

A Survey of the Recent Architectures of Deep Convolutional Neural Networks

Asifullah Khan^{1, 2*}, Anabia Sohail^{1, 2}, Umme Zahoora¹, and Aqsa Saeed Qureshi¹

¹ Pattern Recognition Lab, DCIS, PIEAS, Nilore, Islamabad 45650, Pakistan

² Deep Learning Lab, Center for Mathematical Sciences, PIEAS, Nilore, Islamabad 45650, Pakistan

asif@pieas.edu.pk

Abstract

Deep Convolutional Neural Networks (CNNs) are a special type of Neural Networks, which have shown exemplary performance on several competitions related to Computer Vision and Image Processing. Interesting application areas of CNN include Image Classification and Segmentation, Object Detection, Video Processing, Natural Language Processing, Speech Recognition, etc. The powerful learning ability of deep CNN is largely due to the use of multiple feature extraction stages that can automatically learn representations from the data. Availability of a large amount of data and improvements in the hardware technology have accelerated the research in CNNs, and recently very interesting deep CNN architectures have been reported. In fact, several interesting ideas to bring advancements in CNNs have been explored such as the use of different activation and loss functions, parameter optimization, regularization, and architectural innovations. However, the major improvement in representational capacity of the deep CNN is achieved through architectural innovations. Especially, the idea of exploiting spatial and channel information, depth and width of architecture, and multi-path information processing has gained substantial attention. Similarly, the idea of using a block of layers as a structural unit is also gaining popularity. This survey thus focuses on the intrinsic taxonomy present in the recently reported deep CNN architectures and consequently, classifies the recent innovations in CNN architectures into seven different categories. These seven categories are based on spatial exploitation, depth, multi-path, width, feature-map exploitation, channel boosting, and attention. Additionally, the elementary understanding of CNN components, current challenges and applications of CNN are also provided.

Keywords: Deep Learning, Convolutional Neural Networks, Architecture, Representational Capacity, Residual Learning, and Channel Boosted CNN.

1 Introduction

Machine Learning (ML) algorithms have the ability to learn the underlying relationship in data and thus are able to make decisions without requiring explicit instructions. Different ML algorithms have been developed since the early 1990s for the emulation of human sensory responses such as speech and vision, but they have generally failed to achieve human-level satisfaction [1]–[6]. In late 1990s, the challenging nature of Machine Vision (MV) tasks gave rise to a new class of Neural Networks (NN), called Convolutional Neural Networks (CNNs) [7].

CNNs are one of the best learning algorithms for understanding image content and have shown exemplary performance in image segmentation, classification, detection, and retrieval related tasks [8], [9]. The success of CNNs has captured attention beyond the academia. In industry, companies such as Google, Microsoft, AT&T, NEC, and Facebook have developed active research groups for exploring new architectures of CNN [10]. At present, most of the frontrunners of image processing and computer vision competitions are employing deep CNN based models.

The attractive feature of CNN is its ability to exploit spatial or temporal correlation in data. The topology of CNN is divided into multiple learning stages composed of a combination of the convolutional layers, non-linear processing units, and subsampling layers [11]. CNNs are feedforward multilayered hierarchical networks, where each layer, using a bank of convolutional kernels, performs multiple transformations [12]. Convolution operation helps in extraction of useful features from locally correlated data points. Output of the convolutional kernels is then assigned to non-linear processing unit (activation function), which not only helps in learning abstractions but also embeds non-linearity in the feature space. This non-linearity generates different patterns of activations for different responses and thus facilitates in learning of semantic differences in images. Output of the non-linear activation function is usually followed by subsampling, which helps in summarizing the results and also makes the input invariant to geometrical distortions [12], [13]. CNN, with the automatic feature extraction ability, reduces the need for a separate feature extractor [14]. Thus, CNN without exhaustive processing can learn good internal representation from raw pixels. Important attributes of CNN are hierarchical learning, automatic feature extraction, multi-tasking, and weight sharing [15]–[17].

CNN first came to limelight through the work of LeCuN in 1989 for processing of grid-like topological data (images and time series data) [7], [18]. The architectural design of CNN was inspired by Hubel and Wiesel's work and thus largely follows the basic structure of primate's visual cortex [19], [20]. Different stages of learning process in CNN shows quite resemblance with primate's ventral pathway of visual cortex (V1-V2-V4-IT/VTC) [21]. The visual cortex of primates first receives input from the retinotopic area, where multi-scale highpass filtering and contrast normalization are performed by the lateral geniculate nucleus. After this, detection is performed by different regions of the visual cortex categorized as V1, V2, V3, and V4. In fact, V1 and V2 portion of visual cortex are similar to convolutional and subsampling layers, whereas inferior temporal region resembles the higher layers of CNN, which makes inference about the image [22]. During training, CNN learns through backpropagation algorithm, by regulating the change in weights with respect to the target. Optimization of an objective function using backpropagation algorithm is similar to the response based learning of human brain. Multilayered, hierarchical structure of deep CNN, gives it the ability to extract low, mid, and high-level features. High-level features (more abstract features) are a combination of lower and mid-level features. Hierarchical feature extraction ability of CNN emulates the deep and layered learning process of the Neocortex in the human brain, which dynamically learns features from the raw data [23]. The popularity of CNN is largely due to its hierarchical feature extraction ability.

Deep architectures often have an advantage over shallow architectures, when dealing with complex learning problems. Stacking of multiple linear and non-linear processing units in a layer wise fashion provides the ability to learn complex representations at different levels of abstraction. Consequently, in recognition tasks consisting of hundreds of image categories, deep CNNs have shown substantial performance improvement over conventional vision based models [4], [24], [25]. The observation that the deep architectures can improve the representational capacity of a CNN heightened the use of CNN in image classification and segmentation tasks [26]. The availability of big data and advancements in hardware are also the main reasons of the recent success of deep CNNs. Empirical studies showed that if given enough training data, deep CNNs are good in learning of invariant representations and may achieve human level performance. In addition to its use as a supervised learning mechanism, the potential of deep CNNs can also be exploited to extract useful representations from large scale of unlabeled data. Recently, it is shown that different level of features including both low and high-level can be

transferred to a generic recognition task by exploiting the concept of Transfer Learning (TL) [27]–[29].

From late 1990s upto 2000, various improvements in CNN learning methodology and architecture were performed to make CNN scalable to large, heterogeneous, complex and multiclass problems. Innovations in CNNs include different aspects such as modification of processing units, parameter and hyper-parameter optimization strategies, design patterns and connectivity of layers, etc. CNN based applications became prevalent after the exemplary performance of AlexNet on ImageNet dataset in 2012 [26]. Major innovations in CNN have been proposed since then and are largely attributed to the restructuring of processing units and designing of new blocks. Zeiler and Fergus [30] gave the concept of layer-wise visualization of CNN to improve the understanding of feature extraction stages, which shifted the trend towards extraction of features at low spatial resolution in deep architecture as performed in VGG [31]. Nowadays, most of the new architectures are built upon the principle of simple and homogenous topology as introduced in VGG. Similarly, the Google deep learning group introduced an interesting idea of split, transform and merge, with the corresponding block known as inception block. The inception block for the very first time gave the concept of branching within a layer, which allows abstraction of features at different spatial scales [32]. In 2015, the concept of skip connections introduced by ResNet [33] for the training of deep CNNs gained popularity, and afterwards, this concept was used by most of the succeeding networks, such as Inception-ResNet, WideResNet, ResNeXt, etc., [34]–[36].

In order to improve the learning capacity of a CNN, different architectural designs such as WideResNet, ResNeXt, Pyramidal Net, Xception, PolyNet etc. explored the effect of multilevel transformations in terms of an additional cardinality and increase in width [34], [36]–[38]. Therefore, the focus of research shifted from parameter optimization and connections readjustment towards improved architectural design of the network. This shift resulted in many new architectural ideas such as channel boosting, spatial and feature-map wise exploitation and attention based information processing etc., [39]–[41].

In the past few years, different interesting surveys are conducted on deep CNNs that elaborate the basic components of CNN and their alternatives. The survey reported in [42] reviewed the famous architectures from 2012-2015 along with their basic components. Similarly, there are prominent surveys that discuss different algorithms and applications of CNN [12], [14], [16], [17], [43]. Likewise, the survey presented in [44] discusses taxonomy of CNNs based on

acceleration techniques. On the other hand, in this survey, we discuss the intrinsic taxonomy present in the recent and prominent CNN architectures reported from 2012-2020. The various CNN architectures discussed in this survey are broadly classified into seven main categories namely; spatial exploitation, depth, multi-path, width, feature-map exploitation, channel boosting, and attention based CNNs.

This survey also gives an insight into the basic structure of CNN as well as its historical perspective; presenting different eras of CNN that trace back from its origin to its latest developments and achievements. This survey will help the readers to develop the theoretical insight into design principles of CNN and thus may further accelerate the architectural innovations in CNN.

The rest of the paper is organized in the following order (shown in Fig. 1): Section 1 develops the systematic understanding of CNN, discusses its resemblance with primate's visual cortex, as well as its contribution in MV. In this regard, Section 2 provides the overview of basic CNN components and Section 3 discusses the architectural evolution of deep CNNs. Whereas, Section 4, discusses the recent innovations in CNN architectures and categorizes CNNs into seven broad classes. Section 5 and 6 shed light on applications of CNNs and current challenges, whereas section 7 discusses future work. Finally, the last section draws conclusion.

2 Basic CNN components

Nowadays, CNN is considered as one of the most widely used ML technique, especially in vision related applications. CNN can learn representations from the grid like data and recently it has shown substantial performance improvement in various ML applications. A typical block diagram of an ML system is shown in Fig. 2. Since, CNN possesses both good feature generation and discrimination ability, therefore in a typical ML system, CNN capabilities are exploited for feature generation and classification.

