering[58]	Noises and redundant samples can be reduced to prevent overfitting	Section 3.2.1
		Videos	Random keyframes selection[59–62]	Minimum computation cost but missing information	Section 3.2.2
			Keyframes prediction by reinforcement learning[63–66], adaptive pooling[67], memory-augmented LSTM (MALSTM)[68]	Select high-quality keyframes and avoid to process redundant frames	Section 3.2.2
		Points	Sampling with modeling attention[69–71]	Superiority over traditional or neural methods	Section 3.2.3
		Events	Event stream denoising[72–74]	Mostly denoised when generated	Section 3.2.4
			Dynamically selection with attention[52]	Scarce and promising	Section 3.2.4
	Data representation	Images	Pre-trained model on other datasets[75–78]	Make the entire training process efficient	Section 3.3.1
		Videos	I3D[60], 2DCNN + 1D temporal convolution[79]	Frames representation and temporal correlations	Section 3.3.2
			Dynamic image[80, 81]	Turn a whole video into one single informative image	Section 3.3.2
		Points	Align the coordinates dimension[82, 83]	One dimension is reduced	Section 3.3.3
		Events	Frame-based[34, 52], graph-based[53], point-based[54]	There are alternative representations	Section 3.3.4

Table 1 (continued) Preliminaries of the efficient methods mentioned in this survey

Task			Method	Characteristic	Location
Network compression	Compact networks	CNNs	Light receptive field[84, 85], topology[86, 87], or block[88, 89]	Some compact designs become standard neural structure such as bottleneck and depthwise convolution	Section 4.1.1
		RNNs	Simpler units[90, 91] or architectures[92, 93]	Hard to implement, limited compression ratio	Section 4.1.2
		NAS	Reinforcement learning[94], evolutionary algorithm[95], Bayesian optimization[96], gradient-based[97]	Can surpass human designs in both accuracy and efficiency, promising but still needs further studies	Section 4.1.3
	Tensor decomposition	Tucker	CP-CNN[98], Tucker-CNN[99], BTD-LSTM[100]	Curse of dimensionality and complex computation	Section 4-B1
		Tensor network	TT-CNN[101–103], TT-RNN[104, 105], TC-RNN[106], HT-RNN[107, 108]	High compression ratio, in situ training, hard to avoid accuracy loss	Section 4.2.2
	Data quantization	Projection	WAGE[109], full 8-bit training[110]	Project floats to distributed integers, mainstream way	Section 4.3.1
		Optimization	XNOR-NET[111], AutoQ[112]	More attention to the whole network	Section 4.3.1
	Pruning	Search	Low-precision estimation[113, 114], negative activation prediction[115]	Vast computing time, extra indices of pruned weights or neurons	Section 4.4.1
		Optimization	Structured sparsity[116], ThiNet[117], SSR[118]	Adaptive to large DNNs, structured pruning	Section 4.4.1
	Joint compression		Decompose + quantize[119, 120], quantize + prune[121, 122]	Extremely high compression ratio, maintaining accuracy is critical	Section 4.5.2
Efficient inference	Fast inference	Data-aware	Recurrent residual module[123], efficient inference engine[124], scale-time lattice[125]	Efforts on reducing the computation on the redundant data are important for inference	Section 5.5.1
		Network-centric	Prune whole blocks[126], dynamic compression ratio[127], integrate resource and input[128]	General network-centric compression for fast inference should be evaluated by the proposed key property indicator (KPI)	Section 5.1.2
	Dynamic inference	Analog-based	Dynamic structure[129–131], dynamic parameters[132–135]	Dynamic networks adapt their structures or parameters to different inputs	Section 5.2.1
		Brain-inspired	Spiking neural networks[18, 52, 136]	Has the spatial-wise, temporal-wise and sample-wise dynamic	Section 5.2.2

data to a more compact form before applying the recognition models. These efforts may be grouped into three categories based on their functionalities: data compression (reducing redundancy and irrelevance), data selection (reducing irrelevance), and data representation (increasing compactness). In this section, we overview and summarize existing advances in all three aspects and organize the contents for each of them according to their related data types. In doing so, their motivations and strategies can be easily understood and searched for.

3.1 Data compression

As datasets and networks grow in size, large memory consumption and high computational complexity have made the training of deep neural networks a challenge and hindered the popularity of AI. Specifically, for deep learning, there are four types of consumption (as shown in Fig. 7) due to large data sets: 1) Storage consumption, most of the commonly used datasets are now hundreds of gigabytes in size, the situation of which puts a lot of pressure on storage hardware; 2) Transmission consumption, the transmission of large amounts of image/video data can be a challenge to network bandwidth; 3) Memory consumption, when training a network, usually the larger the batch size (the amount of data fed into the network each time) is, the better the performance is, and thus a larger memory is always desired; 4) Computing consumption, DNNs usually need to rely on powerful computing resources (e.g., a GPU), especially when large datasets have to be handled, and such a demand has a rising trend. Increasing data size may lead to better recognition performance, but it can also increase all four types of consumption.

3.1.1 Image compression

Image compression and image recognition can be linked together to save the consumption on image decompression, which has been shown to be more efficient and, in many cases, can be made more effective. A general framework for decompression-free joint compression and recognition is shown in Fig. 8, which illustrates both the training and test/inference phases. While the recognition module in the framework is generally DNN-based in the scope of this survey, the compression module can be either hand-crafted or learning-based. Though many researchers may expect a purely learning-based (DNN-based) framework, the current reality is that the sophisticated hand-crafted image compression models are still

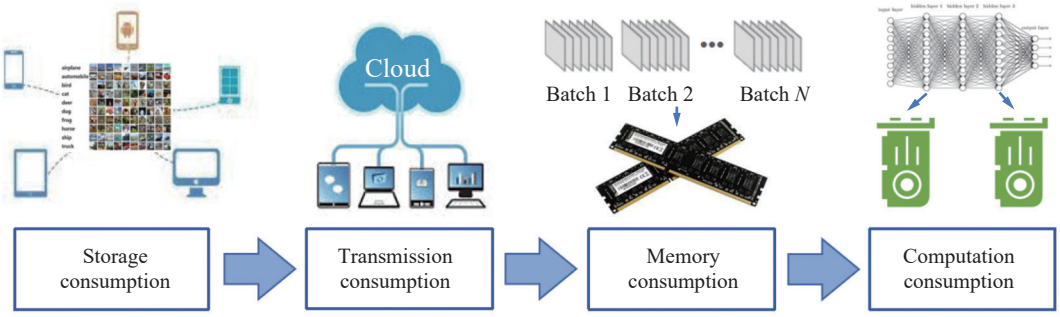


Fig. 7 Major types of consumption in deep learning

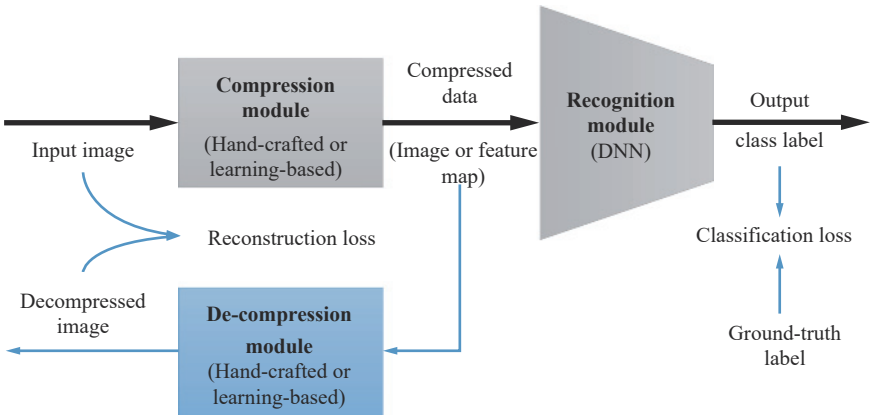


Fig. 8 A general framework for joint image compression and recognition. The upper row in black shows the inference (test) phase, while the whole figure (including the blue parts) describes the training phase. Note that the compression module has to be optimized to minimize both the reconstruction loss and the classification loss, different from the recognition module, which is only optimized for the latter one.

more popular. It is not easy to invent a new DNN-based compression model that outperforms the greatly optimized hand-crafted models. Researchers who know much more about recognition than compression usually tend to keep the compression module as is and put more effort on how to make their recognition modules better to receive the compressed data.

Hand-crafted lossy compression. Although several more complicated compression standards have been developed, including JPEG, WebP, and BPG, JPEG remains the most widely used for lossy image compression. Recently, many of interesting works [35–37, 137–139] have been developed to learn the feature distribution directly from compressed data. Wu et al.[35] formulate the popular SIFT feature extraction in JPEG's discrete cosine transform (DCT) domain, which indicates that effective visual features can be directly extracted from the compressed data. Ehrlich and Davis[137] introduce a general method to learn a residual network in the JPEG transform domain. Liu et al.[138] combine compression and recognition based on JPEG transformation, and it has been proved by experiments that it could achieve 3.5× compression rate improvement, while its consumption is only 30% of the conventional JPEG without classification accuracy degradation. In [36], it is proposed to learn DNN parameters directly from the DCT coefficient of JPEG compression, which is 1.77× faster than ResNet-50 at the same accuracy. Javed et al.[139] propose a method for reviewing document images, which is directly based on compressed document data. In [37], the Haar wavelet transform is utilized to compress and decompress high-resolution iris images, enabling fast and accurate iris recognition.

Learning based lossy compression. Although traditional compression algorithms are carefully constructed, there is still room for improvement in compression efficiency. For example, the conversions are fixed and cannot be adaptive to fit different inputs. In addition, predefined quantization implementations can result in data redundancy. Moreover, limited by manual design, the algorithms are usually hard to be optimized for a specific metric, even if the metric is a perfect assessment of image reconstruction quality. Therefore, more attention has been paid to learning based compression methods in recent years. Different from traditional methods, in a learning-based approach, the parameters of the neural network are automatically learned from a large amount of data by definite optimization objectives to deal with specific situations. A general pipeline is that the input image x is firstly processed by the analysis network (encoder) to generate compressed feature-maps, which are then converted into a set of bit-streams by quantization and lossless arithmetic coding. After that, they are used to generate a recovered image $\hat{x}$ by a re-factoring net-

work (decoder), and the entire network is trained end-to-end until convergence. On the basis of this pipeline, many excellent image compression methods[38–41] have been proposed.

Joint compression and recognition. Stimulated by the tremendous memory reduction caused by compression algorithms, some methods[140–143] combine compression with recognition to improve effectiveness and efficiency. Specifically, in order to save the decompression consumption, a well-learned analysis network could be directly cascaded with a downstream recognition network, as shown in Fig. 9. Such a joint optimization not only tries to retain as much classification-relevant information as possible during compression, but also accelerates the speed of inference and optimizes the consumption of training compared to models directly trained on the input images. Detailed models can be found in [140–143].

3.1.2 Video compression

As the most common media, videos were said to take more than 70% of all Internet traffic, according to the white paper of "Cisco Visual Networking Index: Forecast and Methodology, 2016–2021", and now the percentage is probably even greater. In the past few years, many representative advances[144–148] have been made in video-based recognition. These methods, however, all focus on designing a special neural network for analyzing frames, ignoring the fact that videos are in a compressed format during transmission and storage. Therefore, extra time and storage are needed for decompression before analysis. In order to improve the effectiveness and efficiency of video recognition, it is necessary to apply efficient compression, and decompression beforehand or directly use compressed data for recognition.

Compression algorithms. As the most popular video compression algorithms, some highly efficient compression standards such as HEVC(H.265)[42] and AVC(H.264)[43] have been used for a long time. Taking the H.264 algorithm as an example, three kinds of frames are defined in the encoding protocol. The fully encoded frame is called I-frame (keyframe), and the frame containing only the difference partial encoding generated by the I-frame is called P-frame. The highly compressed frame obtained using both previous and forward frames for data reference is called B-frame. There are two core algorithms used by H.264: intra-frame and inter-frame compression. Among them, intra-frame compression is an algorithm for generating I-frames, and inter-frame compression can generate highly compressed B-frames and P-frames.

To achieve inter-frame compression, H.264 relies on many hand-crafted modules, such as the DCT transform module, block-based motion estimation module, and motion compensation module. Although these modules are well designed, they are not optimized end-to-end.

Recently, several DNN-based video compression methods have been proposed for intra prediction & residual coding[44], post-processing for predicted frames[45], inter-frame interpolation[46], and full network-based video compression[149, 150]. Chen et al.[44] designed two convolutional neural networks to encode predicted images and residual images, respectively, and the reliable experimental results prove that deep neural networks can achieve better results than hand-crafted modules. Lu et al.[45] model the video artifact reduction task as a Kalman filtering procedure and restore decoded frames through a deep Kalman filtering network. By constructing a recursive filtering scheme based on the Kalman model, more accurate time information can be used to obtain better reconstruction quality. Wu et al.[46] regard the video compression challenge as a repeated image interpolation challenge so that the remaining frames can be reconstructed from the keyframes through an interpolation reconstruction network. In addition to this, their algorithm provides a compressible code to disambiguate different interpolations and encode keyframes as faithfully as possible. However, although the interpolation network is end-to-end optimized, motion information still requires additional calculations, which depend in part on other algorithms. In [149], an end-to-end video compression deep model that jointly optimizes all the components for video compression has been proposed. Specifically, the optical flow estimation network is used to obtain motion information, and the compressed network is used to compress both motion information and residuals. These two different networks are jointly learned through end-to-end optimization. Habibian et al.[150] present a depth generation model for lossy video compression, which consists of a three-dimensional automatic encoder with discrete potential space and an autoregressive prior for entropy coding. The self-encoder and the transcendental encoder are trained jointly to

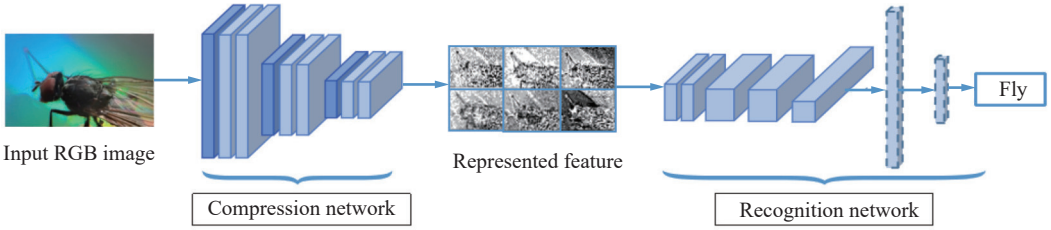


Fig. 9 The input image is represented as a series of compressed feature maps, and the subsequent recognition network learns classification information from these feature maps.

achieve the best rate-distortion curve. This method is superior to the latest learning video compression network based on motion compensation or interpolation. In addition, three extension directions are proposed: semantic, adaptive, and multimodal compression. These directions mentioned above will undoubtedly lead to new video compression applications, which may be realized by DNNs but not classical codecs.

Utilizing compressed data. Due to the time redundancy and the enormous size of data streams, there is a large amount of redundant or irrelevant information in the video data, which makes learning neural networks difficult and slow. Therefore, most video recognition algorithms use a video compression algorithm such as H.264 as pre-processing, which can reduce superfluous information by two orders of magnitude. In recognition, the mainstream method is to restore the compressed data to a raw video format and then handle each frame as an RGB image. Considering that neural networks can learn features from data, the process of decoding compressed data may be skipped.

Zhu et al.[47] present a fast and accurate video recognition framework, namely deep feature flow. It extracts depth features on keyframes (I-frames) through a convolutional network and maps them to other frames for auxiliary prediction through the flow field. In this process, significant efficiency gains can be achieved by reducing the amount of computation. In [48], a network trained directly on compressed videos is proposed, as shown in Fig. 10. The benefits of this design are three-fold. Firstly, the compressed video representation removes large amounts of redundant information and preserves useful motion vectors. Secondly, compressed video representations are more convenient for exploring video correlation than individual images. Finally, such an approach is more efficient because only informative signals are processed rather than near-duplicates, and the efficiency can also be improved by skipping the steps of decoding the video as the video is stored in a compressed version.

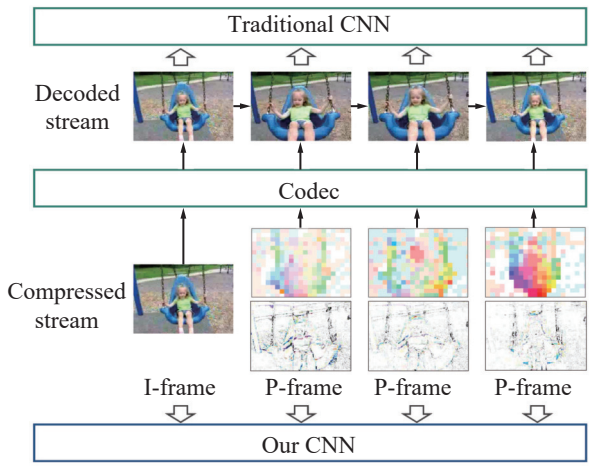


Fig. 10 A recently proposed video recognition network, which was directly trained on compressed videos[48]

In general, video compression technology has already played a significant role in video-based recognition tasks, no matter whether it is about a traditional video compression standard or a network-based learning algorithm. In most user-oriented application scenarios, such as public security monitoring and real-time scene replacement, real-time processing is usually a must, and stability of the algorithm is a desire. In these cases, video compression has been shown to have great importance and application potential.

3.1.3 Point cloud compression

Different from image/video data, point cloud contains more complex structures, which makes the compression much more difficult. The most efficient way of compressing the point cloud is to utilize its geometric characteristics. Quach et al.[151] present a novel data-driven geometry compression method for static point clouds based on learned convolutional transforms and uniform quantization. The first auto-encoder-based geometry compression codec is proposed in [49], where the point cloud is treated as input rather than voxel grids or collections of images. Inspired by the great success of variational auto-encoders (VAE) in image/video compression, Wang et al.[50] present a DNN based end-to-end point cloud geometry compression framework, in which the point cloud geometry is first voxelized, scaled, and partitioned into non-overlapped 3D cubes which is then fed into a 3D convolutional networks for generating the latent representation. Furthermore, Yang et al.[51] present a novel and interesting end-to-end deep auto-encoder to achieve the state-of-the-art performance for supervised learning tasks on point clouds. In this work, a graph-based enhancement is firstly enforced (in the encoder) to promote local structures and form a low dimensional codeword, which can be used to deform/fold a canonical 2D grid and then reconstruct the 3D object surface of the input point cloud. The learned encoder can be treated as a compression module, which has been proved to have great generalization abilities: experiments on major datasets show that the folding can achieve higher classification accuracy than other unsupervised methods with significantly fewer parameters (7% parameters of a decoder with fully-connected neural networks). The research on learning point cloud compression has just started and was tested on recognition tasks, and this new topic remains largely unexplored. Encouraged by the successes of these pioneering works, more and bigger advances can be expected in the coming near future.

3.1.4 Event stream compression

Data compression is an optional operation for the above three kinds of data, while it is necessary for the event stream. Unlike image/video/point data, the event stream contains very abundant temporal information because of the μs level temporal resolution, which makes the execution of data compression work on the temporal axis. The number of events processed simultaneously determ-

ines the latency of task output, and plays an essential role in the way events are processed[12]. One method is handled event-by-event, and the other method operates on groups of events where several events are processed together. To yield sufficient signal-to-noise ratios (SNR) for the task accuracy, compressing the events at the temporal axis as groups is the most common method. In the field of event-based vision, data compression is equivalent to transforming the event streams into alternative representations, such as frame-based representation with CNN[34] or spiking neural network (SNN)[52], graph-based representation with graph-based convolutional network (GCN)[53], point-based representation with PointNet[54], etc. We will describe the specific compression (representation) method in Section 3.3.4.

3.2 Data selection

Selecting only relevant and informative parts from the raw data for recognition can significantly reduce the computational cost and may also lead to a better recognition accuracy (as disturbance by irrelevant information/noise is reduced). Due to the great differences between the data types in terms of data structure and information characteristics, the research focus and the methods applied are also quite different. For image data, sample selection for training is the main concern. In the case of video data, frame selection inside each video (for both training and testing) matters most, as continuous video frames contain a lot of redundant information. Similarly, point data also have much redundancy, and subset sampling is the main-stream. By contrast, the selection of events focuses on denoising because of the unique way of visual information acquisition. Details on the motivations, strategies, and methods for them are given below.

3.2.1 Image data

Currently, DNNs are generally data-hungry, namely, the more labeled training data, the better performance they can achieve. However, more data also means more costs, including the effort for data acquisition and labeling and all the consumption (storage, transmission, memory, and computation) for learning. Given a fixed amount of training data, directly scaling up the computation (by increasing the number of parameters and/or doing more iterations) usually has a performance upper bound, and more computation after that goes more towards overfitting. Such overfitting is believed to be the result of the inherent noise (or certain redundant samples in a softer tone) in the training data[55]. Therefore, reducing such noise or redundant samples not only saves consumption for model learning, but also has the potential to even boost the model′s performance.

Subsampling the training data has recently been studied under such motivation, and it has already shown quite promising results in the past couple of years. So far, such research has only been concerned with image recognition, and subsampling can be regarded as data selection; we introduce them here. In a more general sense, these approaches shall also be applicable or at least have the potential to be made applicable for other data types. Since DNNs prefer large data, naive random subsampling likely ends up with inferior performance. Therefore, some efforts have focused on how to get subsets better than randomly sampled ones or revealing the inequality of training samples. Core-set selection[56] and representative subset finding[57] are good examples in this direction. However, doing better than random subsampling does not guarantee any drop in performance. More recent works improve these by identifying redundant samples for removing them without sacrificing the recognition performance. Clustering in the DNN feature space[58] shows a successful removal of 10% semantically redundant samples from CIFAR-10 and ImageNet datasets, while a slightly later work on “select via proxy” is able to push this reduction to 40% on CIFAR-10 with no performance loss with the help of three uncertainty metrics. The very recent work on active dataset subsampling (ADS)[55] presents even more encouraging results: Removing 50% of CIFAR-10 training samples yet performing slightly better than training on the full dataset. Similarly, on the ImageNet dataset, they are able to outperform training with all data by using only 80% of it. As shown in [55], the advantages of subsampling are not limited to its ability to maintain or even improve recognition performance whilst saving all learning consumption. It may be applicable to many tasks and various data size settings. An important and highly valuable problem is how to do that with noisy or weak labels, as collecting high-quality labels is usually a great challenge.

3.2.2 Video data

Since the huge redundant information in the spatio-temporal domain is prime for videos, instead of performing expensive processing on every frame to approach target tasks, such as object detection and action recognition, selecting keyframes and performing the major processing sparsely is a more efficient choice (see Fig. 11 for the motivation). However, how to efficiently select proper keyframes remains an open issue. Since relevant and discriminative video information could be unequally located in its temporal domain, obtaining a few high-quality keyframes without losing that information could cost significant extra computation. How to do that efficiently is an important issue for video-based recognition tasks.

Generally, existing approaches to selecting keyframes could be divided into two types. The first is to randomly pick up keyframes at predefined intervals. It is commonly applied in video recognition tasks, which may also include temporal boundary detection[59–62]. Although this method takes the minimum computation cost, picking up frames by a large interval may miss critical information, while using a small interval could increase the post hoc computation costs. Alternatively, in another type of ap-

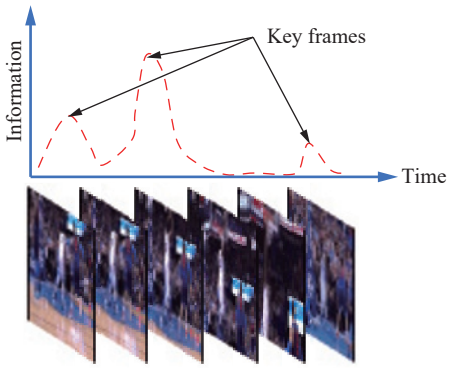


Fig. 11 Video keyframes which are informative for recognition could be sparse and unequally located.

proach, the information from processed frames can be used to predict the possible keyframes in the future, which could avoid processing many redundant frames and select high-quality keyframes[63–65, 67, 68]. The sampling and early stopping can be learned by an end-to-end deep reinforcement learning model, as studied in [66].

3.2.3 Point data

Generally, directly working on all the point data can be heavy and expensive for all the types of consumption mentioned earlier. On the contrary, the shape or skeleton is the main object for recognition tasks. Usually, the same shape may still be recognizable with only a subset of points collected from the sensors. In many cases, only a small portion of the data is relevant for the recognition task. Therefore, recently quite a few researchers have tried to embed data selection components/concepts in their DNN models for handling point data to achieve better performance in terms of effectiveness and efficiency.