A typical CNN architecture generally comprises of alternate layers of convolution and pooling followed by one or more fully connected layers at the end. In some cases, fully connected layer is replaced with global average pooling layer. In addition to different mapping functions, different regulatory units such as batch normalization and dropout are also incorporated to optimize CNN performance [45]. The arrangement of CNN components play a

fundamental role in designing new architectures and thus achieving enhanced performance. This section briefly discusses the role of these components in a CNN architecture.

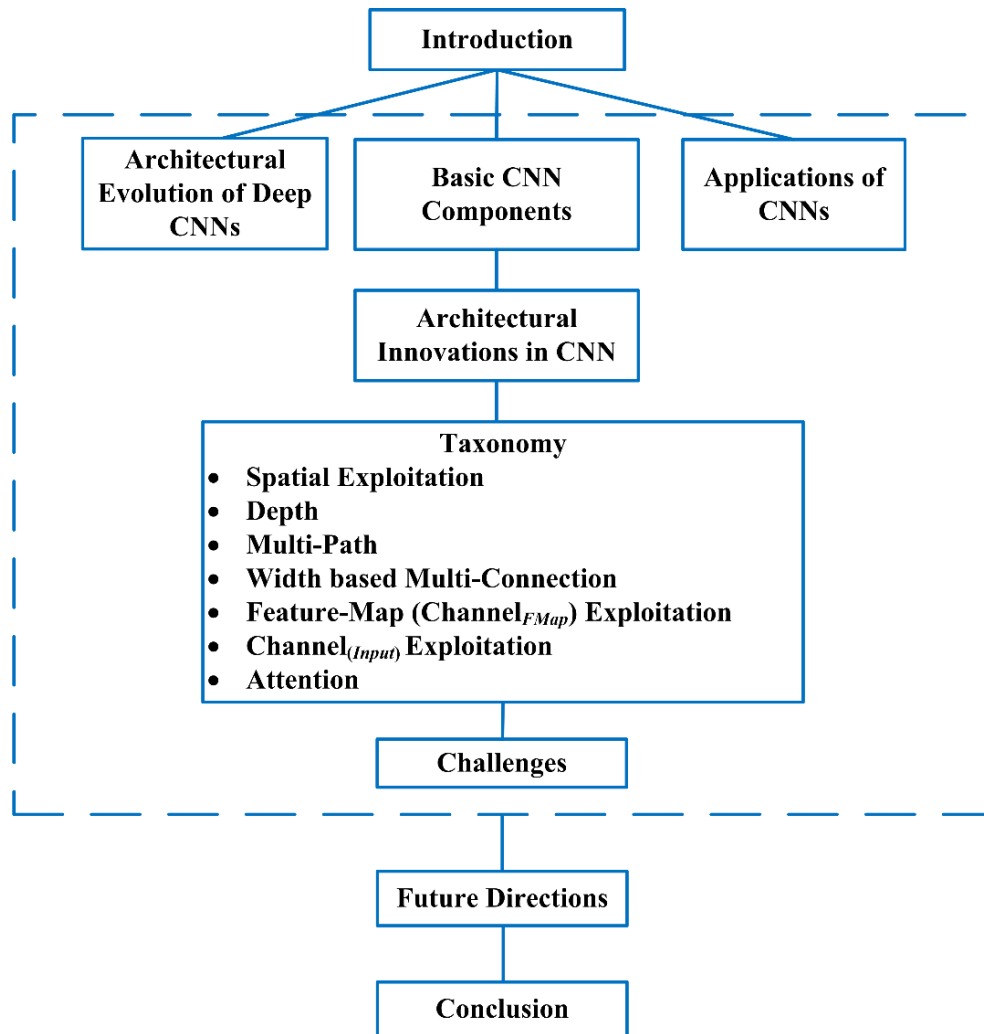


Fig. 1 Organization of the survey paper showing different sections.

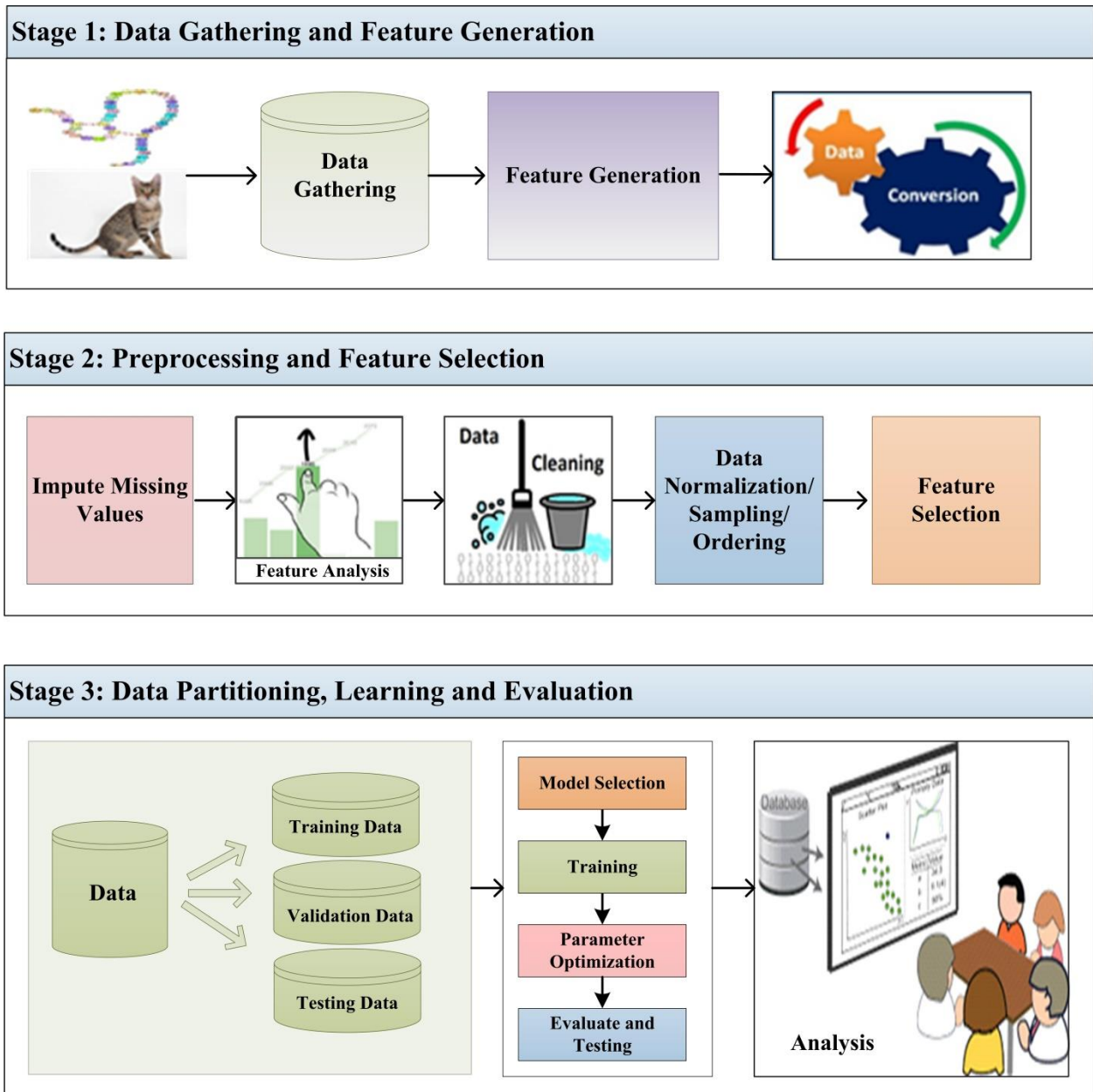


Fig. 2 Basic layout of a typical ML system having several stages.

Table 1 Definition of mathematical symbols

Symbol	Description
X	Total x coordinates of an image
x	x coordinate under consideration of an image
Y	Total y coordinates of an image
y	y coordinate under consideration of an image
c	Channel index
$i_c(x, y)$	(x, y) element of c^{th} channel of an image
L	Total number of layers
l	Layer number
K_l	Total number of kernels of l^{th} layer
k_l	Kernel number of l^{th} layer
U	Total number of rows of k^{th} kernel
u	u^{th} row under consideration
V	Total number of columns of k^{th} kernel
v	v^{th} column under consideration
$e_l^k(u, v)$	(u, v) element of k^{th} kernel of l^{th} layer
\mathbf{F}_l^k	Input feature matrix for l^{th} layer and k^{th} neuron
P	Total number of rows of feature matrix
p	p^{th} row under consideration
Q	Total number of columns of feature matrix
q	q^{th} column under consideration
$f_l^k(p, q)$	(p, q) element of feature matrix
$g_c(.)$	Convolution operation
$g_p(.)$	Pooling operation
$g_a(.)$	Activation function
$g_k(.)$	Concatenation operation
g_{t_g}	Transformation gate
g_{c_g}	Carry gate
$g_{sq}(.)$	Squeeze operation
$g_{ex}(.)$	Excitation operation
\mathbf{Y}_{l+1}^K	Weight vector showing feature-maps importance learned using SE operation
g_t	Transformation function for 2 layer NN implemented by SE block
g_{s_g}	Sigmoid gate implemented by SE block
g_{sm}	Soft mask
g_{tm}	Trunk mask
\mathbf{I}_B	Channel boosted input tensor

2.1 Convolutional layer

Convolutional layer is composed of a set of convolutional kernels (each neuron acts as a kernel). However, if the kernel is symmetric, the convolution operation becomes a correlation operation [18]. Convolutional kernel works by dividing the image into small slices commonly known as receptive fields. Division of an image into small blocks helps in extracting feature motifs. Kernel convolves with the images using a specific set of weights, by multiplying its elements with the corresponding elements of the image receptive field [45]. Convolution operation can be expressed as follows:

$$f_l^k(p, q) = \sum_c \sum_{x, y} i_c(x, y) \cdot e_l^k(u, v) \quad (1)$$

where, $i_c(x, y)$ is an element of the input image tensor I_c , which is element wise multiplied by $e_l^k(u, v)$ index of the k^{th} convolutional kernel k_l of the l^{th} layer. Whereas output feature-map of the k^{th} convolutional operation can be expressed as $\mathbf{F}_l^k = [f_l^k(1, 1), \dots, f_l^k(p, q), \dots, f_l^k(P, Q)]$. The different mathematical symbols used are defined in Table 1.