The selection of point data is usually made by sampling or modeling attention. A simple strategy for sampling is called furthest point sampling (FPS), which can be done efficiently[152]. However, FPS has significant weaknesses of being task-dependent, low-level (can not handle semantically high-level representations), permutation-variant, and sensitive to outliers. Recently, quite a few works have explored new sampling models for point clouds with the help of DNNs. Within them, the work "Learning to Sample"[153] introduces a neural network termed S-NET, which takes a point cloud and produces a smaller one that is optimized for a particular task. The simplified point cloud is not guaranteed to be a subset of the original point cloud, but a post-processing step is adopted to match it to a subset of the original point set. S-NET has a space consumption linearly proportional to its output point set size and offers a trade-off between space and inference time. As an example mentioned in the paper, cascading S-NET that samples a point cloud of 1 024 to 16 points with PointNet[154] reduces the inference time by over 90% compared to running PointNet on the complete point cloud, with only a 5% increase in space and 4% decrease in recognition accuracy (much better than FPS and random sampling). There are also several other works on attention modeling, such as the attentional PointNet[70] for 3D-object detection and the point attention network[71] for gesture recognition, etc. Due to the space limitation, only a brief description is provided in this paper. Since both the research and applications on point data are now growing very rapidly, it is expected that there will be an explosive growth of interest and efforts in designing more efficient and effective recognition models.

3.2.4 Event data

In contrast to the motivations of data selection in the above three kinds of data, one of the direct motivations for event selection is that there is too much noise in the event stream. All vision sensors are noisy because of the inherent shot noise in photons and transistor circuit noise, and this situation is especially true for DVS cameras. Hence, denoising is essential to work for event streams. The key point of the noise cancellation technique is identifying whether an event is a noise. Some typical methods include exploiting the motion consistency in the event stream[53], modeling the randomness of noise[72, 73], recovering events that are mistakenly classified as noise[74], etc. Due to the importance of event stream denoising, most event-based datasets are denoised when they are generated, and users generally do not need to consider noise when processing event data.

Event streams also have huge redundancies in the spatio-temporal domain, similar to the video data. Obviously, the event streams are sparse and non-uniform, which is caused by the unique dynamic vision sense paradigm and irregular dynamic scene changes. A recent work termed temporal-wise attention (TA)-SNN verifies this point[52], as shown in Fig. 12. With the help of temporal-wise attention, TA-SNN uses binary attention scores to mask parts of input event streams. Experimental results demonstrate that TA-SNN can get similar or even better performance with only half of the input events. Therefore, carefully reducing such selected events shall not only be efficient for the event-based task, but also have great potential to improve the model's performance. How to effectively and efficiently deal with the event streams by exploiting the sparse and non-uniform characteristics is of great value and has various real-life applications. The current exploration in this direction is scarce, and we think that data selection for event data is a promising area to investigate.

3.3 Data representation

3.3.1 Image data

Image representation for recognition in the DNN era is usually part of the representation learning network, not a separate pre-recognition operation. However, when there is not enough training data (i.e., a small training set) or only a small part of the training data gets labeled (i.e., a semi-supervised setting), it can be an effective and also

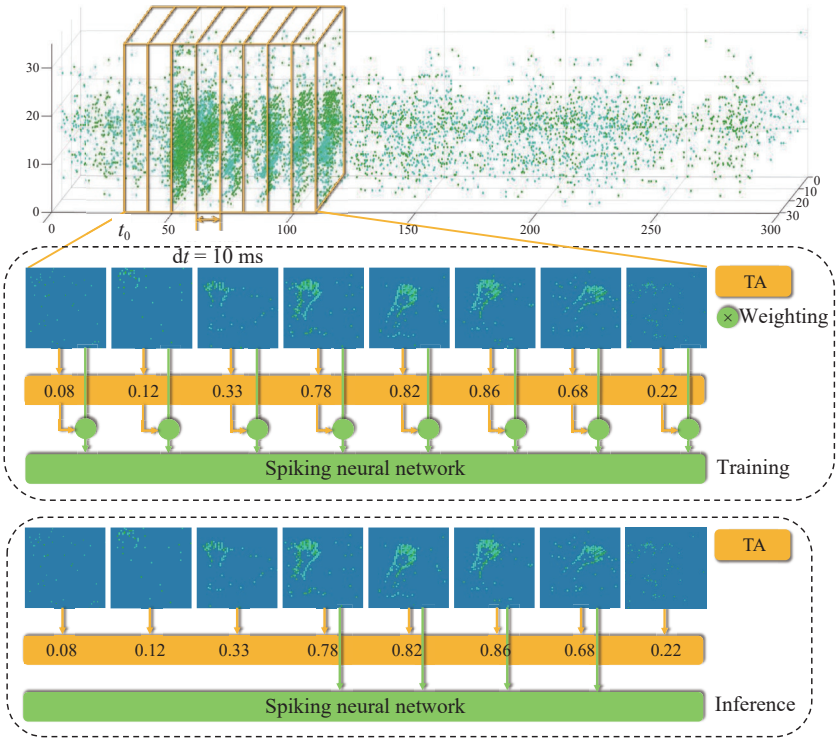


Fig. 12 Only parts of the events are selected with the help of attention.

efficient choice to borrow image representation from some suitable pre-trained model (trained on some source data) or from a model trained under an unsupervised setting on the target data before the main task-related training with the limited labeled data.

There are some representative references for these two cases. For the first one about borrowing representation from pre-trained models, a typical scenario is medical image analysis or recognition, where people have to face the reality of insufficient training data for many reasons, including the privacy issue. A highly cited work in the field[75] discussed full training versus fine-tuning (with pre-trained models) for medical imaging applications in three specialties (radiology, cardiology, and gastroenterology) on three vision tasks, including classification, detection, and segmentation with extensive experiments, and concluded that the use of a pre-trained CNN with adequate fine-tuning outperformed or, in the worst case, performed as well as a CNN trained from scratch, while at the same time enjoying the benefit of being more robust to the size of training sets (with the help of a layer-wise fine-tuning scheme). A similar conclusion was researched in a study on the effectiveness of using pre-trained CNNs as feature extractors for tuberculosis detection[76], where three different ways of utilizing pre-trained CNNs (with three different CNN structures) are discussed, and, in some cases, directly using pre-trained CNNs can even beat their fine-tuned versions. For the second case where data are sufficient but available labels are scarce, pre-training a network under the unsupervised setting, e.g., using a spatial prediction task with "Contrastive Predictive Coding", has been shown to be effective for fast further training in recognition tasks with little labeled data[77], which is called "data-efficient" as it allows task-related training on only a small account of data. Computationally, the task-related training shall also be very efficient, as it can inherit the weights from a pre-trained model, which is task-independent and can be obtained beforehand. Actually, besides the models trained under an unsupervised setting, models trained on a large general dataset (e.g., ImageNet) were also found to be very effective for image representation for many visual tasks, superior to well-designed hand-crafted features[78], and such direct adoption of pre-trained models has served as a good baseline for exploring new DNN models.

3.3.2 Video data

Data representation for efficient video-based recognition seen has two representative trends in recent years. Both of them are very new and look rather promising.

One is decomposing video sequences into individual frames for frame-based representation and then representing the motion information by exploring temporal correlations among the high-level features of frames. This is contradictory to the slightly earlier work (Carreira and Zisserman, CVPR 2017[60]) on the inception 3D (I3D) architecture which is about 3DCNN. Wu et al.[79] proposed to apply 2DCNN on individual video frames and then do computationally highly efficient 1D temporal convolution on the extracted 2DCNN features, which is both more effective and much more efficient than 3DCNN models for the task of video-based person re-identification. Later, Xie et al.[61] replaced the 3D convolutions at the bottom

of the 3D CNN network with low-cost 2D convolutions and temporal convolution on high-level "semantic" features (outputs of those 2D convolutions) and also got better performance and faster speed for action classification. Zhao et al.[155] further found that making the temporal convolution along the feature trajectories so that the representation can be robust to deformations, and thus they got improved accuracy on action recognition.

The other representative trend is using pooling to generate a super compact and sparse representation for videos before feeding them into recognition models. A very successful example in this direction is "dynamic image"[80, 81], which is generated by an efficient and effective approximate rank pooling operator, turning a whole video into just one single highly informative image. The dynamic image can simultaneously capture foreground appearance and temporal evolution information, while at the same time excluding irrelevant background appearance information. Therefore, even a simple 2DCNN model built on top of it can generate superior RGB video recognition results. Soon, the model got extended for RGB-D video-based activity recognition and showed good results[156], and very recently, it has also been used for generating multi-view dynamic images for the task of depth video-based action recognition[157].

3.3.3 Point data

In order to perform visual recognition tasks on point data, traditional works apply CNN with the same dimension as point data[158–160], which comes with huge computation costs, as shown in Fig. 13(a). However, compared to RGB data, point data might be inherently sparse. Motivated by such a property, computationally efficient networks are developed. A general approach is to align the coordinates dimension to the CNN channel dimension so that one dimension is reduced compared with the original point data, as shown in Fig. 13(b). Nonetheless, such an approach arises a problem: The index-adjacent points could be locally uncorrelated in the spatial domain no matter how to assign the point indices. Since the RNN/CNN inherently assumes local correlation exists, it is inappropriate to directly process locally uncorrelated features. Therefore, it is preferred to model all points simultaneously in the network and let the network automatically learn the proper relationship between them[82].

HCN[83] and PointNet[154] are among the most efficient and superior networks in skeleton-based action recognition and point-cloud-based object recognition, respectively. Although they may look different at first glance, they all align the coordinates dimension to the CNN channel dimension to make the computation efficient and simultaneously model all points to avoid the influence of point orders.

3.3.4 Event data

Due to the requirement of input SNR, events are often transformed into various representations that help to extract meaningful information to solve a given task. The

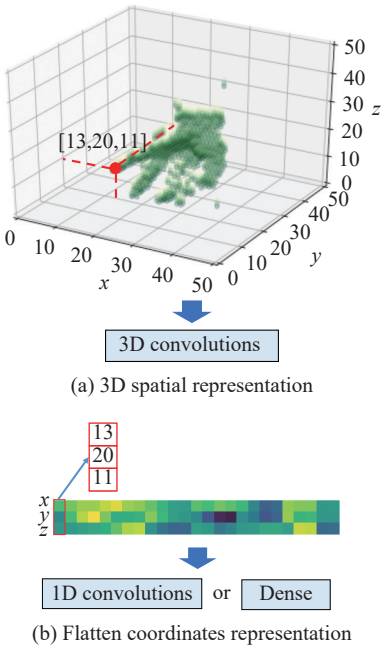


Fig. 13 Two kinds of point data representation and the corresponding neural networks for their processing

representation of the event data is highly related to the handling method. Here we only review popular deep learning representations, please refer to Gallego et al.[12] for a comprehensive review of event-based representation.

The event-based processing can be viewed as a trade-off between output latency and task accuracy. The number of events processed simultaneously is important since the more input is fed, the better performance will be achieved. Currently, DNNs are usually adopted to process event data. They adopt alternative representations such as frame-based representation with CNN[34] or spiking neural network (SNN)[52], graph-based representation with graph-based convolutional network (GCN)[53], point-based representation with PointNet[161], etc. Fig. 14 illustrates the three main styles of representations, i.e., frame-based, graph-based, and point-based event streams. Aggregating event streams into frames has an intuitive interpretation and is naturally compatible with the traditional computer vision framework. Temporal resolution is a crucial hyper-parameter for frame-based representation, which can be used to control latency and accuracy. Thus, transforming event streams into frames is a flexible and simple way of event processing. In contrast, graph-based representation aims to represent the stream of events as a graph and perform convolution on the graph for object classification. The compact graph representation can reduce computation and memory requirements while paying the price of latency. Moreover, events in a spatio-temporal neighborhood can be treated as points in 3D space. This is a sparse representation and is used in point-based geometric processing methods. However, similar to graph-based representation, the acquisition of a geometric struc-

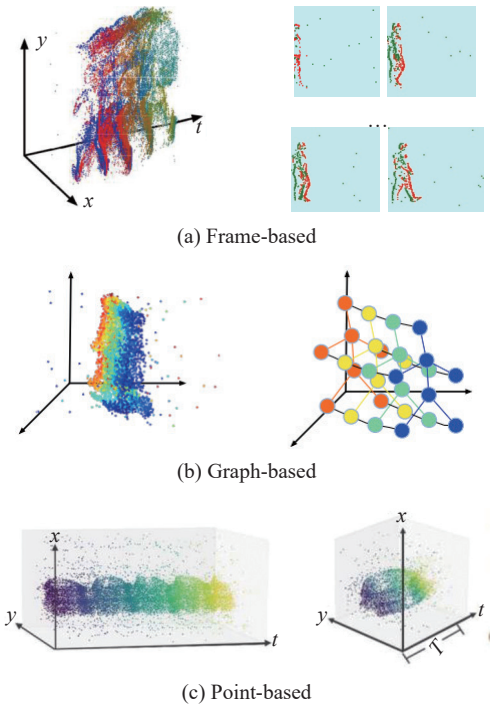


Fig. 14 Only parts of the events are selected with the help of attention.

ture sacrifices the real-time response of the network for event input.

4 On recognition: Network compression

DNNs are always redundant in most cases; thus the great potential ability of compression is inherent in this topic[162]. Moreover, since raw visual data usually have some internal high dimensions, corresponding DNNs become more redundant consequentially. In this section, four approaches of network compression in the field of visual recognition are introduced: 1) compact networks; 2) tensor decomposition; 3) data quantization; and 4) pruning. These approaches can make huge visual recognition networks efficient. Furthermore, some joint compression practices which synthesize multiple specific approaches are also presented. However, it should be clarified that network compression might still be an open issue in the field of brain-inspired models such as SNNs. Only a small number of practices have set foot in this direction[163, 164].

4.1 Compact networks

4.1.1 Compact CNNs

In fact, CNNs are born to deal with visual recognition, and the key natural feature of CNNs is weight sharing, which can be regarded as the earliest practice of compact networks to match the data structure of images. Thus, based on the characteristics of visual recognition tasks, further efforts have been proposed to make CNNs more compact to reduce the ever-growing network size[87, 165, 166]. In general, the most compact design for CNNs can be concluded from two perspectives, one of which is based on the receptive field of filters, and the other one is based on the topology within a single convolutional layer (intra-layer) or between convolutional layers (inter-layer). Besides, some more crazy ideas, which invent alternative building blocks for reducing the parameters to learn, appear to be another novel aspect.

Receptive field aspect. It is clear that designing an effective receptive field[167] is crucial to the representation capability of convolutions, which is jointly determined by the filter size and pattern. Regarding filter size, VGGNet[168] proposes a stack of two 3×3 convolutional layers to replace a 5×5 convolutional layer because their receptive fields are equivalent. In contrast, there are more practices in the aspect of filter pattern. For instance, an $n \times n$ convolutional layer can be split into two chained layers, which have an $n \times 1$ layer ahead, and a $1 \times n$ layer behind[84, 169], atrous or dilated convolution[85, 170, 171] uses irregular filters with holes, and deformable convolution[172] generalizes the atrous convolution to learn the offsets of sampling directly from the target tasks.

Topology aspect. The topology art of changing connections in CNNs can bring more ability of expression or lighter convolutional units. Since the well-known network in network (NIN)[173] was proposed, it has been known that vanilla convolutional kernels may be redundant, especially in the situation of multiple stacked convolutional kernels. Besides, Inception[174] and ResNet[175, 176] are proposed to enhance the performance of CNNs, which inspired many researchers to think about efficient convolutional units under the well-designed topology, e.g., SqueezeNet[86] uses amounts of 1×1 convolutions to replace 3×3 convolutions and reduce the counts of channels in the rest 3×3 convolutions. The residual unit has led to relatively more results in efficient structures, e.g., bottleneck architecture[84] and a similar one with depth-wise convolution[177] in ShuffleNet[87], MobileNetV2[178], and MobileFaceNet[179]. Fig. 15 illustrates several typical compact network designs in the aspect of topology.

New compact building blocks. The local binary convolutional neural networks (LBCNNs) introduced in [89] propose using an alternative to the traditional convolutional layer called the local binary convolution (LBC) layer inspired by the design of traditional local binary patterns (LBP), which uses a set of filters with sparse, binary and randomly generated weights that are fixed to replace the traditional convolutional filters whose weights need to be learned. A set (which can be just a small number) of learnable linear weights is used to integrate the feature maps after convolution. Later, a new network type called perturbative neural networks (PNNs)[88] replaced each convolutional layer with a so-called perturbation layer, which computes its response as a weighted linear combination of non-linearly activated additive noise

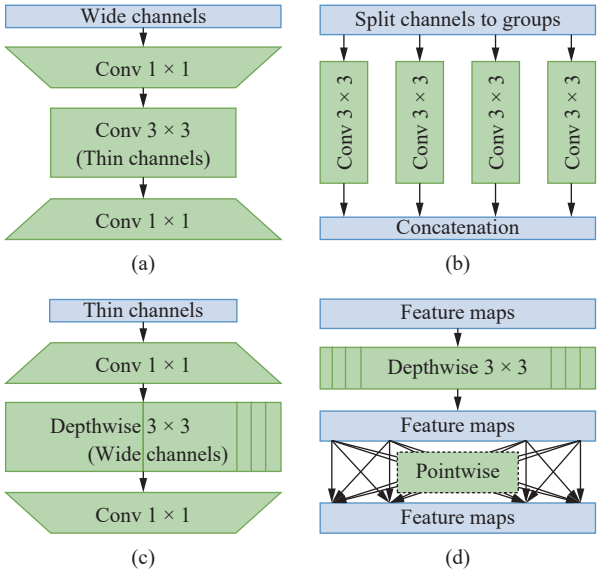


Fig. 15 Typical efficient inter-layer and intra-layer channel correlations: (a) Bottleneck architecture[84]; (b) Group convolution with 4 groups[180]; (c) Reversed bottleneck architecture with depthwise convolution[178]; (d) Depthwise-separable convolution[177].

perturbed inputs. Both the LBC and perturbation layer are shown to be a good approximation of the convolutional layer but with much fewer parameters to learn, and both LBCNNs and PNNs share the same idea of replacing weighted convolution with a linear weighted combination of feature maps.

4.1.2 Compact RNNs

RNNs are often used for video recognition to extract the temporal features, but designing a compact RNN is harder than CNN because the inner structures of gated units are complex, such as those of long short term memory (LSTM)[181, 182] and gated recurrent unit (GRU)[183]. That is to say, any single whole RNN contains several layers in which some units have elaborate internal structures. This situation is the most prominent characteristic, which does not exist in neurons of other kinds of DNNs. Therefore, compact RNNs can be designed at the level of units or at the level of whole networks. To discuss expediently and concisely, we give a simple equation that can abstract any gated connection, such as forget gate, input gate, output gate, etc.

$$\boldsymbol{y} = \sigma(\boldsymbol{W}\boldsymbol{x}(t) + \boldsymbol{U}\boldsymbol{h}(t-1) + \boldsymbol{b}) \tag{1}$$

where $\boldsymbol{y}$ is the data that passed the gate, $\boldsymbol{W}$ is the weight matrix corresponding to the input data for current time $\boldsymbol{x}(t)$, $\boldsymbol{U}$ is the weight matrix corresponding to the status of the previous time $\boldsymbol{h}(t-1)$, $\boldsymbol{b}$ is the bias vector, and the $\sigma(\cdot)$ is the activation function.

Units level. The topology of gated units in RNNs is complex; however, some approximate reconstruction may be considered to remove some subordinate connections. In fact, GRU is actually a compact design based on LSTM. In detail, there are four gated connections described by (1) in LSTM, whereas GRU has only three. Thus, the space and computation consumption of GRU can be less than that of LSTM. Other than GRU, S-LSTM[90] and JANET[91] contain forget gates only in their units. Minimal gated unit (MGU)[184] integrates the reset and update gates. Contrary to the elaborate gates, FastGRNN[185] uses a shared matrix that connects both input and state so that the number of gated connections is kept, but parameters are cut down. Finally, a quasi-recurrent neural network (QRNN)[186] replaces the previous output with previous input in the gate calculation, and accordingly, gated connections are transformed from (1) to

$$\boldsymbol{y} = \sigma(\boldsymbol{W}[\boldsymbol{x}(t), \boldsymbol{h}(t-1)] + \boldsymbol{b}). \tag{2}$$

Networks level. One can also simplify the architectures of stacked layers through multiple basic units, just like compact CNNs. Sak et al.[92] introduce a linear recurrent projection layer to reduce the dimensions of the output of the LSTM layers. Wu et al.[93] introduce residual connections into their 8-layer translation LSTM. Some other analogous compact RNNs include skip-connected RNN[187], Grid LSTM[188], and sequential recurrent neural network (SRNN)[189].

However, compact design is comparatively hard to implement and lacks uniform principles. Even worse, the compact method is not easy to achieve a high compression ratio, making it devoid of significance in dealing with large-scale visual recognition neural networks. Thus, this method is often used to combine with other compression methods or sometimes relies on expensive extra disposing, such as the transfer learning-based distillation[190].

4.1.3 Neural architecture search

Unlike the rigid human-designed or hand-crafted network architectures, neural architecture search (NAS) is a promising domain that can automatically construct a compact DNN. For conventional practice[94], NAS is the process of gradually training an RNN as the controller to generate a good architecture. In detail, this RNN should train repeatedly to gain the produced architecture and update the RNN controller itself based on evaluating the architecture by the environment or evaluator. Obviously, this process leads to massive training costs, since the search space will enlarge exponentially when the target DNN is deeper. More broadly, according to [17], there are three aspects of the challenges that lie in the field of NAS, i.e., search space, search algorithm and evaluation strategy, and many researchers focus their attention to solving these problems. It is interesting that even though the controller is usually an RNN, most of the target DNNs are oriented to CNNs, and only a few researchers are considered for RNNs[191–194]; thus, the review of NAS is mainly focused on CNNs here.

Search space. Search space contains various basic network elements, e.g., normal convolution, asymmetric convolution, depthwise convolution, any kind of pooling,

different activation functions, etc., and the laws to connect these elements together. Generally, by restricting the search space, the controller can search the basic network cells, consisting of some concrete elements to represent local structures, to construct the target DNN sequentially and efficiently[192, 195]. Moreover, the search complexity can be further reduced in the range of the cell, e.g., by progressively increasing the number of blocks within one layer, as shown in Fig. 16(a) and searching hierarchical topology within a cell as illustrated in Fig. 16(b).

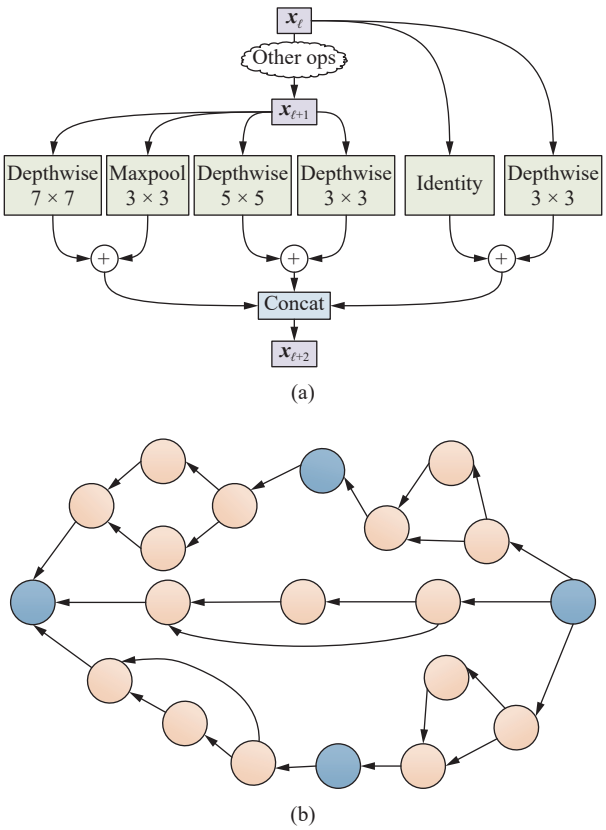


Fig. 16 Typical cells of NAS: (a) Three-block cell[196]; (b) Three-level hierarchical cell[197].