Due to weight sharing ability of convolutional operation, different sets of features within image can be extracted by sliding kernel with the same set of weights on the image and thus makes CNN parameter efficient as compared to the fully connected networks. Convolution operation may further be categorized into different types based on the type and size of filters, type of padding, and the direction of convolution [46]. For simplicity, we can drop the channel index,

2.2 Pooling layer

Feature motifs, which result as an output of convolution operation can occur at different locations in the image. Once features are extracted, its exact location becomes less important as long as its approximate position relative to others is preserved. Pooling or downsampling is an interesting local operation. It sums up similar information in the neighborhood of the receptive field and outputs the dominant response within this local region [47].

$$\mathbf{Z}_l^k = g_p(\mathbf{F}_l^k) \quad (2)$$

Equation (2) shows the pooling operation in which \mathbf{Z}_l^k represents the pooled feature-map of l^{th} layer for k^{th} input feature-map \mathbf{F}_l^k , whereas $g_p(\cdot)$ defines the type of pooling operation. The use of pooling operation helps to extract a combination of features, which are invariant to translational shifts and small distortions [13], [48]. Reduction in the size of feature-map to invariant feature set not only regulates complexity of the network but also helps in increasing the generalization by reducing overfitting. Different types of pooling formulations such as max, average, L2, overlapping, spatial pyramid pooling, etc. are used in CNN [49]–[51].

2.3 Activation function

Activation function serves as a decision function and helps in learning of complex patterns. Selection of an appropriate activation function can accelerate the learning process. Activation function for a convolved feature-map is defined in equation (3).

$$\mathbf{T}_l^k = g_a(\mathbf{F}_l^k) \quad (3)$$

In above equation, \mathbf{F}_l^k is an output of a convolution, which is assigned to activation function $g_a(\cdot)$ that adds non-linearity and returns a transformed output \mathbf{T}_l^k for l^{th} layer. In literature, different activation functions such as sigmoid, tanh, maxout, SWISH, ReLU, and variants of ReLU such as leaky ReLU, ELU, and PReLU are used to inculcate non-linear combination of features [42], [50], [52]–[54]. However, ReLU and its variants are preferred as they help in overcoming the vanishing gradient problem [55], [56]. One of the recently proposed activation function is MISH, which has shown better performance than ReLU in most of the recently proposed deep networks on benchmark datasets [57].

2.4 Batch normalization

Batch normalization is used to address the issues related to internal covariance shift within feature-maps. The internal covariance shift is a change in the distribution of hidden units' values, which slows down the convergence (by forcing learning rate to small value) and requires careful initialization of parameters. Batch normalization for a transformed feature-map \mathbf{F}_l^k is shown in equation (4).

$$\mathbf{N}_l^k = \frac{\mathbf{F}_l^k - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \quad (4)$$

In equation (4), \mathbf{N}_l^k represents normalized feature-map, \mathbf{F}_l^k is the input feature-map, μ_B and σ_B^2 depict mean and variance of a feature-map for a mini batch respectively. In order to avoid division by zero, ε is added for numerical stability. Batch normalization unifies the distribution of feature-map values by setting them to zero mean and unit variance [58]. Furthermore, it smoothens the flow of gradient and acts as a regulating factor, which thus helps in improving generalization of the network.

2.5 Dropout

Dropout introduces regularization within the network, which ultimately improves generalization by randomly skipping some units or connections with a certain probability. In NNs, multiple connections that learn a non-linear relation are sometimes co-adapted, which causes overfitting [59]. This random dropping of some connections or units produces several thinned network architectures, and finally one representative network is selected with small weights. This selected architecture is then considered as an approximation of all of the proposed networks [60].

2.6 Fully connected layer

Fully connected layer is mostly used at the end of the network for classification purpose. Unlike pooling and convolution, it is a global operation. It takes input from feature extraction stages and globally analyses output of all the preceding layers [61]. This makes a non-linear combination of selected features, which are used for the classification of data. [62].

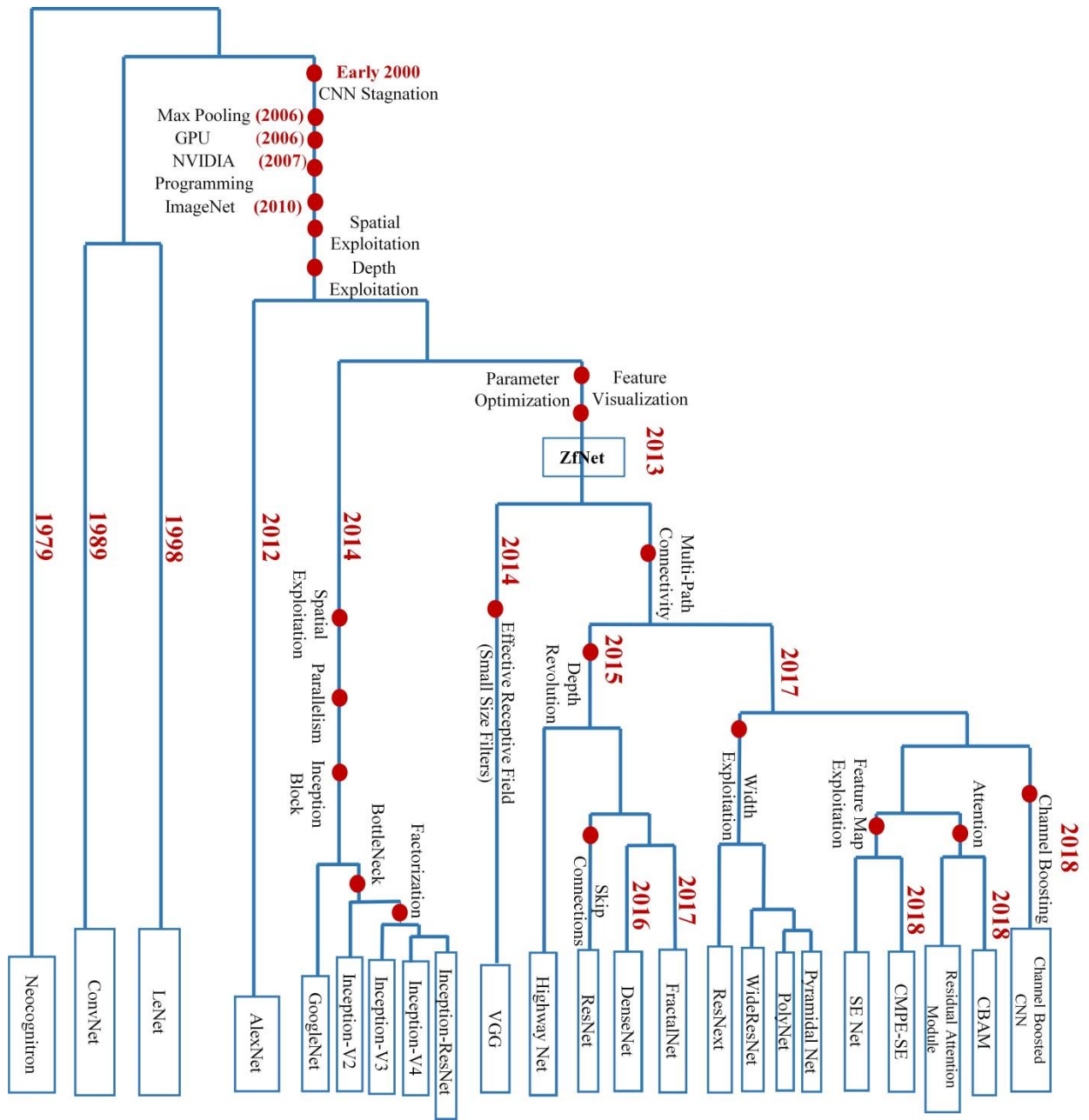


Fig. 3 Evolutionary history of deep CNNs showing architectural innovations from ConvNet till to date architectures.

3 Architectural evolution of deep CNNs

Nowadays, CNNs are considered as the most widely used algorithms among biologically inspired Artificial Intelligence (AI) techniques. CNN history begins with the neurobiological experiments conducted by Hubel and Wiesel (1959, 1962) [19], [63]. Their work provided a platform for many cognitive models and almost all of these were replaced by CNN. Over the decades, different efforts have been carried out to improve the performance of CNNs. Evolutionary history of deep CNN architectures is pictorially represented in Fig. 3. Improvements in CNN architectures can be categorized into five different eras and are discussed below.

3.1 Origin of CNN: Late 1980s-1999

CNNs have been applied to visual tasks since the late 1980s. In 1989, LeCuN et al. proposed the first multilayered CNN named as ConvNet, whose origin rooted in Fukushima's Neocognitron [64], [65]. LeCuN proposed supervised training of ConvNet, using Backpropagation algorithm in comparison to the unsupervised reinforcement learning scheme used by its predecessor Neocognitron [7], [66]. LeCuN's work thus made a foundation for the modern 2D CNNs. This ConvNet showed successful results for handwritten digit and zip code recognition related problems [67]. In 1998, LeCuN proposed an improved version of ConvNet, which was famously known as LeNet-5, and it started the use of CNN in classifying characters in a document recognition related applications [68], [69]. Due to the good performance of CNN in optical character and finger print recognition, its commercial use in ATM and Banks started in 1993 and 1996, respectively. In this era, LeNet-5 achieved many successful milestones for optical character recognition tasks, but it didn't perform that well on other image recognition problems.