Search algorithm. Clearly, an appropriate search algorithm is critical since its target is to efficiently combine basic network elements together to be a whole network that fits the most specific task or dataset. Roughly speaking, except for naive random search, these algorithms can be classified into four approaches, i.e., 1) reinforcement learning (RL) including Q-learning[198, 199], REINFORCE[94, 200, 201], and proximal policy optimization (PPO)[192]; 2) neural evolutionary algorithm (EA)[95, 197, 202]; 3) Bayesian optimization (BO)[96, 203]; and 4) differentiable gradient for architecture (DGA)[97, 193]. It is hard to say which algorithm is better; however, the classical RL approach still reveals the higher boundary of performance[204]. Nowadays, some other minor improvements are also used to improve search efficiency, e.g., exploring the search space with the network transformation and weight reusing[200, 201, 205], and sequential model-based optimization (SMBO)[196], which can choose the most likely direction of optimization rather than blindly searching with a wide range.

Evaluation strategy. It is clear that the searched architecture should be evaluated to verify whether it can fit the task. The first strategy is simple, i.e., finitely train and evaluate the target DNN to observe its trends, e.g., just train on a subset of the dataset[206], build an accuracy predictor trained on the limited search space to guide the following search process[196, 207], predict the search direction based on partial training curves of current searched architectures[194, 208], etc. The second way is to use the searched architecture to inherit the weights that are produced in previous searched architectures[200, 201, 209]. The third aspect is called one-shot NAS, which only trains the one-shot model, from which the searched architectures can directly share the one-shot weights without extra training[97, 193].

To fully show its promising development tendency, we use Table 2 to compare some well-known human-designed networks with NAS models nowadays. On the whole, NAS models can surpass human designs in both accuracy and efficiency, especially EfficientNet[204] with the highest accuracy and several lightweight models for mobile format[193, 200, 204] with a tiny number of parameters and operations (GFLOPs). Additionally, the search time of NAS algorithms has also decreased significantly from 22 400[94] to 4[193] GPU days, making it possible to apply NAS models on real embedded devices. Nevertheless, there are still some problems to be solved in this direction, e.g., a lack of practical reports on deploying NAS models in fickle real data, enhancing the NAS models for mobile platforms to find the limitation of compact architecture, handling the trade-off between accuracy and size, etc. Hence, all signs indicate that this kind of approach is promising, but still needs further studies.

4.2 Tensor decomposition

Traditionally, since the parameters in DNNs are mostly stored as matrices, matrix decomposition is widely used in network compression, especially in singular value decomposition (SVD)[210–217]. However, the restriction of orders limits the use of matrix decomposition to compress larger and larger visual recognition neural networks. Besides, tensors may have some inherent connections with neural networks[218]. Hence, discussing tensor decomposition here is sufficient because a matrix is actually a second-order tensor.

4.2.1 Classical tucker decomposition

No matter canonical polyadic (CP) decomposition[219] or higher-order singular value decomposition (HOSVD)[220, 221], all classical tensor decomposition methods for a dth-order tensor $\mathcal{A} \in \mathbf{R}^{n_1 \times n_2 \times \cdots \times n_d}$ can be represented uniformly in Tucker decomposition[222] like [223]:

Table 2 Comparison between state-of-the-art human-designed networks (the upper part) and NAS models (the lower part) proposed in recent years based on ImageNet

References	Top-1 Acc (%)	Params (10^6)	Ops (10^9)	Algo
ResNet-152[175]	78.6	60.3	11.3	–
DenseNet-264[224]	79.2	32	15	–
ResNeXt-101[180]	80.9	44	7.8	–
PolyNet[225]	81.3	92	–	–
DPN-131[226]	81.5	79.5	16	–
Hierarchical[197]	79.7	64	–	EA
NAS-Net-A[192]	82.7	88.9	23.8	RL
PNASNet[196]	82.9	86.1	25	SMBO
AmoebaNet-A[95]	82.8	86.7	23.1	EA
TreeCell (mobile)[200]	74.6	–	0.59	RL
DARTS (mobile)[193]	73.3	4.7	0.57	DGA
Proxy (mobile)[97]	74.6	5.7	–	DGA RL
EfficientNet[204]	84	43	19	RL

$$\mathcal{A} = \mathcal{K} \times_1 \boldsymbol{F}^{(1)} \times_2 \boldsymbol{F}^{(2)} \times_3 \cdots \times_d \boldsymbol{F}^{(d)} \tag{3}$$

where $\mathcal{K} \in \mathbf{R}^{r_1 \times r_2 \times \cdots \times r_d}$ denotes the kernel tensor, any one $\boldsymbol{F}^{(i)} \in \mathbf{R}^{r_i \times n_i}$ $(i \in \{1, 2, \cdots, d\})$ is the factor matrix, and the operation $\times_i$ means the mode-i contracted product[223]. If every r_i equals a positive integer r_C and the kernel tensor $\mathcal{K}$ presents like a superdiagonal tensor, which means all elements in $\mathcal{K}$ are 0 except $\mathcal{K}(x_1, x_2, \cdots, x_d)$ with $x_1 = x_2 = \cdots = x_d$, then (3) will become CP (Canonical Polyadic) decomposition[223]. If all the factor matrices $\boldsymbol{F}^{(i)}$ are orthogonal and the kernel tensor $\mathcal{K}$ is so-called all-orthogonal[220, 221], then (3) will become HOSVD.

Many researchers have applied CP and Tucker to compress the weights in neural networks in recent years, especially CNNs for visual recognition[98, 99, 227–235]. In general, to utilize (3) efficiently, any single weight matrix $\boldsymbol{W} \in \mathbf{R}^{M \times N}$ should be mapped into a dth-order tensor $\mathcal{W} \in \mathbf{R}^{m_1 n_1 \times m_2 n_2 \times \cdots \times m_d n_d}$ where $M = \prod_{i=1}^{d} m_i$ and $N = \prod_{i=1}^{d} n_i$[101]. Thus, the greater value of d can bring about more effective compression, which decreases the complexity from $\mathcal{O}((mn)^d)$ to $\mathcal{O}(dmnr + r^d)$. However, the curse of dimensionality[236, 237] has not been solved completely by Tucker because of $\mathcal{O}(r^d)$. In addition, the relatively new block term decomposition (BTD)[238], which is the sum of multiple Tucker blocks, can ease this curse to some extent, since the size of each kernel tensor may be smaller. In [100], BTD-LSTM shows a superior ability to keep information for visual recognition, but the corresponding computation process is relatively complex. On the other hand, the most recent Kronecker canonical polyadic (KCP) decomposition[239], which combines the characteristic of Kronecker products and the sparsity of the kernel tensor of CP[240], can implement very fast calculation and a considerable compression ratio[163]. It is worth mentioning that this work[163] further extends tensor decomposition to the brain-inspired SNNs by considering SNN as a variant of RNN[241, 242].

4.2.2 Tensor network

The tensor network [243, 244], which represents a tensor with a link of matrices or low order tensors with contracted products, is promising to avoid the curse of dimensionality by eliminating the high order kernel tensor with r^d elements according to (3). Commonly, there are three types of tensor networks applied, i.e., tensor train (TT)[245, 246], tensor chain (TC)[247, 248] or tensor ring (TR)[249, 250], and hierarchical Tucker (HT)[251, 252].

Tensor train. According to[253], a dth-order tensor $\mathcal{A} \in \mathbf{R}^{n_1 \times n_2 \times \cdots \times n_d}$ can be represented as

$$\mathcal{A} = \mathcal{G}_1 \times^1 \mathcal{G}_2 \times^1 \cdots \times^1 \mathcal{G}_d \tag{4}$$

where the operation $\times^1$ is the mode-$(N, 1)$ contracted product[254], and the core tensors $\mathcal{G}_i \in \mathbf{R}^{r_{i-1} \times n_i \times r_i}$ $(i = 1, 2, \cdots, d)$ always have $r_0 = r_d = 1$. Apparently, according to the equation above, the spatial complexity of the TT format is $\mathcal{O}(dnr^2)$, where n is the maximum value of the modes and r is the maximum value of the TT ranks. It is obvious that the $\mathcal{O}(r^d)$ in classical decomposition, i.e., (3), is avoided, and the curse of dimensionality is solved.

Tensor chain. The TC format, which can be regarded as a variant of the TT format, was first proposed by Khoromskij[247] and proved to have a similar approximation capability as TT by Espig et al.[248] Nowadays, the TC format has been introduced in the field of DNNs as the TR format by Zhao et al.[249, 250] Compared to TT, the TC format of a dth-order tensor $\mathcal{A} \in \mathbf{R}^{n_1 \times n_2 \times \cdots \times n_d}$ has only one difference, which is $r_0 = r_d \neq 1$. Thus, in contrast to (4), there are two pairs of equal modes that must be contracted at the end, like

$$\mathcal{A} = (\mathcal{G}_1 \times^1 \mathcal{G}_2 \times^1 \cdots \times^1 \mathcal{G}_{d-1}) \times_{1,d+1}^{3,1} \mathcal{G}_d \tag{5}$$

where $(\mathcal{G}_1 \times^1 \mathcal{G}_2 \times^1 \cdots \times^1 \mathcal{G}_{d-1}) \in \mathbf{R}^{r_d \times n_1 \times n_2 \times \cdots \times n_{d-1} \times r_{d-1}}$, $\mathcal{G}_d \in \mathbf{R}^{r_{d-1} \times n_d \times r_d}$, and $\times_{1,d+1}^{3,1}$ means the paired contracted modes are, the 1st of the former versus the 3rd of the latter while the $(d+1)$th of the former versus the 1st of the latter. Fig. 17 shows TT and TC in tensor network graphs, where every node represents a tensor, and each edge is a mode of its connected tensor.

Hierarchical tucker. In fact, HT is the fountainhead of TT, i.e., TT is a special form of HT[255], since HT has an extremely flexible organizational structure. Particularly, for a dth-order tensor $\mathcal{A} \in \mathbf{R}^{n_1 \times n_2 \times \cdots \times n_d}$, their modes could be divided into two sets as $t = \{t_1, t_2, \cdots, t_k\}$ and $s = \{s_1, s_2, \cdots, s_{d-k}\}$, then we have

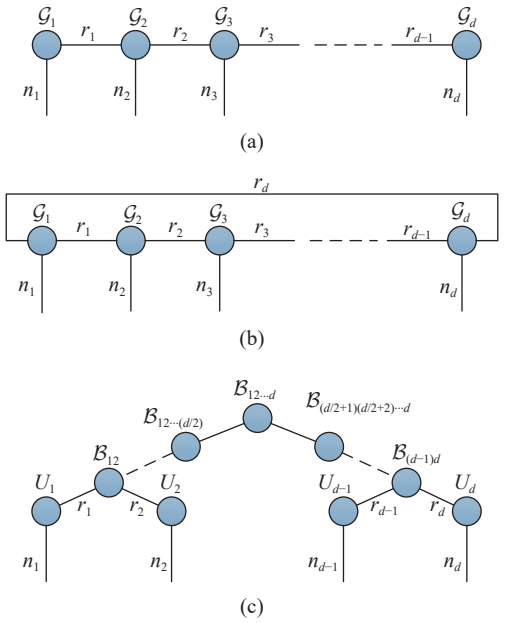


Fig. 17 Tensor network graphs for (a) tensor train, (b) tensor chain and (c) hierarchical tucker format of a dth-order tensor $\mathcal{A} \in \mathbf{R}^{n_1 \times n_2 \times \cdots \times n_d}$.

$$\boldsymbol{U}_t = (\boldsymbol{U}_{t_l} \otimes \boldsymbol{U}_{t_v})\boldsymbol{B}_t \tag{6}$$

where $\boldsymbol{U}_t \in \mathbf{R}^{n_{t_1} n_{t_2} \cdots n_{t_k} \times r_t}$, $\boldsymbol{U}_{t_l} \in \mathbf{R}^{n_{t_{l_1}} n_{t_{l_2}} \cdots n_{t_{l_i}} \times r_{t_l}}$, $\boldsymbol{U}_{t_v} \in \mathbf{R}^{n_{t_{v_1}} n_{t_{v_2}} \cdots n_{t_{v_{k-i}}} \times r_{t_v}}$ are called truncated matrices, $\boldsymbol{B}_t \in \mathbf{R}^{r_{t_l} r_{t_v} \times r_t}$ is termed as a transfer matrix, and $\otimes$ is the Kronecker product. One can continuously use (6) until all the truncated matrices become $\boldsymbol{U}_i \in \mathbf{R}^{n_i \times r_i}$. Thus, the HT format of $\mathcal{A}$ will be done. Obviously, there are multiple ways to split the modes of $\mathcal{A}$. However, even the simplest format with the binary tree is still formidable to write in the formulation[107, 108]. Fortunately, the tensor network graph in Fig. 17(c) can provide a convenient description.

4.2.3 Typical practices

In general, the TT format represents the most vibrant tensor decomposition method. We list the current applications of TT compressed neural networks, including both CNNs and RNNs for visual recognition, in Table 3, where a handful of TC and HT are also introduced. We find an interesting phenomenon that the tensor-decomposed CNNs for image tasks are hard to avoid the accuracy loss, while RNNs for video tasks are easy to achieve higher accuracy in compressed forms. We guess that RNNs have a larger scale than CNNs in general, which verifies that a larger network is easier to compress[256]. The practice in 3DCNNs further verified this, as 3D convolutional kernels have heavier redundancy[103]. However, how to deal with the accuracy loss still needs to be learned.

4.3 Data quantization

Data quantization can project concrete data, including weight matrices, gradients, nonlinear activation functions, etc., from high-precision value space to low-precision value space, e.g., from real number field $\mathbf{R}$ to integer field $\mathbf{N}$. This approach can not only reduce the space complexity of network parameters, but also accelerate the running time of neural networks because bit operations are much faster than float operations. As raw visual data is generally represented by integers, quantized neural networks may be more suitable for visual recognition tasks.

4.3.1 Method

Problem formulation. In general, there are two for-

Table 3 Applications of neural networks compressed by tensor networks for visual recognition tasks

References	Format	Compressed parts	Dataset	Compression ratio	Accuracy loss
Novikov et al.[101]	TT CNN	FC	CIFAR10	11.9×	1.26%
Zhao et al.[249]	TC CNN	FC	CIFAR10	444×/1 300×	0.13%/2.18%
Huang et al.[257]	TT CNN	FC	MINST	14.85×	1.5%
Su et al.[258]	TT CNN	FC	MINST	500×	2%
Garipov et al.[102]	TT CNN	Conv & FC	CIFAR10	82.87×	1.1%
Wang et al.[103]	TT 3DCNN	Conv & FC	UCF11/ModelNet40	107.5×/160.7×	0.22%/0.14%
Wang et al.[259]	Nonlinear TT CNN	Conv & FC	CIFAR10/ImageNet	13.03×/7.65×	1.57%/1.68%
Tjandra et al.[104]	TT GRU	$\boldsymbol{W}$ & $\boldsymbol{U}$	Sequential MNIST	43.52×/69.80×	0.3%/0.3%
Yang et al.[105]	TT LSTM	$\boldsymbol{W}$	UCF11/Hollywood2	17 554.3×/23 158.8×	–30.4%/–43.8%
	TT GRU			13 687.1×/18 313×	–32.5%/–28.8%
Pan et al.[106]	TC LSTM	$\boldsymbol{W}$	UCF11/HMDB51	34 192×	–1.5%/–0.9%
Wu et al.[107]	HT LSTM	$\boldsymbol{W}$ & $\boldsymbol{U}$	UCF11/UCF50	58.41×/57.96×	0.12%/1.37%
Yin et al.[108]	HT LSTM	$\boldsymbol{W}$	UCF11/Youtube Face	47 375×/72 818×	–17.5%/–54.9%
Wang et al.[163]	KCP LSTM	$\boldsymbol{W}$ & $\boldsymbol{U}$	UCF11/UCF50	59 338×/278 219×	–13.1%/–19.8%

mulations to describe how to transform floating-point weights, gradients, activations, etc., to quantized data types. One category, which is the most widely used, projects the original high-precision value to the space of quantized data as

$$Q(x) = \Delta \cdot round\left(\frac{x}{\Delta}\right) \tag{7}$$

where x is the original high-precision value in the continuous space, $Q(x)$ is the quantized data in a discrete space, $round(\cdot)$ is the rounding operation, and Δ is the quantization step length if the discrete states have a uniform distribution. If K-bit quantization is used, we have $\Delta = 1/(2^{K-1})$ to discretize $x \in [0, 1]$ to 2^K states. This category, as described in (7), is straightforward and easy to consider. Therefore, most practices have followed this direction, which can be observed in Table 4.

The other category regards quantization as an optimization problem and tries to solve it approximately. The classic model can be generally governed by

$$\min_{Q} \|\boldsymbol{X} - Q(\boldsymbol{X})\|_2^2, \quad \text{s.t. } Q_i \in X_Q \text{ for all } i \tag{8}$$

where $X_Q = \{Q_1, Q_2, \cdots, Q_n\} (i \in \{1, 2, \cdots, n\})$ is a set that has n discrete states for quantization. The earliest and the most well-known quantization in the category of optimization is XNOR-NET[111]. An obvious motivation is that the optimization problem pays more attention to the whole network rather than local quantization, so (8) may be more appropriate for large-scale neural networks.

Quantized objects. As mentioned above, except for weight (W), there are also several other different objects in neural networks that can be quantized, such as activation (A), error (E), gradient (G), and weight update (U). Fig. 18 illustrates the data of these quantized objects W_Q, A_Q, G_Q, E_Q, and U_Q existing in forward pass, backward pass, and weight update processes. The parameter (W) is the most straightforward to be dealt with. The propagation data (A, E) correlate strongly with the data flow during forward and backward passes, which greatly influence the accelerating. Quantized gradient and update greatly help train the full quantized networks, but are harder to implement. Thus, the corresponding practices are fewer, as shown in Table 4.

Algorithm description. Generally, if the bit-width K is given, (7) can be rewritten as

$$x_Q = Q(x, K) \tag{9}$$

where x_Q is the quantized data with K-bits, i.e., quantized objects W_Q, A_Q, G_Q, E_Q and U_Q described

Table 4 Typical practices of quantization for CNNs

References	Formulation	Objects	State distribution	State projection	Performance
VQN[260]	Optimization	W	Uniform	Deterministic	CIFAR10, DenseNet, 91.22%
ADMM[261]	Optimization	W	Non-uniform	Deterministic	ImageNet, ResNet50, 72.5%
INQ[262]	Projection	W	Non-uniform	Deterministic	ImageNet, ResNet18, 66.02%
Joint training[263]	Optimization	WA	Uniform	Deterministic	ImageNet, ResNet34, 73.7%
Balanced DoReFa[264]	Projection	WA	Uniform	Deterministic	ImageNet, ResNet18, 59.4%
Regularization[265]	Projection	WA	Uniform	Deterministic	ImageNet, ResNet18, 61.7%
HAQ[266]	Projection	WA	Uniform	Deterministic	ImageNet, ResNet50, 76.14%
GXNOR-Net[267]	Projection	WA, U	Uniform	Stochastic (W) Deterministic (A)	CIFAR10, VGG8, 92.5%
QBPv2[268]	Projection	WA, E	Uniform	Deterministic	CIFAR10, ResNet18, 89.2%
WAGE[109]	Projection	WA, G, E, U	Uniform	Deterministic (WA, E, U) Stochastic (G)	ImageNet, AlexNet, 48.4%
FX training[269]	Projection	WA, G, E, U	Uniform	Deterministic	CIFAR10, ResNet20, 92.76%
8b training[270]	Projection	WA, G, E, U	Uniform	Deterministic Stochastic	ImageNet, ResNet50, 71.72%
OCS[271]	Projection	WA	Non-uniform	Stochastic	ImageNet, ResNet50, 75.7%
Full 8-bit[110]	Projection	WA, G, E, U	Uniform	Deterministic	ImageNet, ResNet50, 69.07%
AutoQ[112]	Optimization	WA	Non-uniform	Stochastic	ImageNet, ResNet50, 74.47%
1b ReActNet[272]	Projection	WA	Non-uniform	Deterministic	ImageNet, MobileNet, 71.4%
HybridQ[273]	Optimization	WA	Non-uniform	Stochastic	ImageNet, ResNet50, 77.74%
VecQ[274]	Optimization	WA	Non-uniform	Stochastic	ImageNet, MobileNetV2, 72.24%

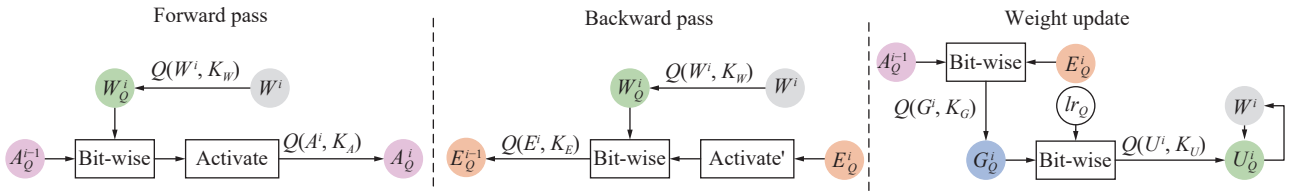


Fig. 18 Data of quantized objects including W_Q, A_Q, G_Q, E_Q and U_Q

above. Data flow among these quantized data can generally be computed by a bit-wise operation which is much faster than the full precision computation. Besides, the performance of quantized networks with appropriate bit-width will not degenerate which can be learned in Table 4. The overall algorithm is briefly described in Fig. 18, and a more comprehensive algorithm description of quantization can be consulted in [110].

State distribution and projection. The data in the quantization set always have concrete, discrete states, which may contain different distributions. For example, uniform distribution is the most widely used one, logarithmic distribution has exponential variance on step length that has obvious benefits to convert multiplication to addition, and adaptive distribution often occurs in the situation when formulating the quantization as an optimization problem such as TTQ[275], ADMM[261], etc. The appropriate distribution will help to achieve considerable precision, even the 1-bit quantization[272]. On the other hand, the primary mission in quantization is projecting the original high-precision data to the discrete state space, of which deterministic and stochastic projections are the two main approaches used widely. The former projects the high-precision data to the nearest discrete state, while the latter projects the data to one of the two adjacent states with probability, which is determined by the distance from the original data to the discrete states. Generally, deterministic projection is easier to handle, so most references listed in Table 4 have selected it.

4.3.2 Typical practices

In the aspect of CNNs, we list a number of typical and latest practices with their best performance on CIFAR10 or ImageNet in Table 4. Note that the compression ratio of quantization is not considered because it relates directly to the bit-width; thus, the storage saving is limited. As can be clearly observed, all the recent practices quantize W, and most works quantize both W and A, while a small number of works quantize G, E, or U. It is worth mentioning that the practice that quantizes W, A, G, and E[109] inspires some practices on semantic segmentation tasks in terms of accelerating corresponding encoder-decoder CNNs[276, 277]. We emphasize quantized objects here rather than other aspects of quantization, e.g., state distribution and state projection, because the choice of which objects are quantized can influence the training and inference processes significantly. Particularly, quantized G and U can simplify training greatly because the back-propagation data flow can be handled as integers. Moreover, Banner et al.[268] propose the range batch normalization (BN) to greatly reduce the numerical instability and arithmetic overflow caused by the popular standard deviation-based BN. Hence, the BN can also be quantized. We optimistically believe that some entirely quantized DNNs will come soon, and then mature and efficient integer neural networks should become the mainstream, especially for embedded surroundings.

The number of practices of quantized RNNs for visual tasks is few, to the best of our knowledge, compared to natural language processing tasks[111, 264, 278–280]. Unlike CNNs, RNNs are dynamic systems that reuse the weight and accumulate the activation error in the temporal dimension. Furthermore, there are two dimensions of back-propagation in RNNs: spatial (layer-by-layer) and temporal (step-by-step). These situations make it harder to clarify the training data flow and quantization sensitivity. In fact, many quantization methods can be shared by both CNNs and RNNs[264, 281]. However, more applications of quantized RNNs for visual tasks should be established considering the complexity of RNNs discussed above.

4.4 Pruning

Pruning can reduce the number of weights or neurons. Thus, the memory and calculation costs are retrenched. However, the additional indices for indicating the location of non-zero elements, and the irregular access or execution pattern, become the two major drawbacks.

4.4.1 Basic method

Problem formulation. Similar to quantization, there are also two formulations that can describe pruning. The first one is direct and naive, i.e., using some search algorithms for trained DNNs to find those "unimportant" weights or neurons to prune, like

$$S(\boldsymbol{X}) = sparse(\boldsymbol{X}) \tag{10}$$

where $\boldsymbol{X}$ is the weight matrix, and $S(\boldsymbol{X})$ is the new weight matrix after pruning. The function $sparse(\cdot)$ is the search algorithm to pre-select the unimportant weights and neurons to be pruned. Such search approaches could be low-precision estimation[113, 114], negative activation prediction[115], etc. Besides, hashing trick[282] may also help to make weights in DNNs sparse[283], even if it seems to be weight sharing rather than a pruning approach in concept. Furthermore, some other hashing methods may

reveal similar abstract thought to normal pruning, e.g., using locality-sensitive hashing (LSH) to collect neural nodes in an active set, and other neural nodes not in this set will not be computed during forward and backward calculations[284].

However, it is obvious that the search algorithm may consume vast computing time for large-scale DNNs. Furthermore, as the same reason for quantization, the problem of pruning as optimization may be a more adaptive formulation for large DNNs because of the layer coupling. A general formulation of pruning in optimization can be described as[116]

$$\min_{W} L_0(W) + \lambda \sum_{g=1}^{G} \|\boldsymbol{W}^{(g)}\|_2 \tag{11}$$

where $W = \{\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)}, \cdots, \boldsymbol{W}^{(G)}\}$ is the set of all weights in G different layers or parts, λ is a penalty parameter that affects the sparsity, and $L_0(W)$ is the normal loss function of DNNs.

Pruning objects. There are two typical pruning objects: weight pruning and neuron pruning, which are illustrated in Fig. 19. The former reduces the number of edges that can make weight matrices sparser, while the latter reduces the number of nodes to make weight matrices smaller. Evidently, the latter method may cause more accuracy loss than pruning weights alone. Using ResNet50 as an example, according to Table 5, weight pruning networks[285–288] can exceed neuron pruning networks[117, 289, 290] in terms of the performance of top-1 accuracy on ImageNet in most cases. Nevertheless, this gap has been reducing recently.

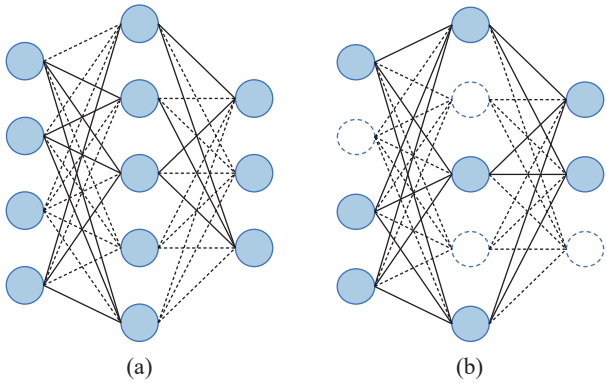


Fig. 19 Pruning objects: (a) Weight pruning; (b) Neuron pruning.

Pruning structure. Generally, the compute acceleration has a great deal to do with the sparse pattern, which is referred to as pruning structure in this survey. It

Table 5 Typical practices of pruning for CNNs

References	Formulation	Object	Structure	Compression ratio	Performance	Accuracy loss
Prune or Not[256]	Search	W	Element	8×	ImageNet, InceptionV3, 74.6%	3.5%
Nest[291]	Search	W & N	Element & Block	15.7×	ImageNet, AlexNet, 57.24%	0.02%
				13.9×	ImageNet, ResNet50, 71.94%	0.35%
DGC[285]	Search	W	Element	597×	ImageNet, AlexNet, 58.2%	– 0.01%
				277×	ImageNet, ResNet50, 76.15%	– 0.06%
ThiNet[117]	Optimization	N	Block	16.64×	ImageNet, VGG16, 67.34%	1%
				2.06×	ImageNet, ResNet50, 71.01%	1.87%
Slimming[292]	Optimization	N	Block	2.87×	CIFAR10, DenseNet40, 94.35%	– 0.46%
				1.54×	CIFAR10, ResNet164, 94.73%	– 0.15%
ISTA[289]	Optimization	N	Block	1.89×	ImageNet, ResNet101, 75.27%	1.13%
AutoPrunner[290]	Optimization	N	Block	3.33×	ImageNet, ResNet50, 73.05%	3.1%
LCP[286]	Search	W	Vector	–	ImageNet, ResNet50, 75.28%	0.85%
AMC[287]	Search	W & N	Element & Block	5×	ImageNet, ResNet50, 76.11%	0.02%
Hybrid prune[288]	Search	W	Element & Vector	3.69×	ImageNet, ResNet50, 74.32%	1.69%
Joint sparsity[122]	Optimization	W	Vector	2.8×	ImageNet, ResNet18, 67.8%	0.4%
Importance[293]	Search	W & N	Element & Block	3.29×	ImageNet, ResNet101, 74.16%	3.21%
SSR[118]	Optimization	N	Block	2.13×	ImageNet, ResNet50, 71.47%	3.65%
Rewinding[294]	Search	W & N	Element & Block	5.96×	ImageNet, ResNet50, 76.17%	0%
SEP[295]	Optimization	W & N	Block	1.75×	ImageNet, ResNet50, 75.22%	0.9%
FSP[296]	Optimization	W & N	Block	1.55×	ImageNet, ResNet50, 75.22%	0.91%
EPruner[297]	Optimization	W & N	Block	2.01×	ImageNet, ResNet50, 74.26%	1.75%

is well-known that the operation in one neural layer can be abstracted as matrix multiplication. Thus, the pruning structure can be described as the number of zeros in the matrix. Besides, the convolutional computation is commonly converted to the modality of GEneral matrix multiplication (GEMM) by lowering the features and weight tensors to matrices[298].

Fig. 20 illustrates different pruning structures: element-wise, vector-wise, and block-wise. Note that different pruning grains produce different pruning structures. For instance, in weight pruning, kernel (discrete), fiber, or filter pruning produces vector sparsity, while channel or kernel (group) pruning produces block sparsity.

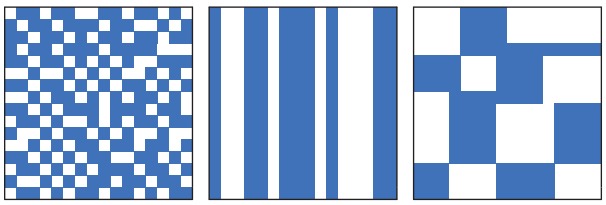


Fig. 20 Pruning structure: (a) Element-wise; (b) Vector-wise; (c) Block-wise.

4.4.2 Typical practices

Here we let W and N denote weight pruning and neuron pruning, respectively. In the aspect of CNNs, according to Table 5, the works that prune W and those that prune N are close in terms of quantity, and just a few practices have considered both of them (W and N)[291]. We emphasize pruning objects here because as mentioned before, pruning only weight or neuron may influence the performance more obviously than other aspects, i.e., formulation and structure. From Table 5, in terms of performance, it cannot be absolutely deemed that optimization is better than searching for large DNNs so far. One certain advantage of optimization is efficient training and avoiding some time-consuming searching procedures. For pruning, element structure can be helpful for achieving a higher compression ratio, especially in the case of DGC[285]. Relatively, other structures may bring a significant acceleration of computing, particularly filter (vector) and channel (block) pruning in convolutional kernels. On the whole, pruning on convolutional kernels is much more difficult than that on fully connected layers.

In the aspect of RNNs, there are two typical image captioning tasks, one of them prunes W[299] and the other prunes N[300]. The former has achieved a 13.12× compression ratio on the MS COCO dataset with a 2.3 improvement in the CIDEr score. The latter has also used the MS COCO dataset and got only a 0.4 reduction in the BLUE-4 score with 50% sparsity. Comparatively speaking, it might imply that there is still a lot of potential for visual RNNs with pruning to advance further.

4.5 Discussions and the derived joint compression

In fact, each compression method has its own characteristics, which other compression methods do not have. Therefore, before listing recent practices of joint compression, we present here our observations and thoughts about what features make each compression method worthy and unique based on the foregoing content in this section.

4.5.1 Main characteristics of each method

Compact networks. As discussed before, there are two levels of compact design, one is a delicate cell, and the other is an NAS algorithm. According to Table 2, NAS shows superior performance compared to human-designed networks in both accuracy and storage saving. The most critical point is that the execution time of the NAS algorithm has been shortened a lot nowadays[193]. Such a situation makes NAS promising and generic for embedded applications with various or changing surroundings, though there is still a lot of detailed work to do towards real applications. While other compression methods, i.e., decomposition, quantization, and pruning, still lack enough flexibility because a single specific network architecture is hard to apply to all kinds of datasets.

Tensor decomposition. In early studies of network compression, tensor decomposition was the only approach that supported the so-called in situ training[301], which means training a new model from scratch. Meanwhile, quantization and pruning needed pre-training in most cases to discover the distribution of weights to quantize or prune further. However, new studies of quantization[263] and pruning[302, 303] have made up this short slab. Moreover, these reports conclude that in situ training could perform better than fine-tuned models. Even so, tensor decomposition still appears to be the most powerful in the aspect of compression ratio, particularly for RNNs, according to Table 3. Currently, the unique characteristic of tensor decomposition is that various tensor network formats may have some inner links to DNN architectures. Thus, a theoretical explanation of efficient DNNs may be explored in this kind of compression. Chien and Bao[99] regard the whole Tucker decomposition process in (3) as a neural network connection. Su et al.[258, 304] use the HT decomposition to explain the depth efficiency of DNNs. Chen et al.[305] find that BTD can describe various bottleneck architectures in ResNet and ResNeXt. Li and Wang[306] propose a new DNN architecture based on the MERA tensor network, which is a kind of renormalization group (RG) transformation[307]. Wang et al.[259] map the TT structure into a sequenced multi-layer architecture. In this sense, it is expected that the studies of compressing DNNs with tensor decomposition can build a bridge between the experiments and the system of theories about DNNs.

Data quantization. Although data quantization is

not very good at reducing model complexity (in terms of the number of parameters) of DNNs compared with decomposition and pruning, it still has a significant advantage of computing acceleration and friendly deployment of embedded hardware, which are guaranteed by the low bit data flow in quantized DNNs. The authors implemented WAGE[109] on an FPGA platform and further found that 8-bit models perform 3× faster in speed, 10× lower in power consumption, and 9× smaller in circuit area than 32 float point models[110]. Although pruning with appropriate structure may also help to reduce computation complexity, bit operations in quantized DNN are still more adaptive for hardware environments.

Pruning. According to Tables 3 and 5, pruning has a better capability of accuracy maintenance or even improvement, though seemingly other methods like tensor decomposition have more potential power to gain a higher compression ratio. However, the loss of accuracy of compressed DNNs under decomposition is hard to avoid, especially for CNNs at present. For example, the performance of VGG16 on CIFAR10 with pruning can achieve an increase in accuracy of almost 0.15% in the new study[302], while the accuracy loss is hard to compensate for, even though the ranks are set very high based on TT convolutions[103]. Additionally, two main advantages of decomposition, i.e., higher compression ratio and in situ training, can be separately obtained by [285] and [302] in the aspect of pruning. Furthermore, the so-called lottery ticket hypothesis (LTH)[303], which is also proposed on the basis of pruning and claims that any DNN must have its compressed one without degradation, appears to be the most promising theoretical explanation for neural network compression[308–311], and can also be extended to tensor decomposition[163] and even dynamic neural networks[52]. Like NAS, the data structure of pruned DNNs may present complexity and chaos, unfriendly to embedded applications, and corresponding theory explanation.

4.5.2 Joint compression

Considering different characteristics of each compression approach, recently, some researchers have made their applications involve more than one class of methods besides using individual compression method, termed joint compression in this survey. Corresponding visual recognition works are illustrated in Table 6. The joint compression has great potential for a higher compression ratio. For instance, compression ratio of 89 × of AlexNet on ImageNet (58.69% accuracy)[262], 28.7 × of ResNet18 on ImageNet (66.6% accuracy)[122], and 1910 × of LeNet5 on Mnist (98% accuracy)[312].

However, researchers should pay more attention to maintaining the model accuracy in joint compression through sufficient analyses and comprehensions of the respective characteristics of each compression component. For example, since there might be a projection between the structure of tensor decomposition and the architecture of DNN as discussed above, the decomposition method appears to be perpendicular to the other methods to some extent. Hence, combining it with quantization or pruning is a promising direction that still needs further research, according to Table 6.

Table 6 Existing works of joint compression for visual recognition tasks

References	Compact	Decompose	Quantize	Prune
Deep compression[121]	–	–	✓	✓
SCNN[119]	–	✓	–	✓
Force regularization[120]	–	✓		✓
INQ[262]	–	–	✓	✓
Quantized distillation[313]	✓	–	✓	–
VNQ[260]	–	–	✓	✓
Joint sparsity[122]	–	–	✓	✓
Regularization[265]	–	–	✓	✓
ADMM[312]	–	–	✓	✓
PQASGD[314]	–	–	✓	✓
DNNC[315]	–	–	✓	✓
QTTNet[316]	–	✓	✓	–

5 On recognition: Efficient inference

Efficient training can allow learning from more data, with more parameter tuning or a complete architecture search, which can all lead to a better trained model. However, it is also critical to make the model run efficiently on affordable or existing devices and easily transfer to new data/tasks. In many cases, training may be done just once or only in one place (e.g., in the cloud) with powerful computational resources. However, run-time inference has to be done on cost-sensitive and thus resource-limited edge computing devices. Therefore, in some sense, the efficient inference is a more serious issue when real applications are concerned. In this section, we focus on fast run-time inference for model deployment at the testing stage and dynamic inference efforts for making the models efficient.

5.1 Fast run-time inference

Efficiency is not only a big concern for training deep learning models, but also rather important and sometimes critical for deployment at the users′ end in real applications. Therefore, the acceleration of run-time inference with limited resources is an indispensable issue of great importance for industrial applications. This is even more critical for applications that require real-time or even faster responses, such as automatic recognition for autonomous driving. There has already been rich literature on accelerating run-time inference with DNNs. We roughly categorize them into two groups based on the difference of focus: data-aware acceleration and network-

centric compression, and detail their recent advances and trends as follows.

5.1.1 Data-aware acceleration

During run-time inference, in many cases, there is no need to check all the input data for generating the final recognition results, especially for the data which may include a lot of irrelevant or redundant information, such as videos. Therefore, efforts to reduce or avoid the computation of the irrelevant/redundant parts of the data are very important for fast inference.

A simple yet very helpful direction is to explore the similarity of the intermediate feature maps of two consecutive video frames to reduce redundant computation. A representative earlier work is the one called deep feature flow[47]. It runs full expensive convolutions only on sparse keyframes, and then propagates their deep feature maps to other frames via a flow field. A significant speedup was achieved as the flow computation is relatively faster than full convolution. This work was extended to a more unified framework[317], which proved to be faster, more accurate, and more flexible. The framework contains three main components: sparsely recursive feature aggregation for ensuring both efficiency and feature quality, spatially-adaptive partial feature updating for improving the quality of features from non-keyframes, and temporally-adaptive keyframe scheduling for more efficient and better quality keyframe usage. However, these two works are still computationally expensive, as they rely on per-pixel flow computation, which is a heavy task.

Recently, Pan et al.[123] have explored another direction. They proposed a novel recurrent residual module (RRM) that only does dense convolution on the first frame and has the following frames fed into a sparse convolution module that only extracts information from the different images of neighboring frames. The sparse convolution has no bias term, and it shares the same filter banks and weights as dense convolution. After enhancing the sparsity of the data (by different images), a general and powerful inference model called EIE (efficient inference engine)[124], is adopted to make the inference efficiently according to the dynamic sparsity of the input. In usage, the video is split into several chunks, which can be processed with RRM-equipped CNN in parallel. Good results (speedup) have been observed in object detection and pose estimation in videos, and the model shall also be applicable to other visual recognition tasks for videos. Moreover, since it only explores the natural sparsity of data, it can ensure that there is no loss of accuracy during the speedup.

Besides information redundancy in consecutive frames, irrelevant information may largely exist for object detection in videos, as objects often occupy only a small fraction of each video frame. Therefore, an intuitive acceleration strategy is to do dense, full processing on only a few frames and make use of the spatio-temporal correlation among nearby frames to save computation on the other frames. A representative work reallocates the computation over a scale-time space called scale-time lattice[125]. It performs expensive detection sparsely on keyframes and propagates the results across both scales and time with substantially cheaper networks by making use of the strong corrections among object scales and time. Together with some other minor novel components (e.g., a network for temporal propagation and an adaptive scheme for keyframe selection), the work achieved a better speed-accuracy trade-off than previous works. Another representative work called spatiotemporal sampling network (STSN) focuses on feature-level propagation across adjacent frames. STSN is mainly about deformable convolutions over time, which is optimized directly with respect to video object detection performance. The approach has natural robustness to occlusion and motion blur, which are two key challenges in detecting objects in videos.

5.1.2 Network-centric compression

The main contents of the last section focus more on theoretical methodology and training. Meanwhile, for practical scenarios, a lot of effort has been made on general network-centric compression for fast inference. A comprehensive survey may have to cover several dozens of publications. Due to the scope and page limits of this survey, only a few representative ones for showing the recent advances and trends are covered here, as briefly introduced and compared in Table 7.

Besides pointing out their key ideas, main approaches, advantages, and disadvantages, we provide a group of key property indicators (KPIs) for briefly evaluating them from the perspective of important factors for real applications. There are five indicators, with the following detailed explanations.

"General or specific?" Whether the model is a general one that can be applied to the compression of any network or it is specific to some data types, tasks, or network structures.

"Static or dynamic?" "Static" means that the compression has to be done together with the training of the original models, and any changes in the compressed model have to be accompanied by retraining of the original model so that once a compression model is optimized, it is static; "dynamic" means that the compression model can be changed dynamically on-demand without requiring retraining of the original model. In the last two years, there has been a clear trend toward shifting to dynamic compression, so most of the references in Table 7 are about dynamic models, and only one representative of the static model is listed.

"Easy user control?" Whether the compression can be easily controlled by the users or not, such as using one or very few hyper-parameters for an intuitive controlling of the compression/speedup rate.

"Input adaptive?" Whether the compression is made adaptive to the input data or not. When it is input adaptive, the network can be changed for each specific in-

Table 7 Recent representative network compression works for fast run-time inference. Please refer to the text for details about the key property indicators.

Reference	Key idea	Main approach	Key property indicator					Pros&Cons (if known)
			General or specific?	Static or dynamic?	Easy user control?	Input adaptive?	Resource aware?	
Liu et al.[292]	Directly enforcing channel-level sparsity	Scaling factors and sparsity-induced penalty	General	Static	Yes	No	Yes	Pros.: Simple and general. Cons.: Static (compression tied up training).
Lin et al.[318]; Rao et al.[319]	Model the pruning of each convolutional layer as a Markov decision process	Reinforcement learning	Seem to be general	Dynamic	Yes	Yes	Yes	Pros.: Good properties. Cons.: Overhead on pruning may be high.
Shazeer et al.[128]	Encourage components to be specialized and perform component selection during inference	A new network with multiple specialized branches, a gate for branching, and a combiner for aggregating	Seem to be general	Dynamic	Yes	Yes	Yes	Pros.: Intuitive, good properties. Cons.: Overhead on training may be high.
Gao et al.[320]	Use a low-overhead extra component to predict convolutional channels' saliency for prunning	A new piecewise differentiable and continuous function for saliency prediction	Seem to be general	Dynamic	No	Yes	No	Pros.: Simple, fast, currently lowest accuracy loss. Cons.: Not directly resource-aware.
Yu et al.[321]	Train a shared network with different widths	Switchable batch normalization	General (verified)	Dynamic	Yes	No	Yes	Pros.: Simple, clean, well-motivated, fast and easy to use. Cons.: Not input-adaptive.
Lee et al.[126]	Using a conditional gating module (CGM) to determine the use of each residual block according to the input image and the desired scale	CGM	Specific to residual networks	Dynamic	Yes	No	Yes	Pros.: Simple, clean, fast, and easy to use. Cons.: Not input-adaptive, and specific to residual networks.
Zhang and Jung[322]	Cost-adjustable inference by varying the unrolling steps of recurrent convolution (RC), with independently learned BN layers	Recurrent convolution (a particular kind of RNN)	Specific to RC networks	Dynamic	Yes	No	Yes	Pros.: New approach, interesting idea. Cons.: Not input-adaptive, and specific to RC networks.
Liu et al.[323]	Select a combination of compression techniques for an optimal balance between user-specified performance goals and resource constraints	Reinforcement learning	General (verified)	Dynamic	Yes	No	Yes	Pros.: Systematic, application oriented. Cons.: Not input-adaptive, and the overhead can be high.
Fang et al.[127]	Enables each DNN (by making it a multi-capacity model) to offer flexible resource-accuracy trade-offs, and then do resource-aware scheduling	A greedy heuristic approximation for optimizing MinTotalCost and MinMaxCost scheduling schemes	General	Dynamic	Yes	No	Yes	Pros.: Highly application oriented, general, little overhead. Cons.: Not input-adaptive; the overall solution looks complicated.
Pan et al.[123]	Using the similarity of the intermediate feature maps of two consecutive frames to largely reduce the redundant computation	The proposed novel recurrent residual module	Video-specific	Dynamic	No	Yes	No	Pros.: Properly explored redundancy in data, no accuracy loss. Cons.: Not user-controllable and not resource-aware.

put so that the inference time may be greatly reduced for easier inputs.

"Resource-aware?" Whether the compression is aware of the available resources (including resources for storage, transmission, and computation) or not. Fewer and weaker resources shall lead to a higher compression rate and vice versa. Resource types shall also matter in detailed control of the compression.

There are a few findings in our study which are worth mentioning.

1) Like the channel-wise sparsity work proposed by Liu et al.[292] or the slimmable neural networks proposed by Yu et al.[321], there is a clear trend that network pruning focuses more and more on whole channels or blocks[126], as such structural pruning is GPU-friendly (can be exploited by GPUs) and allows the acceleration to work on dense operations of fewer components, instead of many sparse individual weights.

2) Models that can be easily controlled by users ("Easy user control") are usually also "Resource-aware", as there is a common assumption that users can control the compression according to the actual situation of resources. However, in real cases, the state of resources can be dynamic even for a specific device instance (e.g., someone′s smartphone), as all the factors of storage, transmission, and computation can change over time. There can be multiple processes/applications running at the same time, and the internet connection speed is hard to be consistent. Therefore, instead of asking the user to specify the compression rate, having a model to dynamically decide it is a more practical and also more promising choice, which is widely ignored and far from being explored. Fang et al.[127] present an inspiring work in this direction.

3) Being "Resource-aware" and being "Input-adaptive" seem hard to be achieved at the same time, as the former is about overall compression, which can be controlled by users, while the latter is about automatically adjusting the compression based on each individual input data instance. However, these two can be solved from different aspects, and there is no conflict between them. Therefore, integrating both factors ("Resource-aware" and "Input-adaptive") is a promising direction worth further investigation, and there are also some good examples[128, 318, 319].

5.2 Efficient dynamic inference

After the training of DNN, traditional static models have fixed computational graphs and parameters in the inference stage. In contrast, to improve the computational efficiency of inference, there is an emerging research topic termed "dynamic neural network (DyNN)" that focuses on adaptively regulating network structures or parameters to different inputs in the inference stage. Recently, there has already been a comprehensive survey[18] on efficient dynamic inference with DNNs. However, it only focuses on classic analog-based DNNs where neurons use activations coded in continuous values. In this survey, we extend the concept of dynamic inference to brain-inspired SNNs where spiking neurons communicate through spike trains coded in binary events rather than continuous activations in analog-based DNNs[136]. Details on the motivations, strategies, and methods of these two cases are given below.

5.2.1 Analog-based dynamic inference

During inference, in many cases, there is no need to employ the same computational resource for each input, since the difficulty levels of the inputs are different. Therefore, efforts to reduce or avoid the computation of the irrelevant/redundant parts of the data are very important for efficient inference. The goal of DyNN is that less computation is spent on canonical samples that are relatively easy to recognize or on less informative spatial/temporal locations of an input. As the description of DyNN in [18] is already very comprehensive, here we only make a simple summary. Specifically, we categorize the works of DyNN into two orthogonal aspects based on the difference of focus: methods of DyNN and work dimensions of DyNN.

Methods of DyNN. Dynamic networks can adapt their structures or parameters to different inputs at the inference stage. Thus, dynamic structure and dynamic parameter are two basic methods to achieve efficient inference.