3.2 Stagnation of CNN: Early 2000

In the late 1990s and early 2000, researchers had little insight into the internal working of CNN and it was considered as a black box. Complex architecture design and heavy processing made it hard to train CNN. It was widely presumed in early 2000 that the backpropagation algorithm used for training of CNN was not effective in converging to global minima of error surface. Thus, CNN was considered as less effective feature extractor compared to handcrafted features

[70]. Moreover, no comprehensive dataset of diverse categories of images was available at that time. Therefore, because of the insignificant improvement in CNN performance at the cost of high computational time, little attention was given to explore its role in different applications such as object detection, video surveillance, etc. At that time, other statistical methods and in particular, SVM became more popular than CNN due to their relatively high performance [71]–[73].

Meanwhile, a few research groups kept on working with CNNs and tried to optimize its performance. In 2003, Simard et al. improved CNN architecture and showed good results compared to SVM on a hand digit benchmark dataset; MNIST [68], [73]–[76]. This improvement in performance expedited the research in CNNs by extending their application's beyond optical character recognition to other script's character recognition, deployment in image sensors for face detection in video conferencing, and regulation of street crimes, etc. [76]–[78]. Likewise, CNN based systems were industrialized in markets for customers' tracking [12], [79], [80]. Moreover, CNN's potential in other applications such as medical image segmentation, anomaly detection, and robot vision was also explored [81]–[83].

3.3 Revival of CNN: 2006-2011

Deep CNNs have generally complex architecture and time intensive training phase that sometimes may span over weeks. In early 2000, there were a few parallel processing techniques and limited hardware resources for the training of deep Networks. Training of a deep CNNs with a typical activation function such as sigmoid may suffer from exponential decay and explosion of a gradient. Since 2006, significant efforts have been made to tackle the CNN optimization problem. In this regard, many interesting initialization and training strategies were reported to overcome the difficulties encountered in the training of deep CNNs and learning of invariant features. Hinton reported the concept of greedy layer-wise pre-training in 2006, which revived the research in deep learning [84], [85]. Experimental studies showed that both supervised and unsupervised pre-training can initialize a network in a better way than random initialization. Bengio and other researchers proposed that sigmoid activation function is not suitable for the training of the deep architectures with random initialization of weights. This observation started the use of activation functions other than sigmoid such as ReLU, tanh etc., [86]. The revival of a deep learning was one of the factors, which brought deep CNNs into limelight [87], [88].

Huang et al. (2006) used max pooling instead of subsampling, which showed good results by learning invariant features [48], [89]. In late 2006, researchers started using graphics processing units (GPUs) to accelerate the training of deep NN and CNN architectures [90]–[93]. In 2007, NVIDIA launched the CUDA programming platform, which allows exploitation of parallel processing capabilities of GPU with a greater degree [94], [95]. In essence, the use of GPUs for NN and CNN training and other hardware improvements were the main factors, which revived the research in CNN [92], [96]. In 2010, Fei-Fei Li’s group at Stanford, established a large database of images known as ImageNet, containing millions of annotated images belonging to a large number of classes [97]. This database was coupled with the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC), where the performances of various models have been evaluated and scored [98]. Similarly, in the same year, Stanford released PASCAL 2010 VOC dataset for object detection. ILSVRC and Neural Information Processing Systems conference (NIPS) are the two platforms that play a dominant role in strengthening research and increasing the use of CNN and thus making it popular.

3.4 Rise of CNN: 2012-2014

Availability of large training data and hardware advancements are the factors that contributed to the advancement in CNN research. But the main driving forces that have accelerated the research and give rise the use of CNNs in image classification and recognition tasks are parameter optimization strategies and new architectural ideas [42], [44], [99]. The main breakthrough in CNN performance was brought by AlexNet, which showed exemplary performance [26] in 2012-ILSVRC (reduced error rate from 25.8 to 16.4) as compared to conventional CV techniques [26].

In this era, a number of attempts were made to improve the performance of CNN; depth and parameter optimization strategies were explored with significant reduction in computational cost. Similarly, different architectural designs were proposed, whereby each new architecture tried to overcome the shortcomings of previously proposed architectures in combination with new structural reformulations. With the trend of designing very deep CNNs, it generally becomes difficult to independently determine filter dimensions, stride, padding, and other hyper-parameters for each layer. This problem is resolved by designing convolutional layers with fixed topology that can be repeated multiple times. This shifted the trend from custom layer design

towards modular and uniform layer design. Concept of modularity in CNNs made it easy to tailor them for different tasks effortlessly [31], [100]. In this connection, a different idea of branching and block within a layer was introduced by Google group [32]. It also should be noted that in this era, two different types of architectures; deep and narrow as well as deep and wide were in use.

3.5 Rapid increase in architectural innovations and applications of CNN: 2015-Present

The research in CNN is still going on and has a significant potential of improvement. It is generally observed that the major improvements in CNN performance occurred from 2015-2019. Representational capacity of a CNN usually depends on its depth and in a sense enriched feature set ranging from simple to complex abstractions can help in learning complex problems. However, the main challenge faced by deep architectures is that of the diminishing gradient. Initially, researchers tried to subside this problem by connecting intermediate layers to auxiliary learners [32]. In 2015, the emerging area of research was largely the development of new connections to improve the convergence rate of deep CNN architectures. In this regards, different ideas such as information gating mechanism across multiple layers, skip connections and cross layer channel connectivity were introduced [33], [101], [102]. Different experimental studies showed that state-of-the-art deep architectures such as VGG, ResNet, etc. not only showed good results for classification problems but, also for challenging recognition and localization problems like semantic and instance based object segmentation, scene parsing, scene location, etc. Most of the famous object detection and segmentation architectures such as Single Shot Multibox Detector (SSD), Region based CNN (R-CNN), Faster R-CNN, Mask R-CNN and Fully Convolutional Neural Network (FCNN) are built on the lines of ResNet, VGG, Inception, etc. Similarly, many interesting detection algorithms such as Feature Pyramid Networks, Cascade R-CNN, Libra R-CNN etc., modified the aforementioned architectures to improve the performance [103]–[105]. Applications of deep CNN were also extended to image captioning by combining these networks with recurrent neural network (RNN) and thus showed state-of-the-art results on MS COCO-2015 image captioning challenge [106]–[110].

Similarly in 2016, it was observed that the stacking of multiple transformations not only depth wise but, also in parallel fashion showed good learning for complex problems [36], [37]. Different researchers used hybrid of the already proposed architectures to improve deep CNN

performance [35], [111]–[115]. In 2017, focus of researchers was largely on designing of generic blocks that can be inserted at any learning stage in CNN architecture to improve the network representation [116]. This is one of the growing areas of research in CNN, where generic blocks are used to assign attention to spatial and feature-map (channel) information [40], [41], [117]. In 2018, a new idea of channel boosting was introduced by Khan et al. [39] to boost the performance of a CNN by learning diverse features as well as exploiting the already learnt features through the concept of TL.

However, two main concerns observed with deep and wide architectures are the high computational cost and memory requirement. As a result, it is very challenging to deploy state-of-the-art wide and deep CNN models in resource-constrained environments. Conventional convolution operation requires huge number of multiplications, which increases the inference time and restricts the applicability of CNN to low memory and time constraint applications [118]. Many real world applications such as autonomous vehicles, robotics, healthcare and mobile applications, perform the tasks that need to be carried on computationally limited platforms in a timely manner. Therefore, different modifications in CNN are performed to make them appropriate for resource constrained environments. Prominent modifications are knowledge distillation, training of small networks or squeezing of pretrained networks (such as pruning, quantization, hashing, huffman coding, etc.) [119]–[122]. GoogleNet exploited the idea of small networks, which replaces the conventional convolution with point-wise group convolution operation to make it computational efficient. Similarly, ShuffleNet used point-wise group convolution but with a new idea of channel shuffle that significantly reduces the number of operations without affecting the accuracy. In the same way, ANTNet proposed a novel architectural block known as ANTBlock, which at low computational cost, achieved good performance on benchmark datasets [123]–[125].

From 2012 up till now, a lot of improvements have been reported in CNN architectures. As regards the architectural advancement of CNNs, recently the focus of research has been on designing of new blocks that can boost network representation by exploiting feature-maps or manipulating input representation by adding artificial channels. Moreover, along with this, trend is towards design of light weight architectures without compromising the performance to make CNN applicable for resource constraint hardware.

4 Architectural innovations in CNN

Different improvements in CNN architecture have been made from 1989 to date. These improvements can be categorized as parameter optimization, regularization, structural reformulation, etc. However, it is observed that the main thrust in CNN performance improvement came from restructuring of processing units and designing of new blocks. Most of the innovations in CNN architectures have been made in relation with depth and spatial exploitation. Depending upon the type of architectural modifications, CNNs can be broadly categorized into seven different classes namely; spatial exploitation, depth, multi-path, width, feature-map exploitation, channel boosting, and attention based CNNs. Taxonomy of CNN architectures is pictorially represented in Fig. 4. Architectural details of the state-of-the-art CNN models, their parameters and performance on benchmark datasets are summarized into Table 2. On the other hand different online resources on deep CNN architectures, vision related dataset and their implementation platforms are mentioned in Table 3. In addition to this, strengths and weaknesses of the various architectures in a category wise manner are discussed in Table 5a-g.

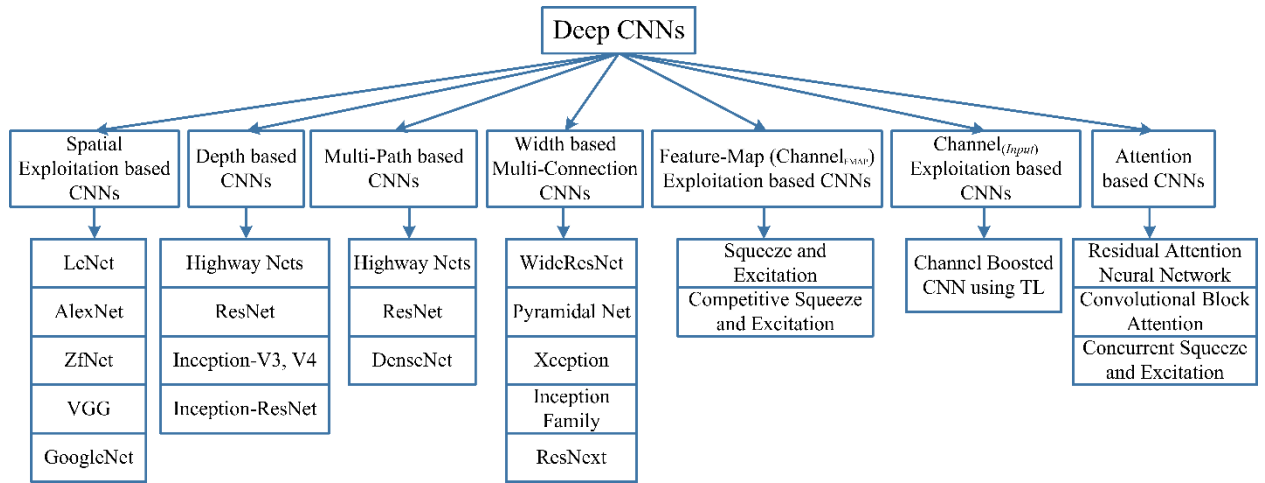


Fig. 4 Taxonomy of deep CNN architectures showing the seven different categories.

Table 2 Performance comparison of the recent architectures of different categories. Top 5 error rate is reported for all architectures.

Architecture Name	Year	Main contribution	Parameters	Error Rate	Depth	Category	Reference
LeNet	1998	- First popular CNN architecture	0.060 M	[dist]MNIST: 0.8 MNIST: 0.95	5	Spatial Exploitation	[69]
AlexNet	2012	- Deeper and wider than the LeNet - Uses Relu ,dropout and overlap Pooling - GPU's NVIDIA GTX 580	60 M	ImageNet: 16.4	8	Spatial Exploitation	[26]
ZfNet	2014	- Visualization of intermediate layers	60 M	ImageNet: 11.7	8	Spatial Exploitation	[30]
VGG	2014	- Homogenous topology - Uses small size kernels	138 M	ImageNet: 7.3	19	Spatial Exploitation	[31]
GoogLeNet	2015	- Introduces block concept - Split transform and merge idea	4 M	ImageNet: 6.7	22	Spatial Exploitation	[32]
Inception-V3	2015	- Handles the problem of a representational bottleneck - Replace large size filters with small filters	23.6 M	ImageNet: 3.5 Multi-Crop: 3.58 Single-Crop: 5.6	159	Depth + Width	[126]
Highway Networks	2015	- Introduced an idea of Multi-path	2.3 M	CIFAR-10: 7.76	19	Depth + Multi-Path	[102]
Inception-V4	2016	- Split transform and merge idea Uses asymmetric filters	35 M	ImageNet: 4.01	70	Depth +Width	[35]
Inception-ResNet	2016	- Uses split transform merge idea and residual links	55.8M	ImageNet: 3.52	572	Depth + Width + Multi-Path	[35]
ResNet	2016	- Residual learning - Identity mapping based skip connections	25.6 M 1.7 M	ImageNet: 3.6 CIFAR-10: 6.43	152 110	Depth + Multi-Path	[33]
DelugeNet	2016	- Allows cross layer information flow in deep networks	20.2 M	CIFAR-10: 3.76 CIFAR-100: 19.02	146	Multi-path	[127]
FractalNet	2016	- Different path lengths are interacting with each other without any residual connection	38.6 M	CIFAR-10: 7.27 CIFAR-10+: 4.60 CIFAR-100+: 4.59 CIFAR-100: 28.20 CIFAR-100+: 22.49 CIFAR100+: 21.49	20 40	Multi-Path	[128]
WideResNet	2016	- Width is increased and depth is decreased	36.5 M	CIFAR-10: 3.89 CIFAR-100: 18.85	28 -	Width	[36]
Xception	2017	- Depth wise convolution followed by point wise convolution	22.8 M	ImageNet: 0.055	126	Width	[129]
Residual Attention Neural Network	2017	- Introduces attention mechanism	8.6 M	CIFAR-10: 3.90 CIFAR-100: 20.4 ImageNet: 4.8	452	Attention	[41]
ResNeXt	2017	- Cardinality - Homogeneous topology - Grouped convolution	68.1 M	CIFAR-10: 3.58 CIFAR-100: 17.31 ImageNet: 4.4	29 - 101	Width	[34]
Squeeze & Excitation Networks	2017	- Models interdependencies between feature-maps	27.5 M	ImageNet: 2.3	152	Feature-Map Exploitation	[116]
DenseNet	2017	- Cross-layer information flow	25.6 M 25.6 M 15.3 M 15.3 M	CIFAR-10+: 3.46 CIFAR100+:17.18 CIFAR-10: 5.19 CIFAR-100: 19.64	190 190 250 250	Multi-Path	[101]
PolyNet	2017	- Experimented structural diversity - Introduces Poly Inception module - Generalizes residual unit using polynomial compositions	92 M	ImageNet: Single:4.25 Multi:3.45	- -	Width	[38]
PyramidalNet	2017	- Increases width gradually per unit	116.4 M 27.0 M 27.0 M	ImageNet: 4.7 CIFAR-10: 3.48 CIFAR-100: 17.01	200 164 164	Width	[37]
Convolutional Block Attention Module (ResNeXt101 (32x4d) + CBAM)	2018	- Exploits both spatial and feature-map information	48.96 M	ImageNet: 5.59	101	Attention	[40]
Concurrent Spatial & Channel Excitation Mechanism	2018	- Spatial attention - Feature-map attention - Concurrent placement of spatial and channel attention	-	MALC: 0.12 Visceral: 0.09	-	Attention	[117]
Channel Boosted CNN	2018	- Boosting of original channels with additional information rich generated artificial channels	-	-	-	Channel Boosting	[39]
Competitive Squeeze & Excitation Network CMPE-SE-WRN-28	2018	- Residual and identity mappings both are used for rescaling the feature-map	36.92 M 36.90 M	CIFAR-10: 3.58 CIFAR-100: 18.47	152 152	Feature-Map Exploitation	[130]

Table 3 Different online available resources for deep CNNs.

Category	Description	Source
Cloud based Platforms	Online free access to GPU and other deep learning accelerators	Google Colab: https://colab.research.google.com/notebooks/welcome.ipynb
		Colac: https://cocalc.com/
	Commercial platforms offered by world leading companies	FloydHub: https://www.floydhub.com/
		Amazon SageMaker: https://aws.amazon.com/deep-learning/
		Microsoft Azure ML Services: https://azure.microsoft.com/en-gb/services/machine-learning/
		Google Cloud: https://cloud.google.com/deep-learning-vm/
		IBM Watson Studio: https://www.ibm.com/cloud/deep-learning
Deep Learning Libraries	Deep learning libraries that provide built-in classes of NNs, fast numerical computation and automated estimation of gradients both for CPUs and GPUs.	Pytorch: https://pytorch.org/
		Tensorflow: https://www.tensorflow.org/
		MatConvNet: http://www.vlfeat.org/matconvnet/
		Keras: https://keras.io/
		Theano: http://deeplearning.net/software/theano/
		Caffe: https://caffe.berkeleyvision.org/
		Julia: https://julialang.org/
Lecture Series	Online available and freely accessible deep learning courses	Stanford Lecture Series: http://cs231n.stanford.edu/
		Udacity: https://www.udacity.com/course/deep-learning-nanodegree--nd101 https://www.udacity.com/course/deep-learning-pytorch--ud188
		Udemy: https://www.udemy.com/course/deep-learning-learn-cnns/ https://www.udemy.com/course/modern-deep-convolutional-neural-networks/ https://www.udemy.com/course/advanced-computer-vision/ https://www.udemy.com/course/deep-learning-convolutional-neural-networks-theano-tensorflow/ https://www.udemy.com/course/deep-learning-pytorch/
		Coursera: https://www.coursera.org/learn/convolutional-neural-networks https://www.coursera.org/specializations/deep-learning
Vision Datasets	Online freely accessible datasets of different categories of annotated images	ImageNet: http://image-net.org/
		COCO: http://cocodataset.org/#home
		Visual Genome: http://visualgenome.org/
		Open images: https://ai.googleblog.com/2016/09/introducing-open-images-dataset.html
		Places: http://places.csail.mit.edu/index.html
		Youtube-8M: https://research.google.com/youtube8m/index.html
		CelebA: http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html
		CIFAR10: https://www.cs.toronto.edu/~kriz/cifar.html
		Indoor Scene Recognition: http://web.mit.edu/torralba/www/indoor.html
		Computer Vision Datasets: https://computervisiononline.com/datasets
Deep Learning Accelerators	Energy and computation efficient deep learning accelerators	Fashion MNIST: https://research.zalando.com/welcome/mission/research-projects/fashion-mnist/
		NVIDIA: http://nvidia.org/
		FPGA: https://www.intel.com/content/www/us/en/artificial-intelligence/programmable/overview.html
		Eyeriss: http://eyeriss.mit.edu/
		Google's TPU: https://cloud.google.com/tpu/

4.1 Spatial exploitation based CNNs

CNNs have a large number of parameters and hyper-parameters such as the weights, biases, number of layers and processing units (neurons), filter size, stride, activation function, learning rate, etc. [131], [132]. As convolutional operation considers the neighborhood (locality) of input pixels, therefore different levels of correlation can be explored by using different filter sizes. Different sizes of filters encapsulate different levels of granularity; usually, small size filters extract fine-grained and large size extract coarse-grained information. Consequently, in early 2000, researchers exploited spatial filters to improve performance and explored the relation of spatial filter with the learning of the network. Different studies conducted in this era suggested that by the adjustment of filters, CNN can perform well both on coarse and fine-grained details.