Dynamic structure models can selectively activate network components conditioned on the input, such as sub-networks[129, 324], layers[130, 325], or channels[320, 326]. For dynamic sub-network, there are two classic approaches to perform inference with dynamic architectures on each sample, including enabling early exiting in cascading multiple models and skipping branches in mixture-of-experts (MoE) via in parallel way. For example, a number of CNNs are cascaded in [327] and [328]. After each sub-network, a decision function is trained to decide whether the process should be an early exit. In contrast, in a parallel way[329], the MoEs can adopt real-valued soft weights to boost the representations obtained from different experts, or use binary-valued hard weights to increase the inference efficiency of the MoE. For dynamic layers (which can be simply viewed as dynamic depth), the intuitive motivation is that "easy" samples may not have to use the entire network to process, as modern DNNs are getting increasingly deep for recognizing more "hard" samples. Layer skipping is one of the most popular methods to obtain dynamic depth, which can exploit hard gates to efficiently produce binary decisions on whether to skip the computation of a residual block or layers[175]. Typical methods to achieve dynamic layers include SkipNet[325], Conv-AIG[330], CoDiNet[331], etc. In contrast to dynamic layers, another alternative idea is performing inference with dynamic channels, which can be seen as a kind of dynamic width. Based on the common belief that modern CNNs usually have considerable channel redundancy, the adaptive width of CNNs could be realized by dynamically activating convolutional channels for different samples, such as dynamic pruning[131, 293] with gating functions and dynamic pruning based on feature activations.

Compared with dynamic structure models, which usually need special designs of architecture, training

strategies, or careful hyper-parameters tuning, the dynamic parameter works to adapt network parameters to different inputs while keeping the architectures fixed. There are two common parameter adaption paths. Based on the specific input, one way is to adjust the trained parameters and the other way is to refine the features. A typical approach to obtaining parameter adjustment is using soft attention to regulate the weights based on their input during inference[132–135]. For example, soft attention can be executed on multiple convolutional kernels, producing an adaptive set of parameters[132, 133]. Similarly, feature refinement can be also achieved by the attention mechanism. Such dynamic regulations on network weights or features are easy to obtain with a minor increase in computational cost, and the representation power of networks will be significantly improved[332, 333].

Work dimensions of DyNN. Dynamic networks can perform adaptive computation at three different work granularities, i.e., sample-wise[129, 318], spatial-wise[334–336], and temporal-wise[66, 337]. Sample-wise dynamic models process each sample with the abovementioned data-dependent dynamic structures or parameters. Spatially adaptive and temporally adaptive models can also be viewed as sample-wise dynamic networks, since they execute adaptive computation within each sample at a finer granularity. In visual learning, the motivation for spatially dynamic computation is that not all locations contribute equally to the final prediction of CNNs[338]. Relevant approaches of spatially dynamic can be divided into three smaller levels: pixel level[334], region level[335], and resolution level[336]. Different from the spatial dimension, adaptive temporal-wise dynamic networks are dedicated to improving network efficiency by dynamically allocating less/no computation to the inputs at unimportant temporal locations[66, 340].

5.2.2 Brain-inspired dynamic inference

Spike-based temporal processing allows for sparse and efficient information transfer in the brain. To mimic the neuronal behaviors of the brain, SNN uses binary spike signals (0-nothing or 1-spike event) for inter-neuron event-driven communication. Each spiking neuron model realizes neuron-wise dynamic behaviors by aggregating spatial information from presynaptic neurons and temporal information from a leaky membrane potential, and only fires when the membrane potential exceeds a threshold. The entire spike signals of SNN are often sparse, and the computation can be smoothly executed on the sparse neuromorphic chip to avoid computing the zero values of input or activation. A classic brain-inspired spiking model, such as the most prominent leaky integrate-and-fire (LIF) neuron[8], is a trade-off between the complex dynamic characteristics of biological neurons and the simplified mathematical form. In this part, we try to re-understand the brain-inspired LIF-SNNs in the framework of dynamic networks. We focus on two questions "Is that SNN a dynamic neural network?" and "How does SNN work as a dynamic network?".

Is that SNN a dynamic neural network? SNN is also a kind of dynamic network. It naturally performs data-dependent dynamic inference that activates different sub-networks for different inputs, due to spike-based neuron-wise dynamic activation. The smallest neural network has only one neuron. In this case, the neuron-wise dynamic network can decide whether to activate the neuron according to the input. For spiking neurons, if there are no input spikes or the membrane potential after synapse accumulation is less than the threshold, the spiking neuron will not be activated[8]. More importantly, this neuron-wise dynamic function of SNN can be smoothly executed on the neuromorphic chip[136]. Spiking neurons are connected hierarchically forming an SNN. We already know that dynamic networks can adapt their structures or parameters to different inputs at the inference stage. Due to the neuron-wise dynamic characteristic of SNN, it activates various sub-networks and parameters for different inputs. Therefore, SNN is a dynamic neural network.

How does SNN work as a dynamic network? When the total simulation steps $T = 1$, SNN can be approximated as a spike-based CNN without temporal information, and the first layer of SNN can be regarded as a spiking encoder layer. SNN certainly has the spatial-wise dynamic because of the unique neuron-wise sparse dynamic activation. It supports the pixel-level spatial-wise dynamic, which performs convolutions only on sampled pixels set. The advantage of spatial-wise dynamic in SNN has been widely used in event-based vision. Based on frame-based representation[52], the event stream is converted into a video-like sequence with many zero areas in each frame by frame-based representation, and non-zero areas of the event-based frame can be considered as informative pixels. For each frame, SNN can skip the computation of zero areas. Without any additional auxiliary controller, SNN is naturally a temporal-wise dynamic network, since it has the unique finest granularity neuron-wise dynamic. At each time step, SNN performs data-dependent processing that only activates a part of spiking neurons, i.e., sub-networks. In summary, SNN can naturally work on all three different dynamic granularities (i.e., spatial-wise, temporal-wise, and sample-wise) without any additional auxiliary controller. The reason is that spiking neurons naturally have a neuron-wise dynamic, while analog-based DNNs do not[18].

6 Summary and discussions

6.1 Summary of recent advances

As detailed in the former sections and also summarized in Fig. 21, to design an efficient visual recognition model or system, one may put effort into data processing (compression, selection, and representation), which is usually data type specific, network compression that consists

of multiple methods, and efficient inference which includes fast run-time and dynamic way, as already explored by the rich literature in the past few years.

An important message is that a lot of things can be done on the data side or having it taken into account when people design new efficient models, which is not only important for the recognition performance but also critical for ensuring that the network compression fits the data well. In this sense, combining the data side and network side together as one will be a significant frontier field, and each side should be dug deeper separately to promote this target as well.

6.2 Unexplored yet promising new directions

Though important and attractive recently-emerged directions have already been discussed in the former sections when individual topics are introduced, there might still be some unexplored yet promising directions worth mentioning for preparing future data and network co-optimization. Within them, the following two are believed by us to be most valuable, though it is still challenging to work on.

6.2.1 Efficient network compression

Network compression is now a vibrant research subject, especially orienting to visual recognition neural networks, which are almost always large-scaled due to high-dimensional raw visual data. However, according to our investigation, we summarize several promising directions of compression methods below, which may further promote the miniaturization of DNNs and brain-inspired SNNs further.

1) NAS may become a promising or even requisite approach, especially for embedded vision applications. Such a particular approach is very suited to the application scenarios with variable surroundings, e.g., fault diagnosis based on visual information, ground object identification based on aerial photography, etc. However, most existing NAS methods are expensive, so optimizing resource cost and searching time should be further researched.

2) Some tensor decomposition methods have not been studied adequately due to their complexity, e.g., HT, BTD, Kronecker tensor decomposition (KTD)[239, 339], etc. Researching these tensor formats may imply potential unrevealed prospects for neural network compression. For instance, Wang et al.[163] have proposed the KCP-RNN, which first achieved both space and computation complexity at the same time, and they claimed that the fine-grained tensor decomposition, such as KCP should be the future.

3) As different compression methods have different characteristics, it is natural to expect a super-synthetical compression method, which contains the flexibility of NAS, the regular architecture of tensor decomposition, efficient computing of quantization in embedded surroundings, and high accuracy of pruning, could be proposed in the future. In other words, there is still much work to be done to reach the culmination of network compression.

4) Except for DNNs, the miniaturization of the brain-inspired SNNs has a great practical significance as well, and such kinds of works are still lacking. On the other hand, SNN can be seen as a specific kind of DNN since its motivation of invention is to borrow the efficiency of bio-

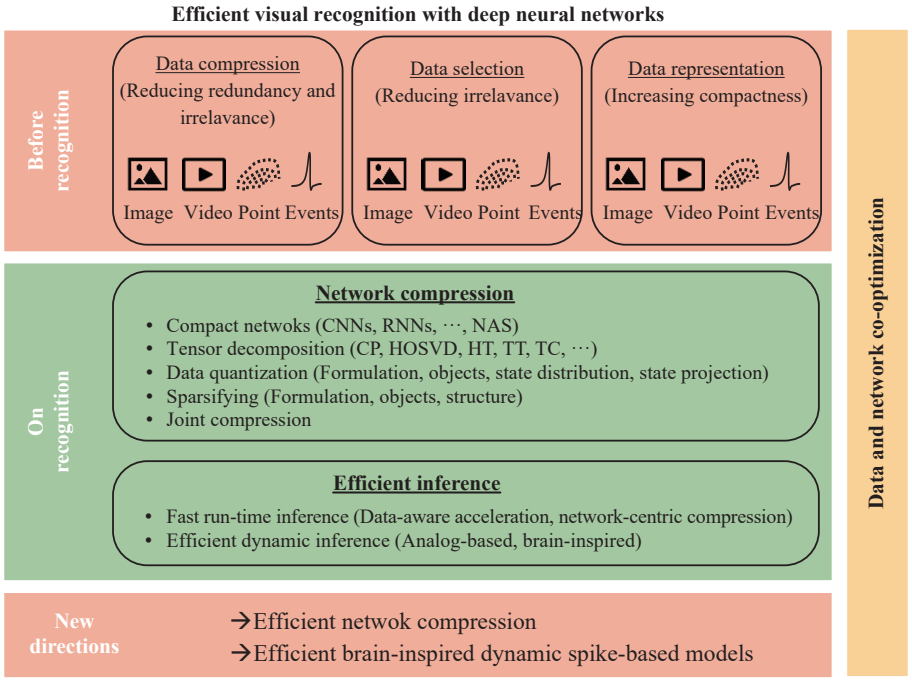


Fig. 21 Overview of the recent advances and new directions

neurons, and current brain-inspired chips[340, 341] can further highlight the advantages of SNNs. Hence, how to land any compression methods to SNNs and further to various brain-inspired hardware has a broad prospect to study.

6.2.2 Efficient brain-inspired dynamic spike-based models

Though analog-based dynamic DNNs have been widely studied due to their notable advantages in terms of accuracy, computational efficiency, adaptiveness, etc., the inherent gap between theoretical and practical efficiency is insurmountable. It is induced by a sparse dynamic computation that runs on dense hardware such as a GPU[18]. By contrast, as described in Section 5.2, performing dynamic sparse brain-inspired SNNs on sparse computation neuromorphic hardware is natural without the gap between algorithm and hardware at the aspect of dynamic computation.

As far as we are aware, there is only one published work related to dynamic SNN, i.e., the TA-SNN work proposed by Yao et al. on ICCV 2021[52]. Based on the observation that the accuracy of the SNN would not become worse even if they masked half of the input events, they used a lightweight temporal-wise attention module to handle event streams efficiently by discarding irrelevant events. In this survey, we present some qualitative analysis about the relationship between brain-inspired SNN and dynamic networks, and explore two basic questions "Is that SNN a dynamic neural network?" and "How dose SNN work as a dynamic network?". We hope that these discussions would further inspire the effectiveness and efficiency of network architecture design in the SNN domain.

6.3 An important new frontier: Data and network co-optimization

Recently, some methods actually have collaboratively optimized the data and networks together, although we reviewed them separately as "Before recognition" and "On recognition" above. For example, in a previously cited paper[141], the network is cascaded by a data compression network and a classification network and trained in end-to-end. Thus, the compression network can compress images more efficiently, retain classification information, and also improve classification efficiency. Similarly, in [48], the compressed video is used as the input of the action recognition network, and the decoding function is assigned to the recognition network through training. Therefore, the network learns the abilities for decoding and recognition at the same time through end-to-end training.

In fact, end-to-end training has been a common desire for DNN-based and SNN-based solutions. However, even though joint compression and recognition are already made possible, so far, most of the existing solutions have still treated them as two separate yet linked modules rather than a single fused model. Compared with linking them for co-optimization, fusing them as a whole model may be more helpful for developing more computationally efficient models with minimum redundant computations for real applications, even though that may lead to more space consumption due to the uncompressed data, which are nowadays relatively easier to handle. As we have not found any corresponding practices to the best of our knowledge, based on the new directions mentioned in the last Section 6.2, here we propose two possible aspects for such a unified model design.

One aspect is to explore quantization: Linking data quantization and network quantization. As discussed earlier in Section 4.3 (especially Fig. 18 and Table 4) and Section 4.5, quantization can be done for the whole data flow inside the neural networks, and it is quite superior for adapting to various hardware platform. Thus, it is closer to real applications. Meanwhile, currently, most visual data are originally represented by low-bit integers thanks to the advancement of digital visual sensors, including the brain-inspired DVS, which make data quantization convenient and straight-forward. However, the main barrier to linking these two is that currently, the raw visual data format may not directly match the quantized network data types. Therefore, we think necessary and highly valuable future efforts should be either on transforming the representation of raw data towards that of the quantized models or going to the extreme to make the new visual sensors able to produce various or more easily adjustable raw data formats for the integration with quantization. At present, one of the most promising instances might be the binary SNNs, since both the spiking data and quantized weights are binary, and such kinds of models might be more amazing on brain-inspired chips[340, 341].

The other aspect is to extend the tensor network to cover the input data. Tensor networks can inherently describe linear transformations, e.g., matrix production and tensor contraction. Meanwhile, a neural network architecture is generally similar to a tensor network except for all the nonlinear activation functions. Therefore, tensor networks can at least inspire us to develop some new neural architectures by analyzing the probable relationship between tensor networks and neural networks, even though the strict mapping from tensor network to neural network may be difficult. Thus, compressed input data and a compressed neural architecture can be both put into a system similar to a tensor network. For example, if the input data and the weights in a neural network are both approximated in TT format, the whole system can be expressed like a projected entangled-pair states (PEPS) tensor network[342], as drawn in Fig. 22(a) where red and green nodes illustrate input and output data, respectively, and the existing efficient computing algorithm, i.e., Algorithm 5 in [250], may help to accelerate this kind

of models. Furthermore, if one layer of the neural network is designed like a multi-scale entanglement renormalization ansatz (MERA)[307], a higher ability of expression may be obtained, as presented in Fig. 22(b). It is easy to observe that the neural network in the MERA architecture has a stronger local correlation than PEPS, e.g., the content of $\mathcal{O}_2$ in Fig. 22(b) comprises the information from $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{G}_3$, and $\mathcal{G}_4$. Through such efforts, the efficiency of data representation and network may be optimized using tensor network models with theoretical support, and there are already a few prospective works[306, 343].

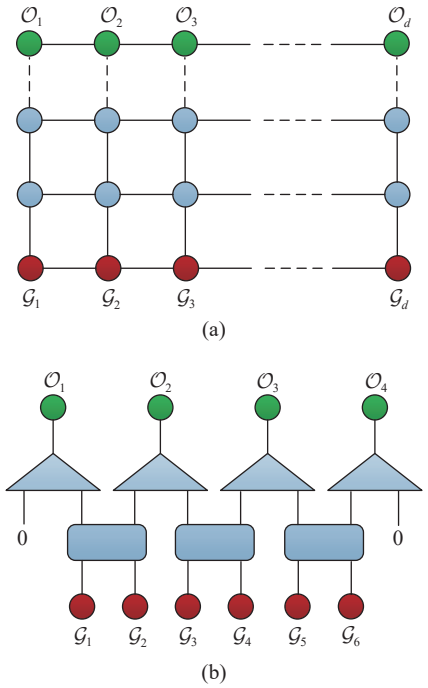


Fig. 22 Input data (denoted by the red nodes $\mathcal{G}_1$, $\mathcal{G}_2$, $\cdots$) and weights in a neural network may be uniformly modeled as a tensor network, with typical examples such as (a) PEPS[342] and (b) MERA[307].

Acknowledgements

This work was supported by National Key R&D Program of China (No. 2018AAA0102600), Beijing Natural Science Foundation, China (No. JQ21015), Beijing Academy of Artificial Intelligence (BAAI), China, and Pengcheng Laboratory, China.

Open Access

References

[1] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. DOI: 10.1109/5.726791.

[2] G. E. Hinton, R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, vol. 313, no. 5786, pp. 504–507, 2006. DOI: 10.1126/science.1127647.

[3] A. Krizhevsky, I. Sutskever, G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, Lake Tahoe, USA, pp. 1106–1114, 2012.

[4] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the 13th European Conference on Computer Vision*, Springer, Zurich, Switzerland, pp. 740–755, 2014. DOI: 10.1007/978-3-319-10602-1_48.

[5] J. K. Song, Y. Y. Guo, L. L. Gao, X. L. Li, A. Hanjalic, H. T. Shen. From deterministic to generative: Multimodal stochastic RNNs for video captioning. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 10, pp. 3047–3058, 2019. DOI: 10.1109/TNNLS.2018.2851077.

[6] L. L. Gao, X. P. Li, J. K. Song, H. T. Shen. Hierarchical LSTMs with adaptive attention for visual captioning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 5, pp. 1112–1131, 2020. DOI: 10.1109/TPAMI.2019.2894139.

[7] S. E. Wei, V. Ramakrishna, T. Kanade, Y. Sheikh. Convolutional pose machines. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, USA, pp. 4724–4732, 2016. DOI: 10.1109/CVPR.2016.511.

[8] W. Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997. DOI: 10.1016/S0893-6080(97)00011-7.

[9] E. Ahmed, A. Saint, A. E. R. Shabayek, K. Cherenkova, R. Das, G. Gusev, D. Aouada, B. Ottersten. A survey on deep learning advances on different 3D data representations. [Online], Available: https://arxiv.org/abs/1808.01462, 2019.

[10] L. Liu, J. Chen, P. Fieguth, G. Y. Zhao, R. Chellappa, M. Pietikäinen. From bow to CNN: Two decades of texture representation for texture classification. *International*

Journal of Computer Vision, vol. 127, no. 1, pp. 74–109, 2019. DOI: 10.1007/s11263-018-1125-z.

[11] L. Liu, W. L. Ouyang, X. G. Wang, P. Fieguth, J. Chen, X. W. Liu, M. Pietikäinen. Deep learning for generic object detection: A survey. *International Journal of Computer Vision*, vol. 128, no. 2, pp. 261–318, 2020. DOI: 10.1007/s11263-019-01247-4.

[12] G. Gallego, T. Delbruük, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, D. Scaramuzza. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, 2022. DOI: 10.1109/TPAMI.2020.3008413.

[13] Q. R. Zhang, M. Zhang, T. H. Chen, Z. F. Sun, Y. Z. Ma, B. Yu. Recent advances in convolutional neural network acceleration. *Neurocomputing*, vol. 323, pp. 37–51, 2019. DOI: 10.1016/j.neucom.2018.09.038.

[14] L. Deng, G. Q. Li, S. Han, L. P. Shi, Y. Xie. Model compression and hardware acceleration for neural networks: A comprehensive survey. *Proceedings of IEEE*, vol. 108, no. 4, pp. 485–532, 2020. DOI: 10.1109/JPROC.2020.2976475.

[15] Y. Cheng, D. Wang, P. Zhou, T. Zhang. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126–136, 2018. DOI: 10.1109/MSP.2017.2765695.

[16] V. Lebedev V. Lempitsky. Speeding-up convolutional neural networks: A survey. *Bulletin of the Polish Academy of Sciences*: *Technical Sciences*, vol. 66, no. 6, pp. 799–810, 2018. DOI: 10.24425/bpas.2018.125927.

[17] T. Elsken, J. H. Metzen, F. Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 1997–2017, 2019. DOI: 10.5555/3322706.3361996.

[18] Y. Z. Han, G. Huang, S. J. Song, L. Yang, H. H. Wang, Y. L. Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, to be published. DOI: 10.1109/TPAMI.2021.3117837.

[19] P. Lichtsteiner, C. Posch, T. Delbruck. A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-state Circuits*, vol. 43, no. 2, pp. 566–576, 2008. DOI: 10.1109/JSSC.2007.914337.

[20] C. Posch, D. Matolin, R. Wohlgenannt. A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS. *IEEE Journal of Solid-state Circuits*, vol. 46, no. 1, pp. 259–275, 2011. DOI: 10.1109/JSSC.2010.2085952.

[21] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images, Master dissertation, University of Toronto, Canada, 2009.

[22] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Miami, USA, pp. 248–255, 2009. DOI: 10.1109/CVPR.2009.5206848.

[23] Y. Xiang, W. Kim, W. Chen, J. W. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, S. Savarese. ObjectNet3D: A large scale database for 3D object recognition. In *Proceedings of the 14th European Conference on Computer Vision*, Springer, Amsterdam, The Netherlands, pp. 160–176, 2016. DOI: 10.1007/978-3-319-46484-8_10.

[24] A. R. Zamir, A. Sax, W. Shen, L. Guibas, J. Malik, S. Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 3712–3722, 2018. DOI: 10.1109/CVPR.2018.00391.

[25] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, M. J. Black. Towards understanding action recognition. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Sydney, Australia, pp. 3192–3199, 2013. DOI: 10.1109/ICCV.2013.396.

[26] A. Shahroudy, J. Liu, T. T. Ng, G. Wang. NTU RGB+D: A large scale dataset for 3D human activity analysis. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, USA, pp. 1010–1019, 2016. DOI: 10.1109/CVPR.2016.115.

[27] C. H. Liu, Y. Y. Hu, Y. H. Li, S. J. Song, J. Y. Liu. PKU-MMD: A large scale benchmark for continuous multi-modal human action understanding. [Online], Available: https://arxiv.org/abs/1703.07475, 2017.

[28] Y. S. Tang, Y. Tian, J. W. Lu, P. Y. Li, J. Zhou. Deep progressive reinforcement learning for skeleton-based action recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, 2018, pp. 5323–5332. DOI: 10.1109/CVPR.2018.00558.

[29] J. X. Hou, G. J. Wang, X. H. Chen, J. H. Xue, R. Zhu, H. Z. Yang. Spatial-temporal attention res-TCN for skeleton-based dynamic hand gesture recognition. In *Proceedings of Computer Vision*, Springer, Munich, Germany, pp. 273–286, 2018. DOI: 10.1007/978-3-030-11024-6_18.

[30] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. X. Huang, Z. M. Li, S. Savarese, M. Savva, S. R. Song, H. Su, J. X. Xiao, L. Yi, F. Yu. ShapeNet: An information-rich 3D model repository. [Online], Available: https://arxiv.org/abs/1512.03012, 2015.

[31] H. Rebecq, R. Ranftl, V. Koltun, D. Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 6, pp. 1964–1980, 2021. DOI: 10.1109/TPAMI.2019.2963386.

[32] W. S. Cheng, H. Luo, W. Yang, L. Yu, S. S. Chen, W. Li. Det: A high-resolution DVS dataset for lane extraction. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, Long Beach, USA, pp. 1666–1675, 2019. DOI: 10.1109/CVPRW.2019.00210.

[33] T. Delbruck, M. Lang. Robotic goalie with 3 ms reaction time at 4% CPU load using event-based dynamic vision sensor. *Frontiers in Neuroscience*, vol. 7, Article number 223, 2013. DOI: 10.3389/fnins.2013.00223.

[34] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, D. Modha. A low power, fully event-based gesture recognition system. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 7388–7397, 2017. DOI: 10.1109/CVPR.2017.781.

[35] Z. Wu, Z. Xu, R. N. Zhang, S. M. Li. SIFT feature extraction algorithm for image in DCT domain. *Applied Mech-*

anics and Materials, vol. 347–350, pp. 2963–2967, 2013. DOI: 10.4028/www.scientific.net/AMM.347-350.2963.

[36] L. Gueguen, A. Sergeev, B. Kadlec, R. Liu, J. Yosinski. Faster neural networks straight from jpeg. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 3937–3948, 2018. DOI: 10.5555/3327144.3327308.

[37] A. Paul, T. Z. Khan, P. Podder, R. Ahmed, M. M. Rahman, M. H. Khan. Iris image compression using wavelets transform coding. In *Proceedings of the 2nd International Conference on Signal Processing and Integrated Networks*, IEEE, Noida, India, pp. 544–548, 2015. DOI: 10.1109/SPIN.2015.7095407.