4.1.1 LeNet

LeNet was proposed by LeCuN in 1998 [69]. It is famous due to its historical importance as it was the first CNN, which showed state-of-the-art performance on hand digit recognition tasks. It has the ability to classify digits without being affected by small distortions, rotation, and variation of position and scale. LeNet is a feed-forward NN that constitutes of five alternating layers of convolutional and pooling, followed by two fully connected layers. In early 2000, GPU was not commonly used to speed up training, and even CPUs were slow [133]. The main limitation of traditional multilayered fully connected NN was that it considers each pixel as a separate input and applies a transformation on it, which was a huge computational burden, specifically at that time [134]. LeNet exploited the underlying basis of image that the neighboring pixels are correlated to each other and are distributed across the entire image. Therefore, convolution with learnable parameters is an effective way to extract similar features at multiple locations with few parameters. This changed the conventional view of training where each pixel was considered as a separate input feature from its neighborhood and ignored the correlation among them. LeNet was the first CNN architecture, which not only reduced the number of parameters but was able to automatically learn features from raw pixels.

4.1.2 AlexNet

LeNet [69] though begin the history of deep CNNs but at that time, CNN was limited to hand digit recognition tasks, and didn't perform well to all classes of images. AlexNet [26] is

considered as the first deep CNN architecture, which showed ground breaking results for image classification and recognition tasks. AlexNet was proposed by Krizhevsky et al., who enhanced the learning capacity of the CNN by making it deeper and by applying a number of parameter optimizations strategies [26]. Basic architectural design of AlexNet is shown in Fig. 5. In early 2000, hardware limitations curtailed the learning capacity of deep CNN architectures by restricting them to small size. In order to get benefit of the representational capacity of deep CNNs, Alexnet was trained in parallel on two NVIDIA GTX 580 GPUs to overcome shortcomings of the hardware. In AlexNet, depth was extended from 5 (LeNet) to 8 layers to make CNN applicable for diverse categories of images. Despite the fact that generally, depth improves generalization for different resolutions of images but, the main drawback associated with increase in depth is overfitting. To address this challenge, Krizhevsky et al. (2012) exploited the idea of Hinton [60], [135], whereby their algorithm randomly skips some transformational units during training to enforce the model to learn features that are more robust. In addition to this, ReLU was employed as a non-saturating activation function to improve the convergence rate by alleviating the problem of vanishing gradient to some extent [56], [136]. Overlapping subsampling and local response normalization were also applied to improve the generalization by reducing overfitting. Other adjustments made were the use of large size filters (11x11 and 5x5) at the initial layers, compared to previously proposed networks. Due to efficient learning approach of AlexNet, it has a significant importance in the new generation of CNNs and has started a new era of research in the architectural advancements of CNNs.

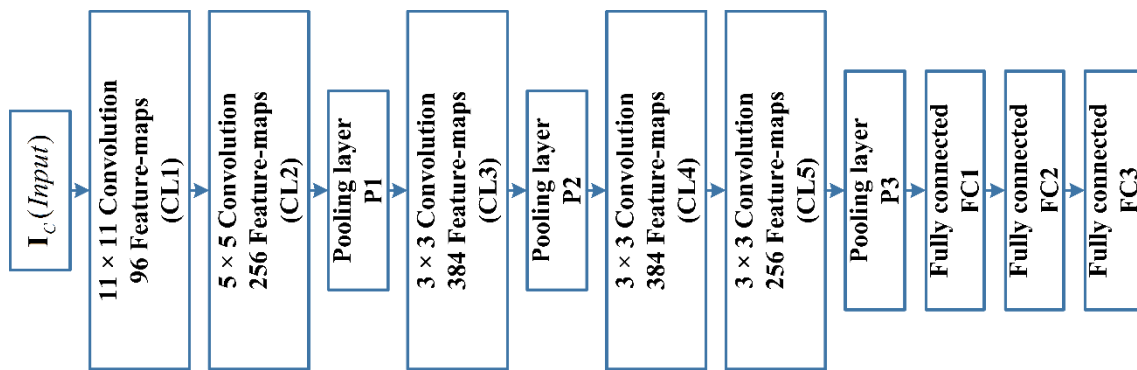


Fig. 5 Basic layout of AlexNet architecture showing its four convolution and three fully connected layers.

4.1.3 ZfNet

Learning mechanism of CNN, before 2013, was largely based on hit-and-trial, without knowing the exact reason behind the improvement. This lack of understanding limited the performance of deep CNNs on complex images. In 2013, Zeiler and Fergus proposed an interesting multilayer Deconvolutional NN (DeconvNet), which got famous as ZfNet [30]. ZfNet was developed to quantitatively visualize network performance. The idea of the visualization of network activity was to monitor CNN performance by interpreting neuron's activation. In one of the previous studies, Erhan et al. (2009) exploited the same idea and optimized performance of Deep Belief Networks (DBNs) by visualizing hidden layers' feature [137]. In the same manner, Le et al. (2011) evaluated the performance of deep unsupervised autoencoder (AE) by visualizing the image classes generated by the output neurons [138]. DeconvNet works in the same manner as the forward pass CNN but, reverses the order of convolutional and pooling operation. This reverse mapping projects the output of convolutional layer back to visually perceptible image patterns consequently gives the neuron-level interpretation of the internal feature representation learned at each layer [139], [140]. The idea of feature visualization proposed by ZfNet was experimentally validated on AlexNet using DeconvNet, which showed that only few neurons were active, while other neurons were dead (inactive) in the first and second layers of the network. Moreover, it showed that the features extracted by the second layer exhibited aliasing artifacts. Based on these findings, Zeiler and Fergus adjusted CNN topology and performed parameter optimization. Zeiler and Fergus maximized the learning of CNN by reducing both the filter size and stride to retain maximum number of features in the first two convolutional layers. This readjustment in CNN topology resulted in performance improvement, which suggested that features visualization can be used for identification of design shortcomings and for timely adjustment of parameters.

4.1.4 VGG

The successful use of CNNs in image recognition tasks has accelerated the research in architectural design. In this regard, Simonyan et al. proposed a simple and effective design principle for CNN architectures. Their architecture named as VGG was modular in layers pattern [31]. VGG was made 19 layers deep compared to AlexNet and ZfNet to simulate the relation of depth with the representational capacity of the network [26], [30]. ZfNet, which was a frontline

network of 2013-ILSVRC competition, suggested that small size filters can improve the performance of the CNNs. Based on these findings, VGG replaced the 11x11 and 5x5 filters with a stack of 3x3 filters layer and experimentally demonstrated that concurrent placement of small size (3x3) filters can induce the effect of the large size filter (5x5 and 7x7). Use of the small size filters provide an additional benefit of low computational complexity by reducing the number of parameters. These findings set a new trend in research to work with smaller size filters in CNN. VGG regulates the complexity of a network by placing 1x1 convolutions in between the convolutional layers, which in addition, learn a linear combination of the resultant feature-maps. For the tuning of the network, max pooling is placed after the convolutional layer, while padding was performed to maintain the spatial resolution [48]. VGG showed good results both for image classification and localization problems. VGG was at 2nd place in 2014-ILSVRC competition but, got fame due to its simplicity, homogenous topology, and increased depth. The main limitation associated with VGG was the use of 138 million parameters, which make it computationally expensive and difficult to deploy it on low resource systems.

4.1.5 GoogleNet

GoogleNet was the winner of the 2014-ILSVRC competition and is also known as Inception-V1. The main objective of the GoogleNet architecture was to achieve high accuracy with a reduced computational cost [32]. It introduced the new concept of inception block in CNN, whereby it incorporates multi-scale convolutional transformations using split, transform and merge idea. The architecture of inception block is shown in Fig. 6. In GoogleNet, conventional convolutional layers are replaced in small blocks similar to the idea of substituting each layer with micro NN as proposed in Network in Network (NIN) architecture [61]. This block encapsulates filters of different sizes (1x1, 3x3, and 5x5) to capture spatial information at different scales including both fine and coarse grain level. The exploitation of the idea of split, transform and merge by GoogleNet, helped in addressing a problem related to the learning of diverse types of variations present in the same category of images having different resolutions. GoogleNet regulates the computations by adding a bottleneck layer of 1x1 convolutional filter, before employing large size kernels. In addition to it, it used sparse connections (not all the output feature-maps are connected to all the input feature-maps), to overcome the problem of redundant information and reduced cost by omitting feature-maps that were not relevant. Furthermore, connection's density was reduced by using global average pooling at the last layer, instead of using a fully connected

layer. These parameter tunings caused a significant decrease in the number of parameters from 138 million to 4 million parameters. Other regulatory factors applied were batch normalization and use of RmsProp as an optimizer [141]. GoogleNet also introduced the concept of auxiliary learners to speed up the convergence rate. However, the main drawback of the GoogleNet was its heterogeneous topology that needs to be customized from module to module. Another, limitation of GoogleNet was a representation bottleneck that drastically reduces the feature space in the next layer and thus sometimes may leads to loss of useful information.

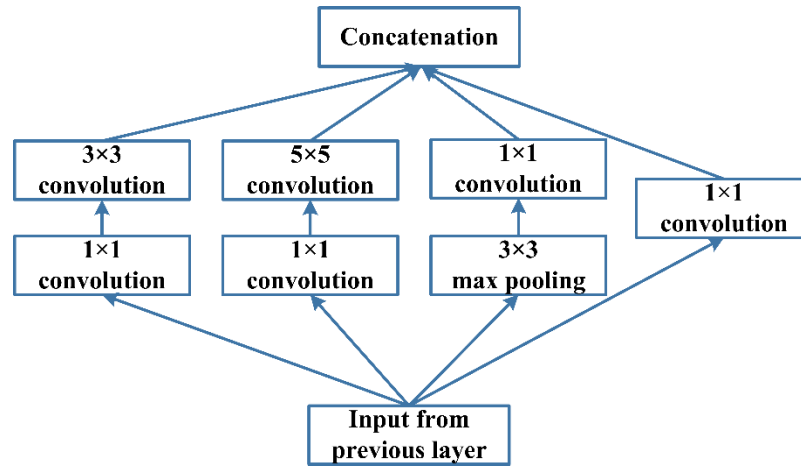


Fig. 6 Basic architecture of the inception block showing the split, transform and merge concept.

4.2 Depth based CNNs

Deep CNN architectures are based on the assumption that with the increase in depth, the network can better approximate the target function with a number of nonlinear mappings and more enriched feature hierarchies [142]. Network depth has played an important role in the success of supervised training. Theoretical studies have shown that deep networks can represent certain classes of function more efficiently than shallow architectures [143]. Csáji represented universal approximation theorem in 2001, which states that a single hidden layer is sufficient to approximate any function, but this comes at the cost of exponentially many neurons thus, often making it computationally unfeasible [144]. In this regard, Bengio and Delalleau [145] suggested that deeper networks have the potential to maintain the expressive power of the network at a reduced cost [146]. In 2013, Bengio et al. empirically showed that deep networks are

computationally more efficient for complex tasks [88], [147]. Inception and VGG, which showed the best performance in 2014-ILSVRC competition, further strengthen the idea that the depth is an essential dimension in regulating learning capacity of the networks [31], [32], [35], [126].

4.2.1 Highway Networks

Based on the intuition that the learning capacity can be improved by increasing the network depth, Srivastava et al. in 2015, proposed a deep CNN, named as Highway Networks [102]. The main problem concerned with deep networks is slow training and convergence speed [148]. Highway Networks exploited depth for learning enriched feature representation and introducing new cross-layer connectivity mechanism (discussed in Section 4.3.1) for the successful training of the deep networks. Therefore, Highway Networks are also categorized as multi-path based CNN architectures. Highway Networks with 50-layers showed better convergence rate than thin but deep architectures [98], [149]. Srivastava et al. experimentally showed that performance of a plain network decreases after adding hidden units beyond 10 layers [86]. Highway Networks, on the other hand, were shown to converge significantly faster than the plain ones, even with depth of 900 layers.

4.2.2 ResNet

ResNet was proposed by He et al., which is considered as a continuation of deep networks [33]. ResNet revolutionized the CNN architectural race by introducing the concept of residual learning in CNNs and devised an efficient methodology for training of deep networks. Similar to Highway Networks, it is also placed under the Multi-Path based CNNs, thus its learning methodology is discussed in Section 4.3.2. ResNet proposed 152-layers deep CNN, which won the 2015-ILSVRC competition. Architecture of the residual block of ResNet is shown in Fig. 7. ResNet, which was 20 and 8 times deeper than AlexNet and VGG respectively, showed less computational complexity than previously proposed networks [26], [31]. He et al. empirically showed that ResNet with 50/101/152 layers has less error on image classification task than 34 layers plain Net. Moreover, ResNet gained 28% improvement on the famous image recognition benchmark dataset named as COCO [150]. Good performance of ResNet on image recognition and localization tasks showed that representational depth is of central importance for many visual recognition tasks.

4.2.3 Inception-V3, V4 and Inception-ResNet

Inception-V3, V4 and Inception-ResNet, are improved versions of Inception-V1 and V2 [32], [35], [126]. The idea of Inception-V3 was to reduce the computational cost of deep networks without affecting the generalization. For this purpose, Szegedy et al. replaced large size filters (5x5 and 7x7) with small and asymmetric filters (1x7 and 1x5) and used 1x1 convolution as a bottleneck prior to the large filters [126]. This makes the traditional convolution operation more like cross-channel correlation. In one of the previous works, Lin et al. exploited the potential of 1x1 filters in NIN architecture [61]. Szegedy et al. [126] used the same concept in an intelligent way. In Inception-V3, 1x1 convolutional operation was used, which maps the input data into 3 or 4 separate spaces that are smaller than the original input space, and then maps all correlations in these smaller 3D spaces, via regular (3x3 or 5x5) convolutions. In Inception-ResNet, Szegedy et al. combined the power of residual learning and inception block [33], [35]. In doing so, filter concatenation was replaced by the residual connection. Moreover, Szegedy et al. experimentally showed that Inception-V4 with residual connections (Inception-ResNet) has the same generalization power as plain Inception-V4 but with increased depth and width. However, they observed that Inception-ResNet converges more quickly than Inception-V4, which clearly depicts that training with residual connections accelerates the training of Inception networks significantly.

4.3 Multi-Path based CNNs

Training of deep networks is a challenging task and this has been the subject of recent research on deep networks. Deep CNN generally, perform well on complex tasks. However, they may suffer from performance degradation, gradient vanishing or explosion problems, which are not caused by overfitting but instead by an increase in the depth [56], [151]. Vanishing gradient problem not only results in higher test error but also in higher training error [151]–[153]. For training deep networks, the concept of multi-path or cross-layer connectivity was proposed [101], [102], [127], [128]. Multiple paths or shortcut connections can systematically connect one layer to another by skipping some intermediate layers to allow the specialized flow of information across the layers [154], [155]. Cross-layer connectivity partitions the network into several blocks. These paths also try to solve the vanishing gradient problem by making gradient

accessible to lower layers. For this purpose, different types of shortcut connections are used, such as zero-padded, projection-based, dropout, skip connections, and 1x1 connections, etc.

4.3.1 Highway Networks

The increase in depth of a network improves performance mostly for complex problems, but it also makes training of the network difficult. In deep networks, due to a large number of layers, the backpropagation of error may results in small gradient values at lower layers. To solve this problem, in 2015, a new CNN architecture named as Highway Networks was proposed based on the idea of cross-layer connectivity [102]. In Highway Networks, the unimpeded flow of information across layers is enabled by imparting two gating units within a layer (equation (5)). The idea of a gating mechanism was inspired from Long Short Term Memory (LSTM) based Recurrent Neural Networks (RNN) [156], [157]. The aggregation of information by combining the l^{th} layer, and previous $l-j$ layers information creates a regularizing effect, making gradient-based training of very deep networks easy. This enables training of a network with more than 100 layers, even as deep as 900 layers with stochastic gradient descent algorithm. Cross-layer connectivity for Highway Network is defined in equation (5 & 6).

$$\mathbf{F}_{l+1}^k = g_c(\mathbf{F}_l^k, \mathbf{k}_l) \cdot g_{t_g}(\mathbf{F}_l^k, {}^{t_g}\mathbf{k}_l) \cdot g_{c_g}(\mathbf{F}_l^k, {}^{c_g}\mathbf{k}_l) \quad (5)$$

$$g_{c_g}(\mathbf{F}_l^k, {}^{c_g}\mathbf{k}_l) = 1 - g_{t_g}(\mathbf{F}_l^k, {}^{t_g}\mathbf{k}_l) \quad (6)$$

In equation (5), $g_c(\mathbf{F}_l^k, \mathbf{k}_l)$ represents the working of l^{th} hidden layer, whereas t_g and c_g are two gates that decide the flow of information across the layers. When t_g gate is open, $t_g = 1$ then transformed input is assigned to the next layer. Whereas, when the value of $t_g = 0$ then c_g gate establishes an effect of information highway and input \mathbf{F}_l^k of l^{th} layer is directly assigned to the next layer $l+1$ without any transformation.

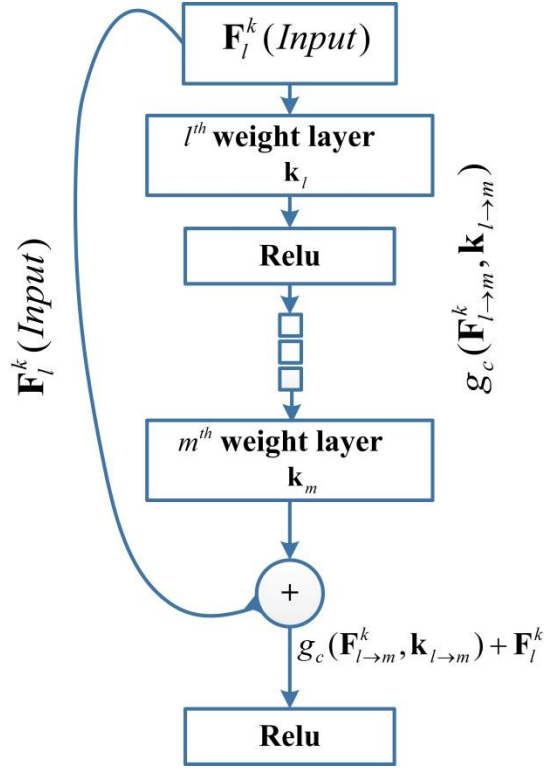


Fig. 7 Residual block as a basic structural unit of ResNet.