[38] O. Rippel, L. Bourdev. Real-time adaptive image compression. In *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, pp. 2922–2930, 2017. DOI: 10.5555/3305890.3305983.

[39] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, N. Johnston. Variational image compression with a scale hyperprior. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, pp. 1–49, 2018.

[40] D. Minnen, G. Toderici, S. Singh, S. J. Hwang, M. Covell. Image-dependent local entropy models for learned image compression. In *Proceedings of the 25th IEEE International Conference on Image Processing*, IEEE, Athens, Greece, pp. 430–434, 2018. DOI: 10.1109/ICIP.2018.8451502.

[41] D. Minnen, J. Ballé, G. D. Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 10794–10803, 2018. DOI: 10.5555/3327546.3327736.

[42] G. J. Sullivan, J. R. Ohm, W. J. Han, T. Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012. DOI: 10.1109/TCSVT.2012.2221191.

[43] T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra. Overview of the H.264/AVC video coding standard.. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003. DOI: 10.1109/TCSVT.2003.815165.

[44] T. Chen, H. J. Liu, Q. Shen, T. Yue, X. Cao, Z. Ma. Deepcoder: A deep neural network based video compression. In *Proceedings of IEEE Visual Communications and Image Processing*, IEEE, St. Petersburg, USA, pp. 1–4, 2017. DOI: 10.1109/VCIP.2017.8305033.

[45] G. Lu, W. L. Ouyang, D. Xu, X. Y. Zhang, Z. Y. Gao, M. T. Sun. Deep Kalman filtering network for video compression artifact reduction. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 591–608, 2018. DOI: 10.1007/978-3-030-01264-9_35.

[46] C. Y. Wu, N. Singhal, P. Krähenbühl. Video compression through image interpolation. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 425–440, 2018. DOI: 10.1007/978-3-030-01237-3_26.

[47] X. Z. Zhu, Y. W. Xiong, J. F. Dai, L. Yuan, Y. C. Wei. Deep feature flow for video recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 4141–4150, 2017. DOI: 10.1109/CVPR.2017.441.

[48] C. Y. Wu, M. Zaheer, H. X. Hu, R. Manmatha, A. J. Smola, P. Krähenbuühl. Compressed video action recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 6026–6035, 2018. DOI: 10.1109/CVPR.2018.00631.

[49] W. Yan, Y. T. Shao, S. Liu, T. H. Li, Z. Li, G. Li. Deep AutoEncoder-based lossy geometry compression for point clouds. [Online], Available: https://arxiv.org/abs/1905.03691, 2019.

[50] J. Q. Wang, H. Zhu, Z. Ma, T. Chen, H. J. Liu, Q. Shen. Learned point cloud geometry compression. [Online], Available: https://arxiv.org/abs/1909.12037, 2019.

[51] Y. Q. Yang, C. Feng, Y. R. Shen, D. Tian. FoldingNet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 206–215, 2018. DOI: 10.1109/CVPR.2018.00029.

[52] M. Yao, H. H. Gao, G. S. Zhao, D. S. Wang, Y. H. Lin, Z. X. Yang, G. Q. Li. Temporal-wise attention spiking neural networks for event streams classification. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, IEEE, Montréal, Canada, pp. 10201–10210, 2021. DOI: 10.1109/ICCV48922.2021.01006.

[53] Y. X. Wang, B. W. Du, Y. R. Shen, K. Wu, G. R. Zhao, J. G. Sun, H. K. Wen. EV-Gait: Event-based robust gait recognition using dynamic vision sensors. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 6351–360, 2019. DOI: 10.1109/CVPR.2019.00652.

[54] Y. Sekikawa, K. Hara, H. Saito. EventNet: Asynchronous recursive event processing. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 3882–3891, 2019. DOI: 10.1109/CVPR.2019.00401.

[55] K. Chitta, J. M. Alvarez, E. Haussmann, C. Farabet. Training data subset search with ensemble active learning. [Online], Available: https://arxiv.org/abs/1905.12737, 2020.

[56] O. Sener, S. Savarese. Active learning for convolutional neural networks: A core-set approach. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[57] K. Vodrahalli, K. Li, J. Malik. Are all training examples created equal? An empirical study. [Online], Available: https://arxiv.org/abs/1811.12569, 2018.

[58] V. Birodkar, H. Mobahi, S. Bengio. Semantic redundancies in image-classification datasets: The 10% you don't need. [Online], Available: https://arxiv.org/abs/1901.11409, 2019.

[59] J. Y. Gao, Z. H. Yang, C. Sun, K. Chen, R. Nevatia. TURN TAP: Temporal unit regression network for temporal action proposals. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 3648–3656, 2017. DOI: 10.1109/ICCV.2017.392.

[60] J. Carreira, A. Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *Proceedings of IEEE Conference on Computer Vision and Pattern Re-*

cognition, IEEE, Honolulu, USA, pp. 4724–4733, 2017. DOI: 10.1109/CVPR.2017.502.

[61] S. N. Xie, C. Sun, J. Huang, Z. W. Tu, K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 318–335, 2018. DOI: 10.1007/978-3-030-01267-0_19.

[62] M. Zolfaghari, K. Singh, T. Brox. ECO: Efficient convolutional network for online video understanding. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 713–730, 2018. DOI: 10.1007/978-3-030-01216-8_43.

[63] S. Yeung, O. Russakovsky, G. Mori, L. Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, USA, pp. 2678–2687, 2016. DOI: 10.1109/CVPR.2016.293.

[64] J. J. Huang, N. N. Li, T. Zhang, G. Li, T. J. Huang, W. Gao. SAP: Self-adaptive proposal model for temporal action detection based on reinforcement learning. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, New Orleans, USA, pp. 6951–6958, 2018. DOI: 10.1609/aaai.v32i1.12229.

[65] S. Y. Lan, R. Panda, Q. Zhu, A. K. Roy-Chowdhury. FFNet: Video fast-forwarding via reinforcement learning. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 6771–6780, 2018. DOI: 10.1109/CVPR.2018.00708.

[66] H. H. Fan, Z. W. Xu, L. C. Zhu, C. G. Yan, J. J. Ge, Y. Yang. Watching a small portion could be as good as watching all: Towards efficient video classification. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, pp. 705–711, 2018. DOI: 10.5555/3304415.3304516.

[67] A. Kar, N. Rai, K. Sikka, G. Sharma. AdaScan: Adaptive scan pooling in deep convolutional neural networks for human action recognition in videos. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 5699–5708, 2017. DOI: 10.1109/CVPR.2017.604.

[68] Z. X. Wu, C. M. Xiong, C. Y. Ma, R. Socher, L. S. Davis. AdaFrame: Adaptive frame selection for fast video recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 1278–1287, 2019. DOI: 10.1109/CVPR.2019.00137.

[69] J. C. Yang, Q. Zhang, B. B. Ni, L. G. Li, J. X. Liu, M. D. Zhou, Q. Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 3318–3327, 2019. DOI: 10.1109/CVPR.2019.00344.

[70] A. Paigwar, O. Erkent, C. Wolf, C. Laugier. Attentional pointNet for 3D-object detection in point clouds. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, Long Beach, USA, pp. 1297–1306, 2019. DOI: 10.1109/CVPRW.2019.00169.

[71] C. Kingkan, J. Owoyemi, K. Hashimoto. Point attention network for gesture recognition using point cloud data. In *Proceedings of the 29th British Machine Vision Conference*, Newcastle, UK, pp. 1–13, 2018. [Online], Available: https://bmvc2018.org/contents/papers/0427.pdf.

[72] A. Khodamoradi, R. Kastner. $O(N)$o(N)-space spatiotemporal filter for reducing noise in neuromorphic vision sensors. *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 1, pp. 15–23, 2021. DOI: 10.1109/TETC.2017.2788865.

[73] H. J. Liu, C. Brandli, C. H. Li, S. C. Liu, T. Delbruck. Design of a spatiotemporal correlation filter for event-based sensors. In *Proceedings of IEEE International Symposium on Circuits and Systems*, IEEE, Lisbon, Portugal, pp. 722–725, 2015. DOI: 10.1109/ISCAS.2015.7168735.

[74] V. Padala, A. Basu, G. Orchard. A noise filtering algorithm for event-based asynchronous change detection image sensors on trueNorth and its implementation on TrueNorth. *Frontiers in Neuroscience*, vol. 12, pp. 1–14, 2018. DOI: 10.3389/fnins.2018.00118.

[75] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, J. M. Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016. DOI: 10.1109/TMI.2016.2535302.

[76] U. K. Lopes, J. F. Valiati. Pre-trained convolutional neural networks as feature extractors for tuberculosis detection. *Computers in Biology and Medicine*, vol. 89, pp. 135–143, 2017. DOI: 10.1016/j.compbiomed.2017.08.001.

[77] O. J. Hénaff, A. Srinivas, J. De Fauw, A. Razavi, C. Doersch, S. M. A. Eslami, A. van den Oord. Data-efficient image recognition with contrastive predictive coding. In *Proceedings of the 37th International Conference on Machine Learning*, Vienna, Austria, vol. 119, pp. 4182–4192, 2020. DOI: 10.5555/3524938.3525329.

[78] A. S. Razavian, H. Azizpour, J. Sullivan, S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, Columbus, USA, pp. 512–519, 2014. DOI: 10.1109/CVPRW.2014.131.

[79] Y. Wu, J. Qiu, J. Takamatsu, T. Ogasawara. Temporal-enhanced convolutional network for person re-identification. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, New Orleans, USA, pp. 7412–7419, 2018. DOI: 10.1609/aaai.v32i1.12264.

[80] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, S. Gould. Dynamic image networks for action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, USA, pp. 3034–3042, 2016. DOI: 10.1109/CVPR.2016.331.

[81] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi. Action recognition with dynamic image networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 2799–2813, 2018. DOI: 10.1109/TPAMI.2017.2769085.

[82] F. Yang, Y. Wu, S. Sakti, S. Nakamura. Make skeleton-based action recognition model smaller, faster and better. In *Proceedings of ACM Multimedia Asia*, ACM, Beijing, China, Article number 31, 2019. DOI: 10.1145/3338533.3366569.

[83] C. Li, Q. Y. Zhong, D. Xie, S. L. Pu. Co-occurrence feature learning from skeleton data for action recognition

and detection with hierarchical aggregation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, pp. 786–792, 2018. DOI: 10.5555/3304415.3304527.

[84] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, USA, pp. 2818–2826, 2016. DOI: 10.1109/CVPR.2016.308.

[85] P. Q. Wang, P. F. Chen, Y. Yuan, D. Liu, Z. H. Huang, X. D. Hou, G. Cottrell. Understanding convolution for semantic segmentation. In *Proceedings of IEEE Winter Conference on Applications of Computer Vision*. IEEE, Lake Tahoe, USA, pp. 1451–1460, 2018. DOI: 10.1109/WACV.2018.00163.

[86] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, K. Keutzer. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size. In *Proceedings of the 5th International Conference on Learning Representations*, [Online], Available: https://arxiv.org/abs/1602.07360, 2016.

[87] X. Y. Zhang, X. Y. Zhou, M. X. Lin, J. Sun. ShuffleNet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 6848–6856, 2018. DOI: 10.1109/CVPR.2018.00716.

[88] F. Juefei-Xu, V. N. Boddeti, M. Savvides. Perturbative neural networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 3310–3318, 2018. DOI: 10.1109/CVPR.2018.00349.

[89] F. Juefei-Xu, V. N. Boddeti, M. Savvides. Local binary convolutional neural networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 4284–4293, 2017. DOI: 10.1109/CVPR.2017.456.

[90] Z. Z. Wu, S. M. King. Investigating gated recurrent neural networks for speech synthesis. [Online], Available: https://arxiv.org/abs/1601.02539, 2016.

[91] J. van der Westhuizen, J. Lasenby. The unreasonable effectiveness of the forget gate. [Online], Available: https://arxiv.org/abs/1804.04849, 2018.

[92] H. Sak, A. W. Senior, F. Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Proceedings of the 15th Annual Conference of the International Speech Communication Association*, Singapore, pp. 338–342, 2014.

[93] Y. H. Wu, M. Schuster, Z. F. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. B. Liu, Ł. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, J. Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. [Online], Available: https://arxiv.org/abs/1609.08144, 2016.

[94] B. Zoph, Q. V. Le. Neural architecture search with reinforcement learning. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[95] E. Real, A. Aggarwal, Y. P. Huang, Q. V. Le. Regularized evolution for image classifier architecture search. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence, the 31st Innovative Applications of Artificial Intelligence Conference, and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence*, Honolulu, USA, pp. 4780–4789, 2019. DOI: 10.1609/aaai.v33i01.33014780.

[96] K. Kandasamy, W. Neiswanger, J. Schneider, B. Póczos, E. P. Xing. Neural architecture search with Bayesian optimisation and optimal transport. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 2020–2029, 2018. DOI: 10.5555/3326943.3327130.

[97] H. Cai, L. G. Zhu, S. Han. Proxylessnas: Direct neural architecture search on target task and hardware. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[98] M. Astrid, S. I. Lee. Cp-decomposition with tensor power method for convolutional neural networks compression. In *Proceedings of IEEE International Conference on Big Data and Smart Computing*, IEEE, Jeju, Korea, pp. 115–118, 2017. DOI: 10.1109/BIGCOMP.2017.7881725.

[99] J. T. Chien, Y. T. Bao. Tensor-factorized neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 5, pp. 1998–2011, 2018. DOI: 10.1109/TNNLS.2017.2690379.

[100] J. M. Ye, L. N. Wang, G. X. Li, D. Chen, S. D. Zhe, X. Q. Chu, Z. L. Xu. Learning compact recurrent neural networks with block-term tensor decomposition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 9378–9387, 2018. DOI: 10.1109/CVPR.2018.00977.

[101] A. Novikov, D. Podoprikhin, A. Osokin, D. P. Vetrov. Tensorizing neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 442–450, 2015. DOI: 10.5555/2969239.2969289.

[102] T. Garipov, D. Podoprikhin, A. Novikov, D. Vetrov. Ultimate tensorization: Compressing convolutional and FC layers alike. [Online], Available: https://arxiv.org/abs/1611.03214, 2016.

[103] D. H. Wang, G. S. Zhao, G. Q. Li, L. Deng, Y. Wu. Compressing 3DCNNs based on tensor train decomposition. *Neural Networks*, vol. 131, pp. 215–230, 2020. DOI: 10.1016/j.neunet.2020.07.028.

[104] A. Tjandra, S. Sakti, S. Nakamura. Compressing recurrent neural network with tensor train. In *Proceedings of International Joint Conference on Neural Networks*, IEEE, Anchorage, USA, pp. 4451–4458, 2017. DOI: 10.1109/IJCNN.2017.7966420.

[105] Y. C. Yang, D. Krompass, V. Tresp. Tensor-train recurrent neural networks for video classification. In *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, pp. 3891–3900, 2017. DOI: 10.5555/3305890.3306083.

[106] Y. Pan, J. Xu, M. L. Wang, J. M. Ye, F. Wang, K. Bai, Z. L. Xu. Compressing recurrent neural networks with tensor ring for action recognition. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence, the 31st Innovative Applications of Artificial Intelligence Conference, and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence*, Honolulu, USA,

pp. 4683–4690, 2019. DOI: 10.1609/aaai.v33i01.33014683.

[107] B. J. Wu, D. H. Wang, G. S. Zhao, L. Deng, G. Q. Li. Hybrid tensor decomposition in neural network compression. *Neural Networks*, vol. 132, pp. 309–320, 2020. DOI: 10.1016/j.neunet.2020.09.006.

[108] M. Yin, S. Y. Liao, X. Y. Liu, X. D. Wang, B. Yuan. Compressing recurrent neural networks using hierarchical tucker tensor decomposition. [Online], Available: https://arxiv.org/abs/2005.04366, 2020.

[109] S. Wu, G. Q. Li, F. Chen, L. P. Shi. Training and inference with integers in deep neural networks. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[110] Y. K. Yang, L. Deng, S. Wu, T. Y. Yan, Y. Xie, G. Q. Li. Training high-performance and large-scale deep neural networks with full 8-bit integers. *Neural Networks*, vol. 125, pp. 70–82, 2020. DOI: 10.1016/j.neunet.2019.12.027.

[111] M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi. XNOR-Net: ImageNet classification using binary convolutional neural networks. In *Proceedings of the 14th European Conference on Computer Vision*. Springer, Amsterdam, The Netherlands, pp. 525–542, 2016. DOI: 10.1007/978-3-319-46493-0_32.

[112] Q. Lou, F. Guo, M. Kim, L. T.s Liu, L. Jiang. AutoQ: Automated kernel-wise neural network quantization. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

[113] Y. Y. Lin, C. Sakr, Y. Kim, N. Shanbhag. PredictiveNet: An energy-efficient convolutional neural network via zero prediction. In *Proceedings of IEEE International Symposium on Circuits and Systems*, IEEE, Baltimore, USA, 2017. DOI: 10.1109/ISCAS.2017.8050797.

[114] M. C. Song, J. C. Zhao, Y. Hu, J. Q. Zhang, T. Li. Prediction based execution on deep neural networks. In *Proceedings of the 45th ACM/IEEE Annual International Symposium on Computer Architecture*, IEEE, Los Angeles, USA, pp. 752–763, 2018. DOI: 10.1109/ISCA.2018.00068.

[115] V. Akhlaghi, A. Yazdanbakhsh, K. Samadi, R. K. Gupta, H. Esmaeilzadeh. SnaPEA: Predictive early activation for reducing computation in deep convolutional neural networks. In *Proceedings of the 45th ACM/IEEE Annual International Symposium on Computer Architecture*, IEEE, Los Angeles, USA, pp. 662–673, 2018. DOI: 10.1109/ISCA.2018.00061.

[116] W. Wen, C. P. Wu, Y. D. Wang, Y. R. Chen, H. Li. Learning structured sparsity in deep neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Barcelona, Spain, pp. 2082–2090, 2016. DOI: 10.5555/3157096.3157329.

[117] J. H. Luo, J. X. Wu, W. Y. Lin. ThiNet: A filter level pruning method for deep neural network compression. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 5068–5076, 2017. DOI: 10.1109/ICCV.2017.541.

[118] S. H. Lin, R. R. Ji, Y. C. Li, C. Deng, X. L. Li. Toward compact convnets via structure-sparsity regularized filter pruning. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 2, pp. 574–588, 2020. DOI: 10.1109/TNNLS.2019.2906563.

[119] B. Y. Liu, M. Wang, H. Foroosh, M. Tappen, M. Penksy. Sparse convolutional neural networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Boston, USA, pp. 806–814, 2015. DOI: 10.1109/CVPR.2015.7298681.

[120] W. Wen, C. Xu, C. P. Wu, Y. D. Wang, Y. R. Chen, H. Li. Coordinating filters for faster deep neural networks. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 658–666, 2017. DOI: 10.1109/ICCV.2017.78.

[121] S. Han, H. Z. Mao, W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. [Online], Available: https://arxiv.org/abs/1510.00149, 2015.

[122] Y. Choi, M. El-Khamy, J. Lee. Compression of deep convolutional neural networks under joint sparsity constraints. [Online], Available: https://arxiv.org/abs/1805.08303, 2018.

[123] B. W. Pan, W. W. Lin, X. L. Fang, C. Q. Huang, B. L. Zhou, C. W. Lu. Recurrent residual module for fast inference in videos. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 1536–1545, 2018. DOI: 10.1109/CVPR.2018.00166.

[124] S. Han, X. Y. Liu, H. Z. Mao, J. Pu, A. Pedram, M. A. Horowitz, W. J. Dally. EIE: Efficient inference engine on compressed deep neural network. In *Proceedings of the 43rd ACM/IEEE Annual International Symposium on Computer Architecture*, IEEE, Seoul, Korea, pp. 243–254, 2016. DOI: 10.1109/ISCA.2016.30.

[125] K. Chen, J. Q. Wang, S. Yang, X. C. Zhang, Y. J. Xiong, C. C. Loy, D. H. Lin. Optimizing video object detection via a scale-time lattice. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 7814–7823, 2018. DOI: 10.1109/CVPR.2018.00815.

[126] S. Lee, S. Chang, N. Kwak. UrnEt: User-resizable residual networks with conditional gating module. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, New York, USA, pp. 4569–4576, 2020. DOI: 10.1609/aaai.v34i04.5886.

[127] B. Y. Fang, X. Zeng, M. Zhang. NestDNN: Resource-aware multi-tenant on-device deep learning for continuous mobile vision. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, ACM, New Delhi, India, pp. 115–127, 2018. DOI: 10.1145/3241539.3241559.

[128] N. Shazeer, K. Fatahalian, W. R. Mark, R. T. Mullapudi. Hydranets: Specialized dynamic architectures for efficient inference. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 8080–8089, 2018. DOI: 10.1109/CVPR.2018.00843.

[129] G. Huang, D. L. Chen, T. H. Li, F. Wu, L. van der Maaten, K. Q. Weinberger. Multi-scale dense networks for resource efficient image classification. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[130] Q. S. Guo, Z. P. Yu, Y. C. Wu, D. Liang, H. Y. Qin, J. J. Yan. Dynamic recursive neural network. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 5142–5151, 2019. DOI: 10.1109/CVPR.2019.00529.

[131] G. Huang, S. C. Liu, L. van der Maaten, K. Q. Weinberger. CondenseNet: An efficient DenseNet using learned group convolutions. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 2752–2761, 2018. DOI: 10.1109/CVPR.2018.00291.

[132] B. Yang, G. Bender, Q. V. Le, J. Ngiam. CondConv: Conditionally parameterized convolutions for efficient inference. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Vancouver, Canada, pp. 1307–1318, 2019. DOI: 10.5555/3454287.3454404.

[133] Y. P. Chen, X. Y. Dai, M. C. Liu, D. D. Chen, L. Yuan, Z. C. Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Seattle, USA, pp. 11027–11036, 2020. DOI: 10.1109/CVPR42600.2020.01104.

[134] A. W. Harley, K. G. Derpanis, I. Kokkinos. Segmentation-aware convolutional networks using local attention masks. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 5048–5057, 2017. DOI: 10.1109/ICCV.2017.539.

[135] H. Su, V. Jampani, D. Q. Sun, O. Gallo, E. Learned-Miller, J. Kautz. Pixel-adaptive convolutional neural networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 11158–11167, 2019. DOI: 10.1109/CVPR.2019.01142.

[136] K. Roy, A. Jaiswal, P. Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, vol. 575, no. 7784, pp. 607–617, 2019. DOI: 10.1038/s41586-019-1677-2.

[137] M. Ehrlich, L. Davis. Deep residual learning in the JPEG transform domain. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, IEEE, Seoul, Korea, pp. 3483–3492, 2019. DOI: 10.1109/ICCV.2019.00358.

[138] Z. H. Liu, T. Liu, W. J. Wen, L. Jiang, J. Xu, Y. Z. Wang, G. Quan. DeepN-JPEG: A deep neural network favorable jpeg-based image compression framework. In *Proceedings of the 55th ACM/ESDA/IEEE Design Automation Conference*, IEEE, San Francisco, USA, 2018, pp. 1–6, 2018. DOI: 10.1109/DAC.2018.8465809.

[139] M. Javed, P. Nagabhushan, B. B. Chaudhuri. A review on document image analysis techniques directly in the compressed domain. *Artificial Intelligence Review*, vol. 50, no. 4, pp. 539–568, 2018. DOI: 10.1007/s10462-017-9551-9.

[140] E. Oyallon, E. Belilovsky, S. Zagoruyko, M. Valko. Compressing the input for CNNs with the first-order scattering transform. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, 2018, pp. 305–320. DOI: 10.1007/978-3-030-01240-3_19.

[141] R. Torfason, F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, L. van Gool. Towards image understanding from deep compression without decoding. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[142] T. Chang, B. Tolooshams, D. Ba. RandNet: Deep learning with compressed measurements of images. In *Proceedings of the 29th IEEE International Workshop on Machine Learning for Signal Processing*, IEEE, Pittsburgh, USA, pp. 1–6, 2019. DOI: 10.1109/MLSP.2019.8918878.

[143] L. D. Chamain, Z. Ding. Faster and accurate classification for JPEG2000 compressed images in networked applications. [Online], Available: https://arxiv.org/abs/1909.05638, 2019.

[144] C. X. Ding, D. C. Tao. Trunk-branch ensemble convolutional neural networks for video-based face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 1002–1014, 2018. DOI: 10.1109/TPAMI.2017.2700390.

[145] L. Pigou, A. van den Oord, S. Dieleman, M. van Herreweghe, J. Dambre. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *International Journal of Computer Vision*, vol. 126, no. 2–4, pp. 430–439, 2018. DOI: 10.1007/s11263-016-0957-7.

[146] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, S. W. Baik. Action recognition in video sequences using deep bi-directional LSTM with CNN features. *IEEE Access*, vol. 6, pp. 1155–1166, 2018. DOI: 10.1109/ACCESS.2017.2778011.

[147] S. Tulyakov, M. Y. Liu, X. D. Yang, J. Kautz. MoCoGAN: Decomposing motion and content for video generation. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 1526–1535, 2018. DOI: 10.1109/CVPR.2018.00165.

[148] S. Y. Sun, Z. H. Kuang, L. Sheng, W. L. Ouyang, W. Zhang. Optical flow guided feature: A fast and robust motion representation for video action recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 1390–1399, 2018. DOI: 10.1109/CVPR.2018.00151.

[149] G. Lu, W. L. Ouyang, D. Xu, X. Y. Zhang, C. L. Cai, Z. Y. Gao. DVC: An end-to-end deep video compression framework. *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 10998–11007, 2019. DOI: 10.1109/CVPR.2019.01126.

[150] A. Habibian, T. van Rozendaal, J. Tomczak, T. Cohen. Video compression with rate-distortion autoencoders. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, IEEE, Seoul, Korea, pp. 7032–7041, 2020. DOI: 10.1109/ICCV.2019.00713.

[151] M. Quach, G. Valenzise, F. Dufaux. Learning convolutional transforms for lossy point cloud geometry compression. In *Proceedings of IEEE International Conference on Image Processing*, IEEE, Taipei, China, pp. 4320–4324, 2019. DOI: 10.1109/ICIP.2019.8803413.

[152] C. Moenning, N. A. Dodgson. Fast marching farthest point sampling. In *Proceedings of the 24th Annual Conference of the European Association for Computer Graphics*, Eurographics Association, Granada, Spain, pp. 39–42, 2003. DOI: 10.2312/egp.20031024.

[153] O. Dovrat, I. Lang, S. Avidan. Learning to sample. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 2755–2764, 2019. DOI: 10.1109/CVPR.2019.00287.

[154] R. Q. Charles, H. Su, M. Kaichun, L. J. Guibas. Point-

Net: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 77–85, 2017. DOI: 10.1109/CVPR.2017.16.

[155] Y. Zhao, Y. J. Xiong, D. H. Lin. Trajectory convolution for action recognition. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 2208–2219, 2018. DOI: 10.5555/3327144.3327148.

[156] S. Mukherjee, L. Anvitha, T. M. Lahari. Human activity recognition in RGB-D videos by dynamic images. *Multimedia Tools and Applications*, vol. 79, no. 27, pp. 19797–19801, 2020. 10.1007/s11042-020-08747-3.

[157] Y. Xiao, J. Chen, Y. C. Wang, Z. G. Cao, J. T. Zhou, X. Bai. Action recognition for depth video using multi-view dynamic images. *Information Sciences*, vol. 480, pp. 287–304, 2019. DOI: 10.1016/j.ins.2018.12.050.

[158] H. Liu, J. H. Tu, M. Y. Liu. Two-stream 3D convolutional neural network for skeleton-based action recognition. [Online], Available: https://arxiv.org/abs/1705.08106, 2017.

[159] D. Maturana, S. Scherer. VoxNet: A 3D convolutional neural network for real-time object recognition. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Hamburg, Germany, pp. 922–928, 2015. DOI: 10.1109/IROS.2015.7353481.

[160] J. Y. Chang, G. Moon, K. M. Lee. V2V-PoseNet: Voxel-to-voxel prediction network for accurate 3D hand and human pose estimation from a single depth map. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 5079–5088, 2018. DOI: 10.1109/CVPR.2018.00533.

[161] Q. Y. Wang, Y. X. Zhang, J. S. Yuan, Y. L. Lu. Space-time event clouds for gesture recognition: From RGB cameras to event cameras. In *Proceedings of IEEE Winter Conference on Applications of Computer Vision*, IEEE, Waikoloa, USA, pp. 1826–1835, 2019. DOI: 10.1109/WACV.2019.00199.

[162] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, N. de Freitas. Predicting parameters in deep learning. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, Lake Tahoe, USA, pp. 2148–2156, 2013. DOI: 10.5555/2999792.2999852.

[163] D. H. Wang, B. J. Wu, G. S. Zhao, M. Yao, H. N. Chen, L. Deng, T. Y. Yan, G. Q. Li. Kronecker CP decomposition with fast multiplication for compressing RNNs. *IEEE Transactions on Neural Networks and Learning Systems*, to be published. DOI: 10.1109/TNNLS.2021.3105961.

[164] L. Deng, Y. J. Wu, Y. F. Hu, L. Liang, G. Q. Li, X. Hu, Y. F. Ding, P. Li, Y. Xie. Comprehensive SNN compression using ADMM optimization and activity regularization. *IEEE Transactions on Neural Networks and Learning Systems*, to be published. DOI: 10.1109/TNNLS.2021.3109064.

[165] A. G. Howard, M. L. Zhu, B. Chen, D. Kalenichenko, W. J. Wang, T. Weyand, M. Andreetto, H. Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. [Online], Available: https://arxiv.org/abs/1704.04861, 2017.

[166] B. C. Wu, A. Wan, X. Y. Yue, P. Jin, S. C. Zhao, N. Golmant, A. Gholaminejad, J. Gonzalez, K. Keutzer. Shift: A zero FLOP, zero parameter alternative to spatial convolutions. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 9127–9135, 2018. DOI: 10.1109/CVPR.2018.00951.

[167] W. J. Luo, Y. J. Li, R. Urtasun, R. Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Barcelona, Spain, pp. 4905–4913, 2016. DOI: 10.5555/3157382.3157645.

[168] K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, USA, 2015.

[169] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello. ENet: A deep neural network architecture for real-time semantic segmentation. [Online], Available: https://arxiv.org/abs/1606.02147, 2016.

[170] M. Holschneider, R. Kronland-Martinet, J. Morlet, P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets: Time-Frequency Methods and Phase Space*, J. M. Combes, A. Grossmann, P. Tchamitchian, Eds. Berlin, Germany: Springer, pp. 286–297, 1989. DOI: 10.1007/978-3-642-97177-8_28.

[171] F. Yu, V. Koltun. Multi-scale context aggregation by dilated convolutions. In *Proceedings of the 4th International Conference on Learning Representations*, San Juan, Puerto Rico, 2016.

[172] J. F. Dai, H. Z. Qi, Y. W. Xiong, Y. Li, G. D. Zhang, H. Hu, Y. C. Wei. Deformable convolutional networks. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 764–773, 2017. DOI: 10.1109/ICCV.2017.89.

[173] M. Lin, Q. Chen, S. C. Yan. Network in network. [Online], Available: https://arxiv.org/abs/1312.4400, 2013.

[174] C. Szegedy, Wei Liu, Y. Q. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich. Going deeper with convolutions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Boston, USA, pp. 1–9, 2015. DOI: 10.1109/CVPR.2015.7298594.

[175] K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, USA, pp. 770–778, 2016. DOI: 10.1109/CVPR.2016.90.

[176] K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun. Identity mappings in deep residual networks. In *Proceedings of the 14th European Conference on Computer Vision*, Springer, Amsterdam, The Netherlands, pp. 630–645, 2016. DOI: 10.1007/978-3-319-46493-0_38.

[177] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 1800–1807, 2017. DOI: 10.1109/CVPR.2017.195.

[178] M. Sandler, A. Howard, M. L. Zhu, A. Zhmoginov, L. C. Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of IEEE/CVF Conference on Com-*

puter Vision and Pattern Recognition, IEEE, Salt Lake City, USA, pp. 4510–4520, 2018. DOI: 10.1109/CVPR.2018.00474.

[179] S. Chen, Y. Liu, X. Gao, Z. Han. Mobilefacenets: Efficient CNNs for accurate real-time face verification on mobile devices. In *Proceedings of the 13th Chinese Conference on Biometric Recognition*, Springer, Urumqi, China, pp. 428–438, 2018. DOI: 10.1007/978-3-319-97909-0_46.

[180] S. N. Xie, R. Girshick, P. Dollár, Z. W. Tu, K. M. He. Aggregated residual transformations for deep neural networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 5987–5995, 2017. DOI: 10.1109/CVPR.2017.634.

[181] S. Hochreiter J. Schmidhuber. Long short-term memory. *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[182] F. A. Gers, J. Schmidhuber, F. Cummins. Learning to forget: Continual prediction with LSTM. In *Proceedings of the 19th International Conference on Artificial Neural Networks*, Edinburgh, UK, pp. 850–855, 1999. DOI: 10.1049/cp:19991218.

[183] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *Proceedings of Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, pp. 1724–1734, 2014. DOI: 10.3115/v1/D14-1179.

[184] G. B. Zhou, J. X. Wu, C. L. Zhang, Z. H. Zhou. Minimal gated unit for recurrent neural networks. *International Journal of Automation and Computing*, vol. 13, no. 3, pp. 226–234, 2016. DOI: 10.1007/s11633-016-1006-2.

[185] A. Kusupati, M. Singh, K. Bhatia, A. Kumar, P. Jain, M. Varma. FastGRNN: A fast, accurate, stable and tiny kilobyte sized gated recurrent neural network. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 9031–9042, 2018. DOI: 10.5555/3327546.3327577.

[186] J. Bradbury, S. Merity, C. M. Xiong, R. Socher. Quasi-recurrent neural networks. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[187] S. Z. Zhang, Y. H. Wu, T. Che, Z. H. Lin, R. Memisevic, R. Salakhutdinov, Y. Bengio. Architectural complexity measures of recurrent neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Barcelona, Spain, pp. 1822–1830, 2016. DOI: 10.5555/3157096.3157301.

[188] N. Kalchbrenner, I. Danihelka, A. Graves. Grid long short-term memory. [Online], Available: https://arxiv.org/abs/1507.01526, 2015.

[189] M. Fraccaro, S. K. Sønderby, U. Paquet, O. Winther. Sequential neural models with stochastic layers. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Barcelona, Spain, pp. 2207–2215, 2016. DOI: 10.5555/3157096.3157343.

[190] G. Hinton, O. Vinyals, J. Dean. Distilling the knowledge in a neural network. [Online], Available: https://arxiv.org/abs/1503.02531, 2015.

[191] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, J. Schmidhuber. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017. DOI: 10.1109/TNNLS.2016.2582924.

[192] B. Zoph, V. Vasudevan, J. Shlens, Q. V. Le. Learning transferable architectures for scalable image recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 8697–8710, 2018. DOI: 10.1109/CVPR.2018.00907.

[193] H. X. Liu, K. Simonyan, Y. M. Yang. Darts: Differentiable architecture search. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[194] A. Rawal, R. Miikkulainen. From nodes to networks: Evolving recurrent neural networks. [Online], Available: https://arxiv.org/abs/1803.04439, 2018.

[195] Z. Zhong, J. J. Yan, W. Wu, J. Shao, C. L. Liu. Practical block-wise neural network architecture generation. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 2423–2432, 2018. DOI: 10.1109/CVPR.2018.00257.

[196] C. X. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L. J. Li, L. Fei-Fei, A. Yuille, J. Huang, K. Murphy. Progressive neural architecture search. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 19–35, 2018. DOI: 10.1007/978-3-030-01246-5_2.

[197] H. X. Liu, K. Simonyan, O. Vinyals, C. Fernando, K. Kavukcuoglu. Hierarchical representations for efficient architecture search. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[198] B. Baker, O. Gupta, N. Naik, R. Raskar. Designing neural network architectures using reinforcement learning. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[199] Z. Zhong, J. J. Yan, W. Wu, J. Shao, C. L. Liu. Practical block-wise neural network architecture generation. [Online], Available: https://arxiv.org/abs/1708.05552, 2017.

[200] H. Cai, J. C. Yang, W. N. Zhang, S. Han, Y. Yu. Path-level network transformation for efficient architecture search. In *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, pp. 678–687, 2018.

[201] H. Cai, T. Y. Chen, W. N. Zhang, Y. Yu, J. Wang. Efficient architecture search by network transformation. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, New Orleans, USA, pp. 2787–2794, 2018. DOI: 10.5555/3504035.3504375.

[202] L. X. Xie, A. L. Yuille. Genetic CNN. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 1388–1397, 2017. DOI: 10.1109/ICCV.2017.154.

[203] A. Klein, E. Christiansen, K. Murphy, F. Hutter. Towards reproducible neural architecture and hyperparameter search. In *Proceedings of the 2nd Reproducibility in Machine Learning Workshop*, Stockholm, Sweden, 2018.

[204] M. X. Tan, Q. V. Le. EfficientNet: Rethinking model scal-

ing for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, USA, pp. 6105–6114, 2019.

[205] T. Elsken, J. Metzen, F. Hutter. Efficient multi-objective neural architecture search via Lamarckian evolution. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[206] A. Klein, S. Falkner, S. Bartels, P. Hennig, F. Hutter. Fast Bayesian optimization of machine learning hyperparameters on large datasets. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, USA, pp. 528–536, 2017.

[207] H. Cai, C. Gan, T. Z. Wang, Z. K. Zhang, S. Han. Once-for-all: Train one network and specialize it for efficient deployment. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

[208] A. Klein, S. Falkner, J. T. Springenberg, F. Hutter. Learning curve prediction with Bayesian neural networks. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[209] T. Wei, C. H. Wang, Y. Rui, C. W. Chen. Network morphism. In *Proceedings of the 33rd International Conference on Machine Learning*, New York, USA, pp. 564–572, 2016.

[210] M. Masana, J. van de Weijer, L. Herranz, A. D. Bagdanov, J. M. Álvarez. Domain-adaptive deep network compression. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 4299–4307, 2017. DOI: 10.1109/ICCV.2017.460.

[211] T. Kumamoto, M. Suzuki, H. Matsueda. Singular-value-decomposition analysis of associative memory in a neural network. *Journal of the Physical Society of Japan*, vol. 86, no. 2, Article number 24005, 2017. DOI: 10.7566/JPSJ.86.024005.

[212] T. Deb, A. K. Ghosh, A. Mukherjee. Singular value decomposition applied to associative memory of Hopfield neural network. *Materials Today*: *Proceedings*, vol. 5, no. 1, pp. 2222–2228, 2018. DOI: 10.1016/j.matpr.2017.09.222.

[213] Z. X. Zou, Z. W. Shi. Ship detection in spaceborne optical image with SVD networks. *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 5832–5845, 2016. DOI: 10.1109/TGRS.2016.2572736.

[214] X. Y. Zhang, J. H. Zou, X. Ming, K. M. He, J. Sun. Efficient and accurate approximations of nonlinear convolutional networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Boston, USA, pp. 1984–1992, 2015. DOI: 10.1109/CVPR.2015.7298809.

[215] X. Y. Zhang, J. H. Zou, K. M. He, J. Sun. Accelerating very deep convolutional networks for classification and detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 1943–1955, 2016. DOI: 10.1109/TPAMI.2015.2502579.

[216] Y. Ioannou, D. Robertson, R. Cipolla, A. Criminisi. Deep roots: Improving CNN efficiency with hierarchical filter groups. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 5977–5986, 2017. DOI: 10.1109/CVPR.2017.633.

[217] B. Peng, W. M. Tan, Z. Y. Li, S. Zhang, D. Xie, S. L. Pu. Extreme network compression via filter group approximation. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 307–323, 2018. DOI: 10.1007/978-3-030-01237-3_19.

[218] G. S. Hu, Y. Hua, Y. Yuan, Z. H. Zhang, Z. Lu, S. S. Mukherjee, T. M. Hospedales, N. M. Robertson, Y. X. Yang. Attribute-enhanced face recognition with neural tensor fusion networks. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 3764–3773, 2017. DOI: 10.1109/ICCV.2017.404.

[219] J. D. Carroll, J. J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970. DOI: 10.1007/BF02310791.

[220] L. De Lathauwer, B. De Moor, J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000. DOI: 10.1137/S0895479896305696.

[221] L. De Lathauwer, B. De Moor, J. Vandewalle. On the best rank-1 and rank-(R_1, R_2, $\cdots$, R_N) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1324–1342, 2000. DOI: 10.1137/S0895479898346995.

[222] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966. DOI: 10.1007/BF02289464.

[223] T. G. Kolda, B. W. Bader. Tensor decompositions and applications. *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009. DOI: 10.1137/07070111X.

[224] G. Huang, Z. Liu, L. van der Maaten, K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 2261–2269, 2017. DOI: 10.1109/CVPR.2017.243.

[225] X. C. Zhang, Z. Z. Li, C. C. Loy, D. H. Lin. PolyNet: A pursuit of structural diversity in very deep networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 3900–3908, 2017. DOI: 10.1109/CVPR.2017.415.

[226] Y. P. Chen, J. N. Li, H. X. Xiao, X. J. Jin, S. C. Yan, J. S. Feng. Dual path networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, USA, pp. 4470–4478, 2017. DOI: 10.5555/3294996.3295200.

[227] Y. D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, D. Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. In *Proceedings of the 4th International Conference on Learning Representations*, San Juan, Puerto Rico, 2016.

[228] J. Kossaifi, A. Khanna, Z. Lipton, T. Furlanello, A. Anandkumar. Tensor contraction layers for parsimonious deep nets. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, Honolulu, USA, pp. 1940–1946, 2017. DOI: 10.1109/CVPRW.2017.243.

[229] J. Kossaifi, Z. C. Lipton, A. Kolbeinsson, A. Khanna, T. Furlanello, A. Anandkumar. Tensor regression networks. *Journal of Machine Learning Research*, vol. 21, no. 123, pp. 1–21, 2020.

[230] M. Janzamin, H. Sedghi, A. Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. [Online], Available: ht-

tps://arxiv.org/abs/1506.08473, 2016.

[231] V. Lebedev, Y. Ganin, M. Rakhuba, I. V. Oseledets, V. S. Lempitsky. Speeding-up convolutional neural networks using fine-tuned CP-decomposition. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, USA, 2015.

[232] D. T. Tran, A. Iosifidis, M. Gabbouj. Improving efficiency in convolutional neural networks with multilinear filters. *Neural Networks*, vol. 105, pp. 328–339, 2018. DOI: 10.1016/j.neunet.2018.05.017.

[233] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, A. Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature Communications*, vol. 8, no. 1, pp. 1–8, 2017. DOI: 10.1038/ncomms13890.

[234] M. Y. Zhou, Y. P. Liu, Z. Long, L. X. Chen, C. Zhu. Tensor rank learning in CP decomposition via convolutional neural network. *Signal Processing*: *Image Communication*, vol. 73, pp. 12–21, 2019. DOI: 10.1016/j.image.2018.03.017.

[235] S. Oymak, M. Soltanolkotabi. End-to-end learning of a convolutional neural network via deep tensor decomposition. [Online], Available: https//arxiv.org/abs/1805.06523, 2018.

[236] L. Grasedyck, D. Kressner, C. Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen*, vol. 36, no. 1, pp. 53–78, 2013. DOI: 10.1002/gamm.201310004.

[237] A. Cichocki, D. Mandic, L. De Lathauwer, G. X. Zhou, Q. B. Zhao, C. Caiafa, H. A. Phan. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015. DOI: 10.1109/MSP.2013.2297439.

[238] L. De Lathauwer. Decompositions of a higher-order tensor in block terms – Part Ⅱ: Definitions and uniqueness. *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1033–1066, 2008. DOI: 10.1137/070690729.

[239] A. H. Phan, A. Cichocki, P. Tichavský, R. Zdunek, S. Lehky. From basis components to complex structural patterns. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, Vancouver, Canada, pp. 3228–3232, 2013. DOI: 10.1109/ICASSP.2013.6638254.

[240] A. H. Phan, A. Cichocki, I. Oseledets, G. G. Calvi, S. Ahmadi-Asl, D. P. Mandic. Tensor networks for latent variable analysis: Higher order canonical polyadic decomposition. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 6, pp. 2174–2188, 2020. DOI: 10.1109/TNNLS.2019.2929063.

[241] W. H. He, Y. J. Wu, L. Deng, G. Q. Li, H. Y. Wang, Y. Tian, W. Ding, W. H. Wang, Y. Xie. Comparing SNNs and RNNs on neuromorphic vision datasets: Similarities and differences. *Neural Networks*, vol. 132, pp. 108–120, 2020. DOI: 10.1016/j.neunet.2020.08.001.

[242] L. Deng, Y. J. Wu, X. Hu, L. Liang, Y. F. Ding, G. Q. Li, G. S. Zhao, P. Li, Y. Xie. Rethinking the performance comparison between SNNs and ANNs. *Neural Networks*, vol. 121, pp. 294–307, 2020. DOI: 10.1016/j.neunet.2019.09.005.

[243] A. Cichocki. Tensor networks for dimensionality reduction, big data and deep learning. In *Advances in Data Analysis with Computational Intelligence Methods*, A. E. Gawęda, J. Kacprzyk, L. Rutkowski, G. G. Yen, Eds., Cham, Germany: Springer, pp. 3–49, 2018. DOI: 10.1007/978-3-319-67946-4_1.

[244] A. Pellionisz, R. Llinás. Tensor network theory of the metaorganization of functional geometries in the central nervous system. *Neuroscience*, vol. 16, no. 2, pp. 245–273, 1985. DOI: 10.1016/0306-4522(85)90001-6.

[245] I. V. Oseledets, E. E. Tyrtyshnikov. Breaking the curse of dimensionality, or how to use SVD in many dimensions. *SIAM Journal on Scientific Computing*, vol. 31, no. 5, pp. 3744–3759, 2009. DOI: 10.1137/090748330.

[246] I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011. DOI: 10.1137/090752286.

[247] B. N. Khoromskij. $O(d \log N)$-quantics approximation of N-d tensors in high-dimensional numerical modeling. *Constructive Approximation*, vol. 34, no. 2, pp. 257–280, 2011. DOI: 10.1007/s00365-011-9131-1.

[248] M. Espig, K. K. Naraparaju, J. Schneider. A note on tensor chain approximation. *Computing and Visualization in Science*, vol. 15, no. 6, pp. 331–344, 2012. DOI: 10.1007/s00791-014-0218-7.

[249] Q. B. Zhao, M. Sugiyama, L. H. Yuan, A. Cichocki. Learning efficient tensor representations with ring-structured networks. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, Brighton, UK, pp. 8608–8612, 2018. DOI: 10.1109/ICASSP.2019.8682231.

[250] Q. B. Zhao, G. X. Zhou, S. L. Xie, L. Q. Zhang, A. Cichocki. Tensor ring decomposition. [Online], Available: https://arxiv.org/abs/1606.05535, 2016.

[251] W. Hackbusch, S. Kühn. A new scheme for the tensor representation. *Journal of Fourier Analysis and Applications*, vol. 15, no. 5, pp. 706–722, 2009. DOI: 10.1007/s00041-009-9094-9.

[252] L. Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 4, pp. 2029–2054, 2010. DOI: 10.1137/090764189.

[253] N. Lee, A. Cichocki. Regularized computation of approximate pseudoinverse of large matrices using low-rank tensor train decompositions. *SIAM Journal on Matrix Analysis and Applications*, vol. 37, no. 2, pp. 598–623, 2016. DOI: 10.1137/15M1028479.

[254] N. Lee, A. Cichocki. Fundamental tensor operations for large-scale data analysis using tensor network formats.. *Multidimensional Systems and Signal Processing*, vol. 29, no. 3, pp. 921–960, 2018. DOI: 10.1007/s11045-017-0481-0.

[255] N. Cohen, O. Sharir, A. Shashua. On the expressive power of deep learning: A tensor analysis. In *Proceedings of the 29th Annual Conference on Learning Theory*, New York, USA, pp. 698–728, 2016.

[256] M. Zhu, S. Gupta. To prune, or not to prune: Exploring the efficacy of pruning for model compression. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[257] H. T. Huang, L. B. Ni, K. W. Wang, Y. G. Wang, H. Yu. A highly parallel and energy efficient three-dimensional multilayer CMOS-RRAM accelerator for tensorized neur-

al network. *IEEE Transactions on Nanotechnology*, vol. 17, no. 4, pp. 645–656, 2018. DOI: 10.1109/TNANO.2017.2732698.

[258] J. H. Su, J. L. Li, B. Bhattacharjee, F. R. Huang. Tensorial neural networks: Generalization of neural networks and application to model compression. [Online], Available: https://arxiv.org/abs/1805.10352, 2018.

[259] D. H. Wang, G. S. Zhao, H. N. Chen, Z. X. Liu, L. Deng, G. Q. Li. Nonlinear tensor train format for deep neural network compression. *Neural Networks*, vol. 144, pp. 320–333, 2021. DOI: 10.1016/j.neunet.2021.08.028.

[260] J. Achterhold, J. M. Köhler, A. Schmeink, T. Genewein. Variational network quantization. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[261] C. Leng, H. Li, S. H. Zhu, R. Jin. Extremely low bit neural network: Squeeze the last bit out with ADMM. [Online], Available: https//arxiv.org/abs/1707.09870, 2017.

[262] A. J. Zhou, A. B. Yao, Y. W. Guo, L. Xu, Y. R. Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[263] S. Jung, C. Son, S. Lee, J. Son, J. J. Han, Y. Kwak, S. J. Hwang, C. Choi. Learning to quantize deep networks by optimizing quantization intervals with task loss. *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 345–4354, 2019. DOI: 10.1109/CVPR.2019.00448.

[264] S. C. Zhou, Y. Z. Wang, H. Wen, Q. Y. He, Y. H. Zou. Balanced quantization: An effective and efficient approach to quantized neural networks. *Journal of Computer Science and Technology*, vol. 32, no. 4, pp. 667–682, 2017. DOI: 10.1007/s11390-017-1750-y.

[265] Y. Choi, M. El-Khamy, J. Lee. Learning sparse low-precision neural networks with learnable regularization. [Online], Available: https//arxiv.org/abs/1809.00095, 2018.

[266] K. Wang, Z. J. Liu, Y. J. Lin, J. Lin, S. Han. HAQ: Hardware-aware automated quantization with mixed precision. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 8604–8612, 2019. DOI: 10.1109/CVPR.2019.00881.

[267] L. Deng, P. Jiao, J. Pei, Z. Z. Wu, G. Q. Li. GXNOR-Net: Training deep neural networks with ternary weights and activations without full-precision memory under a unified discretization framework. *Neural Networks*, vol. 100, pp. 49–58, 2018. DOI: 10.1016/j.neunet.2018.01.010.

[268] R. Banner, I. Hubara, E. Hoffer, D. Soudry. Scalable methods for 8-bit training of neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 5151–5159, 2018. DOI: 10.5555/3327345.3327421.

[269] C. Sakr, N. R. Shanbhag. Per-tensor fixed-point quantization of the back-propagation algorithm. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[270] N. G. Wang, J. Choi, D. Brand, C. Y. Chen, K. Gopalakrishnan. Training deep neural networks with 8-bit floating point numbers. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 7686–7695, 2018. DOI: 10.5555/3327757.3327866.

[271] R. Zhao, Y. W. Hu, J. Dotzel, C. De Sa, Z. R. Zhang. Improving neural network quantization without retraining using outlier channel splitting. In *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, USA, pp. 7543–7552, 2019.

[272] Z. C. Liu, Z. Q. Shen, M. Savvides, K. T. Cheng. ReActNet: Towards precise binary neural network with generalized activation functions. In *Proceedings of the 16th European Conference on Computer Vision*, Springer, Glasgow, UK, pp. 143–159, 2020. DOI: 10.1007/978-3-030-58568-6_9.

[273] G. Tej Pratap, R. Kumar, N. S. Pradeep. Hybrid and non-uniform quantization methods using retro synthesis data for efficient inference. In *Proceedings of International Joint Conference on Neural Networks*, IEEE, Shenzhen, China, 2021. DOI: 10.1109/IJCNN52387.2021.9533724.

[274] C. Gong, Y. Chen, Y. Lu, T. Li, C. Hao, D. M. Chen. VecQ: Minimal loss DNN model compression with vectorized weight quantization. *IEEE Transactions on Computers*, vol. 70, no. 5, pp. 696–710, 2021. DOI: 10.1109/TC.2020.2995593.

[275] C. Z. Zhu, S. Han, H. Z. Mao, W. J. Dally. Trained ternary quantization. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[276] R. P. K. Poudel, U. Bonde, S. Liwicki, C. Zach. ContextNet: Exploring context and detail for semantic segmentation in real-time. [Online], Available: https://arxiv.org/abs/1805.04554, 2018.

[277] R. P. K. Poudel, S. Liwicki, R. Cipolla. Fast-SCNN: Fast semantic segmentation network. In *Proceedings of the 30th British Machine Vision Conference*, Cardiff, UK, 2019.

[278] M. Courbariaux, Y. Bengio, J. P. David. BinaryConnect: Training deep neural networks with binary weights during propagations. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 3123–3131, 2015.

[279] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, Y. Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6869–6898, 2017. DOI: 10.5555/3122009.3242044.

[280] S. C. Zhou, Y. X. Wu, Z. K. Ni, X. Y. Zhou, H. Wen, Y. H. Zou. DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients. [Online], Available: https://arxiv.org/abs/1606.06160, 2016.

[281] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, Y. Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or −1. [Online], Available: https://arxiv.org/abs/1602.02830, 2016.

[282] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, J. Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ACM, Montréal, Canada, pp. 1113–1120, 2009. DOI: 10.1145/1553374.1553516.

[283] W. L. Chen, J. T. Wilson, S. Tyree, K. Q. Weinberger, Y. X. Chen. Compressing neural networks with the hashing trick. In *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, pp. 2285–2294, 2015.

[284] R. Spring, A. Shrivastava. Scalable and sustainable deep learning via randomized hashing. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Halifax, Canada, pp. 445–454, 2017. DOI: 10.1145/3097983.3098035.

[285] Y. J. Lin, S. Han, H. Z. Mao, Y. Wang, B. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[286] T. W. Chin, C. Zhang, D. Marculescu. Layer-compensated pruning for resource-constrained convolutional neural networks. [Online], Available: https://arxiv.org/abs/1810.00518, 2018.

[287] Y. H. He, J. Lin, Z. J. Liu, H. R. Wang, L. J. Li, S. Han. AMC: AutoML for model compression and acceleration on mobile devices. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 815–832, 2018. DOI: 10.1007/978-3-030-01234-2_48.

[288] X. F. Xu, M. S. Park, C. Brick. Hybrid pruning: Thinner sparse networks for fast inference on edge devices. [Online], Available: https://arxiv.org/abs/1811.00482, 2018.

[289] J. B. Ye, X. Lu, Z. Lin, J. Z. Wang. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[290] J. H. Luo, J. X. Wu. AutoPruner: An end-to-end trainable filter pruning method for efficient deep model inference. *Pattern Recognition*, vol. 107, Article number 107461, 2020. DOI: 10.1016/j.patcog.2020.107461.

[291] X. L. Dai, H. X. Yin, N. K. Jha. NeST: A neural network synthesis tool based on a grow-and-prune paradigm. *IEEE Transactions on Computers*, vol. 68, no. 10, pp. 1487–1497, 2019. DOI: 10.1109/TC.2019.2914438.

[292] Z. Liu, J. G. Li, Z. Q. Shen, G. Huang, S. M. Yan, C. S. Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp. 2755–2763, 2017. DOI: 10.1109/ICCV.2017.298.

[293] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, J. Kautz. Importance estimation for neural network pruning. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 11256–11264, 2019. DOI: 10.1109/CVPR.2019.01152.

[294] A. Renda, J. Frankle, M. Carbin. Comparing rewinding and fine-tuning in neural network pruning. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

[295] G. G. Ding, S. Zhang, Z. Z. Jia, J. Zhong, J. G. Han. Where to prune: Using LSTM to guide data-dependent soft pruning. *IEEE Transactions on Image Processing*, vol. 30, pp. 293–304, 2021. DOI: 10.1109/TIP.2020.3035028.

[296] M. B. Lin, L. J. Cao, S. J. Li, Q. X. Ye, Y. H. Tian, J. Z. Liu, Q. Tian, R. R. Ji. Filter sketch for network pruning. *IEEE Transactions on Neural Networks and Learning Systems*, to be published. DOI: 10.1109/TNNLS.2021.3084206.

[297] M. B. Lin, R. R. Ji, S. J. Li, Y. Wang, Y. J. Wu, F. Y. Huang, Q. X. Ye. Network pruning using adaptive exemplar filters. *IEEE Transactions on Neural Networks and Learning Systems*, to be published. DOI: 10.1109/TNNLS.2021.3084856.

[298] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, E. Shelhamer. cuDNN: Efficient primitives for deep learning. [Online], Available: https://arxiv.org/abs/1410.0759, 2014.

[299] X. L. Dai, H. X. Yin, N. K. Jha. Grow and prune compact, fast, and accurate LSTMs. *IEEE Transactions on Computers*, vol. 69, no. 3, pp. 441–452, 2020. DOI: 10.1109/TC.2019.2954495.

[300] M. H. Zhu, J. Clemons, J. Pool, M. Rhu, S. W. Keckler, Y. Xie. Structurally sparsified backward propagation for faster long short-term memory training. [Online], Available: https://arxiv.org/abs/1806.00512, 2018.

[301] F. Alibart, E. Zamanidoost, D. B. Strukov. Pattern classification by memristive crossbar circuits using *ex situ* and *in situ* training. *Nature Communications*, vol. 4, no. 1, Article number 2072, 2013. DOI: 10.1038/ncomms3072.

[302] Z. Liu, M. J. Sun, T. H. Zhou, G. Huang, T. Darrell. Rethinking the value of network pruning. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[303] J. Frankle, M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[304] N. Cohen, A. Shashua. Convolutional rectifier networks as generalized tensor decompositions. In *Proceedings of the 33rd International Conference on Machine Learning*, New York City, USA, pp. 955–963, 2016.

[305] Y. P. Chen, X. J. Jin, B. Y. Kang, J. S. Feng, S. C. Yan. Sharing residual units through collective tensor factorization in deep neural networks. [Online], Available: https://arxiv.org/abs/1703.02180v2, 2017.

[306] S. H. Li, L. Wang. Neural network renormalization group. *Physical Review Letters*, vol. 121, no. 26, Article number 260601, 2018. DOI: 10.1103/PhysRevLett.121.260601.

[307] G. Evenbly, G. Vidal. Algorithms for entanglement renormalization. *Physical Review B*, vol. 79, no. 14, Article number 144108, 2009. DOI: 10.1103/PhysRevB.79.144108.

[308] A. S. Morcos, H. N. Yu, M. Paganini, Y. D. Tian. One ticket to win them all: Generalizing lottery ticket initializations across datasets and optimizers. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Vancouver, Canada, Article number 444, 2019. DOI: 10.5555/3454287.3454731.

[309] H. N. Yu, S. Edunov, Y. D. Tian, A. S. Morcos. Playing the lottery with rewards and multiple languages: Lottery tickets in RL and NLP. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, pp. 1–12, 2020.

[310] E. Malach, G. Yehudai, S. Shalev-Shwartz, O. Shamir.

Proving the lottery ticket hypothesis: Pruning is all you need. In *Proceedings of the 37th International Conference on Machine Learning*, Vienna, Austria, pp. 6682–6691, 2020.

[311] L. Orseau, M. Hutter, O. Rivasplata. Logarithmic pruning is all you need. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Vancouver, Canada, Article number 246, 2020. DOI: 10.5555/3495724.3495970.

[312] S. K. Ye, T. Y. Zhang, K. Q. Zhang, J. Y. Li, K. D. Xu, Y. F. Yang, F. X. Yu, J. Tang, Fardad, S. J. Liu, X. Chen, X. Lin, Y. Z. Wang. Progressive weight pruning of deep neural networks using ADMM. [Online], Available: https://arxiv.org/abs/1810.07378, 2018.

[313] A. Polino, R. Pascanu, D. Alistarh. Model compression via distillation and quantization. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[314] P. Jiang, G. Agrawal. A linear speedup analysis of distributed deep learning with sparse and quantized communication. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montréal, Canada, pp. 2530–2541, 2018. DOI: 10.5555/3327144.3327178.

[315] G. Tzelepis, A. Asif, S. Baci, S. Cavdar, E. E. Aksoy. Deep neural network compression for image classification and object detection. In *Proceedings of the 18th IEEE International Conference on Machine Learning and Applications*, IEEE, Boca Raton, USA, pp. 1621–1628, 2019. DOI: 10.1109/ICMLA.2019.00266.

[316] D. Lee, D. H. Wang, Y. K. Yang, L. Deng, G. S. Zhao, G. Q. Li. QTTnet: Quantized tensor train neural networks for 3D object and video recognition. *Neural Networks*, vol. 144, pp. 420–432, 2021. DOI: 10.1016/j.neunet.2021.05.034.

[317] X. Z. Zhu, J. F. Dai, L. Yuan, Y. C. Wei. Towards high performance video object detection. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 7210–7218, 2018. DOI: 10.1109/CVPR.2018.00753.

[318] J. Lin, Y. M. Rao, J. W. Lu, J. Zhou. Runtime neural pruning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, USA, pp. 2178–2188, 2017. DOI: 10.5555/3294771.3294979.

[319] Y. M. Rao, J. W. Lu, J. Lin, J. Zhou. Runtime network routing for efficient image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 10, pp. 2291–2304, 2019. DOI: 10.1109/TPAMI.2018.2878258.

[320] X. T. Gao, Y. R. Zhao, L. Dudziak, R. D. Mullins, C. Z. Xu. Dynamic channel pruning: Feature boosting and suppression. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[321] J. H. Yu, L. J. Yang, N. Xu, J. C. Yang, T. S. Huang. Slimmable neural networks. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[322] Z. D. Zhang, C. Jung. Recurrent convolution for compact and cost-adjustable neural networks: An empirical study. [Online], Available: https://arxiv.org/abs/1902.09809, 2019.

[323] S. C. Liu, Y. Y. Lin, Z. M. Zhou, K. M. Nan, H. Liu, J. Z. Du. On-demand deep model compression for mobile devices: A usage-driven model selection framework. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, ACM, Munich, Germany, pp. 389–400, 2018. DOI: 10.1145/3210240.3210337.

[324] T. Bolukbasi, J. Wang, O. Dekel, V. Saligrama. Adaptive neural networks for efficient inference. In *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, pp. 527–536, 2017.

[325] X. Wang, F. Yu, Z. Y. Dou, T. Darrell, J. E. Gonzalez. SkipNet: Learning dynamic routing in convolutional networks. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 420–436, 2018. DOI: 10.1007/978-3-030-01261-8_25.

[326] A. Ehteshami Bejnordi, R. Krestel. Dynamic channel and layer gating in convolutional neural networks. In *Proceedings of the 43rd German Conference on Artificial Intelligence*, Springer, Bamberg, Germany, pp. 33–45, 2020. DOI: 10.1007/978-3-030-58285-2_3.

[327] J. Q. Guan, Y. Liu, Q. Liu, J. Peng. Energy-efficient amortized inference with cascaded deep classifiers. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI.org, Stockholm, Sweden, pp. 2184–2190, 2018. DOI: 10.24963/ijcai.2018/302.

[328] H. X. Li, Z. Lin, X. H. Shen, J. Brandt, G. Hua. A convolutional neural network cascade for face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Boston, USA, pp. 5325–5334, 2015. DOI: 10.1109/CVPR.2015.7299170.

[329] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991. DOI: 10.1162/neco.1991.3.1.79.

[330] A. Veit, S. Belongie. Convolutional networks with adaptive inference graphs. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 3–18, 2018. DOI: 10.1007/978-3-030-01246-5_1.

[331] H. Y. Wang, Z. Q. Qin, S. Y. Li, X. Li. CoDiNet: Path distribution modeling with consistency and diversity for dynamic routing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, to be published. DOI: 10.1109/TPAMI.2021.3084680.

[332] J. Hu, L. Shen, G. Sun. Squeeze-and-excitation networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 7132–7141, 2018. DOI: 10.1109/CVPR.2018.00745.

[333] F. Wang, M. Q. Jiang, C. Qian, S. Yang, C. Li, H. G. Zhang, X. G. Wang, X. O. Tang. Residual attention network for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp. 6450–6458, 2017. DOI: 10.1109/CVPR.2017.683.

[334] M. Y. Ren, A. Pokrovsky, B. Yang, R. Urtasun. SBNet: Sparse blocks network for fast inference. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 8711–8720, 2018. DOI: 10.1109/CVPR.2018.00908.

[335] A. Recasens, P. Kellnhofer, S. Stent, W. Matusik, A. Torralba. Learning to zoom: A saliency-based sampling layer for neural networks. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 52–67, 2018. DOI: 10.1007/978-3-030-01240-3_4.

[336] Z. R. Yang, Y. H. Xu, W. R. Dai, H. K. Xiong. Dynamic-stride-net: Deep convolutional neural network with dynamic stride. In *Proceedings of SPIE 11187, Optoelectronic Imaging and Multimedia Technology VI*, SPIE, Hangzhou, China, Article number 1118707, 2019. DOI: 10.1117/12.2537799.

[337] W. H. Wu, D. L. He, X. Tan, S. F. Chen, Y. Yang, S. L. Wen. Dynamic inference: A new approach toward efficient video action recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, Seattle, USA, pp. 2890–2898, 2020. DOI: 10.1109/CVPRW50498.2020.00346.

[338] B. L. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba. Learning deep features for discriminative localization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, USA, pp. 2921–2929, 2016. DOI: 10.1109/CVPR.2016.319.

[339] A. H. Phan, A. Cichocki, P. Tichavský, D. P. Mandic, K. Matsuoka. On revealing replicating structures in multiway data: A novel tensor decomposition approach. In *Proceedings of the 10th International Conference on Latent Variable Analysis and Signal Separation*, Springer, Tel Aviv, Israel, pp. 297–305, 2012. DOI: 10.1007/978-3-642-28551-6_37.

[340] J. Pei, L. Deng, S. Song, M. G. Zhao, Y. H. Zhang, S. Wu, G. R. Wang, Z. Zou, Z. H. Wu, W. He, F. Chen, N. Deng, S. Wu, Y. Wang, Y. J. Wu, Z. Y. Yang, C. Ma, G. Q. Li, W. T. Han, H. L. Li, H. Q. Wu, R. Zhao, Y. Xie, L. P. Shi. Towards artificial general intelligence with hybrid Tianjic chip architecture. *Nature*, vol. 572, no. 7767, pp. 106–111, 2019. DOI: 10.1038/s41586-019-1424-8.

[341] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, D. S. Modha. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, vol. 345, no. 6197, pp. 668–673, 2014. DOI: 10.1126/science.1254642.

[342] N. Schuch, I. Cirac, D. Pérez-García. Peps as ground states: Degeneracy and topology. *Annals of Physics*, vol. 325, no. 10, pp. 2153–2192, 2010. DOI: 10.1016/j.aop.2010.05.008.

[343] A. Hallam, E. Grant, V. Stojevic, S. Severini, A. G. Green. Compact neural networks based on the multiscale entanglement renormalization ansatz. In *Proceedings of British Machine Vision Conference*, Newcastle, UK, 2018.

Yang Wu received the B. Sc. degree in information and the Ph. D. degree in control science and engineering from Xi'an Jiaotong University, China in 2004 and 2010, respectively. He is currently a principal researcher with Applied Research Center (ARC) Laboratory, Tencent Platform and Content Group (PCG), China. From July 2019 to May 2021, he was a program-specific senior lecturer with Department of Intelligence Science and Technology, Kyoto University, Japan. He was an assistant professor of the Nara Institute of Science and Technology (NAIST) International Collaborative Laboratory for Robotics Vision, NAIST, from December 2014 to June 2019. From 2011 to 2014, he was a program-specific researcher with the Academic Center for Computing and Media Studies, Kyoto University, Japan.

His research interests include computer vision, pattern recognition, as well as multimedia content analysis, enhancement and generation.

E-mail: dylanywu@tencent.com

ORCID iD: 0000-0001-8010-6857

Ding-Heng Wang received the B. Eng. degree in mechanical engineering and automation, the M. Eng. degree in software engineering from Xi'an Jiaotong University, China in 2010 and 2014, respectively, and the Ph. D. degree in control science and engineering from Xi'an Jiaotong University, China in 2022. From 2014 to 2017, he was a software engineer in China Aerospace Science and Industry Corporation Limited, China.

His research interests include tensor decomposition, neural network compression, and efficient machine learning model.

E-mail: wangdai11@stu.xjtu.edu.cn

Xiao-Tong Lu received the B. Sc. degree in electronic engineering from Xidian University, China in 2016, where he is currently a Ph. D. degree candidate in intelligent information processing.

His research interests include deep learning, compressive sensing, image restoration and deep neural network compression.

E-mail: dmptcode@163.com

Fan Yang received the B. Sc. degree in geographical informational system from Nanjing University, China in 2012, and the M. Sc. degree in information science from Nara Institute of Science and Technology, Japan in 2018. He is currently a Ph. D. degree candidate in information science at Nara Institute of Science and Technology, Japan.

His research interest is on video processing.

E-mail: yang.fan.xv6@is.naist.jp

Man Yao received the M. Eng. degree in the electronic and communication engineering from Xi'an Jiaotong University, China in 2018. He is currently a Ph. D. degree candidate in control science and engineering at Xi'an Jiaotong University, China. From May 2021 to the present, he is doing an internship in Peng Cheng Laboratory, China.

His research interests include spiking neural network and dy-

namic neural network.

E-mail: manyao@stu.xjtu.edu.cn

Wei-Sheng Dong received the B. Sc. degree in electronic engineering from Huazhong University of Science and Technology, China in 2004, and the Ph. D. degree in circuits and system from Xidian University, China in 2010. He was a visiting student with Microsoft Research Asia, China in 2006. From 2009 to 2010, he was a research assistant with Department of Computing, Hong Kong Polytechnic University, China. In 2010, he joined School of Electronic Engineering, Xidian University, China as a lecturer, where he has been a professor since 2016. He was a recipient of the Best Paper Award at the SPIE Visual Communication and Image Processing (VCIP) in 2010. He has served as an Associate Editor of *IEEE Transactions on Image Processing* and is currently an Associate Editor of *SIAM Journal of Imaging Sciences*.

His research interests include inverse problems in image processing, deep learning, and parse representation.

E-mail: wsdong@mail.xidian.edu.cn

Jian-Bo Shi received the B. A. degree in computer science and mathematics from Cornell University, USA in 1994, and the Ph. D. degree in computer science from the University of California at Berkeley, USA in 1998. He joined The Robotics Institute at Carnegie Mellon University, USA in 1999 as a research faculty, and in 2003, University of Pennsylvania where he is currently a professor of Computer and Information Science. In 2007, he was awarded the Longuet-Higgins Prize for his work on Normalized Cuts.

His research focuses on first person vision, human behavior analysis and image recognition-segmentation. His other research interests include image/video retrieval, 3D vision, and vision based desktop computing. His long-term interests center around a broader area of machine intelligence, he wishes to develop a "visual thinking" module that allows computers not only to understand the environment around us, but also to achieve cognitive abilities such as machine memory and learning.

E-mail: jshi@seas.upenn.edu

Guo-Qi Li received the B. Eng. degree in automation from the Xi′an University of Technology, China in 2004, the M. Eng. degree in control engineering from Xi′an Jiaotong University, China in 2007, and the Ph. D. degree in electrical and electronic engineering from Nanyang Technological University, Singapore in 2011. From 2011 to 2014, he was a scientist with the Data Storage Institute and the Institute of High Performance Computing, Agency for Science, Technology and Research, Singapore. From 2014 to 2022, he was an assistant professor and associate professor at Tsinghua University, China. Since 2022, he has been with Institute of Automation, Chinese Academy of Sciences and the University of Chinese Academy of Sciences, where he is currently a full professor. He has authored or co-authored more than 150 journal and conference papers. He has been actively involved in professional services such as serving as a Tutorial Chair, an International Technical Program Committee Member, a PC member, a Publication Chair and a Track Chair for several international conferences. He is an Editorial-Board Member for *Control and Decision*, and served as Associate Editors for *Journal of Control and Decision* and *Frontiers in Neuroscience: Neuromorphic Engineering*. He is a reviewer for *Mathematical Reviews* published by the American Mathematical Society and serves as a reviewer for a number of prestigious international journals and top AI conferences including ICLR, NeurIPS, ICML, AAI, etc. He was the recipient of the First Class Prize in Science and Technology of the Chinese Institute of Command and Control in 2018, the Second Prize of Fujian Provincial Science and Technology Progress Award in 2020. He received the outstanding Young Talent Award of the Beijing Natural Science Foundation in 2021.

His research interests include brain-inspired intelligence, neuromorphic computing and spiking neural networks.

E-mail: guoqi.li@ia.ac.cn (Corresponding author)

ORCID iD: 0000-0002-8994-431X