4.3.2 ResNet

In order to address the problems faced during training of deep networks, ResNet exploited the idea of bypass pathways used in Highway Networks [33]. Mathematical formulation of ResNet is expressed in equation (7, 8 & 9).

$$\mathbf{F}_{m+1}^{k'} = g_c(\mathbf{F}_{l \rightarrow m}^k, \mathbf{k}_{l \rightarrow m}) + \mathbf{F}_l^k \quad m \geq l \quad (7)$$

$$\mathbf{F}_{m+1}^k = g_a(\mathbf{F}_{m+1}^{k'}) \quad (8)$$

$$g_c(\mathbf{F}_{l \rightarrow m}^k, \mathbf{k}_{l \rightarrow m}) = \mathbf{F}_{m+1}^{k'} - \mathbf{F}_l^k \quad (9)$$

where, $g_c(\mathbf{F}_{l \rightarrow m}^k, \mathbf{k}_{l \rightarrow m})$ is a transformed signal, and \mathbf{F}_l^k is an input of l^{th} layer. In equation (7), $\mathbf{k}_{l \rightarrow m}$ shows the k^{th} processing unit (kernel), whereas $l \rightarrow m$ suggests that the residual block can be consists of one or more than one hidden layers. Original input \mathbf{F}_l^k is added to transformed signal ($g_c(\mathbf{F}_{l \rightarrow m}^k, \mathbf{k}_{l \rightarrow m})$) through bypass pathway (equation (7)) and thus results in an aggregated output $\mathbf{F}_{m+1}^{k'}$, which is assigned to the next layer after applying activation function $g_a(\cdot)$.

Whereas, $(\mathbf{F}_{m+1}^k - \mathbf{F}_l^k)$, returns a residual information, which is used to perform reference based optimization of weights. The distinct feature of ResNet is reference based residual learning framework. ResNet suggested that residual functions are easy to optimize and can gain accuracy for considerably increased depth.

ResNet introduced shortcut connections within layers to enable cross-layer connectivity that are data independent and parameter free in comparison to the gates of Highway Networks. In Highway Networks, when a gated shortcut is closed, the layers represent non-residual functions. However, in ResNet, residual information is always passed and identity shortcuts are never closed. Residual links (shortcut connections) speed up the convergence of deep networks, thus giving ResNet the ability to avoid gradient diminishing problems.

4.3.3 DenseNet

Similar to Highway Networks and ResNet, DenseNet was proposed to solve the vanishing gradient problem [33], [101], [102]. The problem with ResNet was that it explicitly preserves information through additive identity transformations due to which many layers may contribute very little or no information. To address this problem, DenseNet used cross-layer connectivity but, in a modified fashion. DenseNet connected each preceding layer to the next coming layer in a feed-forward fashion, thus feature-maps of all previous layers were used as inputs into all subsequent layers as expressed in equation (10 & 11).

$$\mathbf{F}_2^k = g_c(I_c, \mathbf{k}_1) \quad (10)$$

$$\mathbf{F}_l^k = g_k(\mathbf{F}_1^k, \dots, \mathbf{F}_{l-1}^k) \quad (11)$$

where \mathbf{F}_2^k and \mathbf{F}_l^k are the resultant feature-maps of 1st and l -1th transformation layers and $g_k(\cdot)$ is a function, which enables cross layer connectivity by concatenating proceeding layers information before assigning to new transformation layer l . This establishes $\frac{l(l+1)}{2}$ direct connections in DenseNet, as compared to l connections between a layer and its preceding layer in the traditional CNNs. It imprints the effect of cross-layer depthwise convolutions. As DenseNet concatenates the previous layers features instead of adding them, thus, the network may gain the ability to explicitly differentiate between information that is added to the network and information that is preserved. DenseNet has narrow layer structure; however, it becomes

parametrically expensive with an increase in a number of feature-maps. The direct admittance of each layer to the gradients through the loss function improves the flow of information throughout the network. This incorporates a regularizing effect, which reduces overfitting on tasks with smaller training sets.

4.4 Width based Multi-Connection CNNs

During 2012-2015, the focus was largely on exploiting the power of depth along with the effectiveness of multi-pass regulatory connections in network regularization [33], [102]. However, Kawaguchi et al. reported that the width of network is also important [158]. Multilayer perceptron gained an advantage of mapping complex functions over perceptron by making parallel use of multiple processing units within a layer. This suggests that width is an important parameter in defining principles of learning along with depth. Lu et al. (2017), and Hanin and Sellke (2017) have recently shown that NNs with ReLU activation function have to be wide enough in order to hold universal approximation property along with increase in depth [159]. Moreover, a class of continuous functions on a compact set cannot be arbitrarily well approximated by an arbitrarily deep network, if the maximum width of the network is not larger than the input dimension [147], [160]. Although, stacking of multiple layers (increasing depth) may learn diverse feature representations, but may not necessarily increase the learning power of the NN. One major problem linked with deep architectures is that some layers or processing units may not learn useful features. To tackle this problem, the focus of research shifted from deep and narrow architecture towards thin and wide architectures.

4.4.1 WideResNet

It is concerned that the main drawback associated with deep residual networks is the feature reuse problem in which some feature transformations or blocks may contribute very little to learning [161]. This problem was addressed by WideResNet [36]. Zagoruyko and Komodakis suggested that the main learning potential of deep residual networks is due to the residual units, whereas depth has a supplementary effect. WideResNet exploited the power of the residual blocks by making ResNet wide rather than deep [33]. WideResNet increased width by introducing an additional factor k , which controls the width of the network. WideResNet showed that the widening of the layers may provide a more effective way of performance improvement

than by making the residual networks deep. Deep networks improved representational capacity, but they have some demerits such as time intensive training, feature reuse, and gradient vanishing and exploding problem. He et al. addressed feature reuse problem by incorporating dropout in residual blocks to regularize network in an effective way [33]. Similarly, Huang et al. introduced the concept of stochastic depth by exploiting dropouts to solve vanishing gradient and slow learning problem [114]. It was observed that even fraction improvement in performance may require the addition of many new layers. However, Zagoruyko and Komodakis (2016), empirically showed that though WideResNet was twice in number of parameters as compared to ResNet, but can be trained in a better way than the deep networks [36]. Wider residual network was based on the observation that almost all architectures before residual networks, including the most successful Inception and VGG, were wide as compared to ResNet. In WideResNet, learning is made effective by adding a dropout in between the convolutional layers rather than inside a residual block.

4.4.2 Pyramidal Net

In earlier deep CNN architectures such as AlexNet, VGG, and ResNet, due to the deep stacking of multiple convolutional layers, depth of feature-maps increase in subsequent layers. However, spatial dimension decreases, as each convolutional layer or block is followed by a sub-sampling layer [26], [31], [33]. Therefore, Han et al. argued that in deep CNNs, drastic increase in the feature-map depth and at the same time the loss of spatial information limits the learning ability of CNN [37]. ResNet has shown remarkable results for image classification problem. However in ResNet, the deletion of residual block, where dimension of both spatial and feature-map (channel) varies (feature-map depth increases, while spatial dimension decreases), generally deteriorates performance. In this regard, stochastic ResNet, improved the performance by reducing information loss associated with the dropping of the residual unit [114]. To increase the learning ability of ResNet, Han et al. proposed, Pyramidal Net [37]. In contrast to the drastic decrease in spatial width with an increase in depth by ResNet, Pyramidal Net increases the width gradually per residual unit. This strategy enables pyramidal Net to cover all possible locations instead of maintaining the same spatial dimension within each residual block until down-sampling occurs. Because of a gradual increase in the depth of features map in a top-down fashion, it was named as pyramidal Net. In pyramidal network, depth of feature-maps is regulated by factor l , and is computed using equation (12).

$$d_l = \begin{cases} 16 & \text{if } l=1, \\ \left\lfloor d_{l-1} + \frac{\lambda}{n} \right\rfloor & \text{if } 2 \leq l \leq n+1 \end{cases} \quad (12)$$

where, d_l denotes the dimensions of l^{th} residual block and n describes the number of the residual block, whereas λ is a step size and $\frac{\lambda}{n}$ regulates the increase in depth. The depth regulating factor tries to distribute the burden of increase in depth of feature-maps. Residual connections were inserted in between the layers by using zero-padded identity mapping. The advantage of zero-padded identity mapping is that it needs less number of parameters as compared to the projection based shortcut connection, hence may result in better generalization [162]. Pyramidal Net uses two different approaches for the widening of the network including addition, and multiplication based widening. The difference between the two types of widening is that additive pyramidal structure increases linearly, whereas multiplicative one increases geometrically [52], [58]. However, major problem with Pyramidal Net is that with the increase in width, a quadratic times increase in both space and time occurs.

4.4.3 Xception

Xception can be considered as an extreme Inception architecture, which exploits the idea of depthwise separable convolution [129]. Xception modified the original inception block by making it wider and replacing the different spatial dimensions (1x1, 5x5, 3x3) with a single dimension (3x3) followed by 1x1 convolution to regulate computational complexity.

Architecture of Xception block is shown in Fig. 8. Xception makes the network computationally efficient by decoupling spatial and feature-map (channel) correlation, which is mathematically expressed in equation (13 & 14). It works by first mapping the convolved output to low dimensional embeddings using 1x1 convolution and then spatially transforms it n times, where n is a width defining cardinality, which determines the number of transformations.

$$f_{l+1}^k(p, q) = \sum_{x, y} f_l^k(x, y) \cdot e_l^k(u, v) \quad (13)$$

$$\mathbf{F}_{l+2}^k = g_c(\mathbf{F}_{l+1}^k, \mathbf{k}_{l+1}) \quad (14)$$

In equation (14), \mathbf{k}_l is a k^{th} kernel of l^{th} layer having depth one, which is spatially convolved across k^{th} feature-map \mathbf{F}_l^k , where (x, y) and (u, v) show the spatial indices of feature-

map and kernel respectively. In depthwise separable convolution, it is to be noted that number of kernels K is equal to number of input feature-maps contrary to conventional convolutional layer where number of kernels are independent of previous layer feature maps. Whereas \mathbf{k}_{l+1} is k^{th} kernel of (1×1) spatial dimension for $l+1^{\text{th}}$ layer, which performs depthwise convolution across output feature-maps $[\mathbf{F}_{l+1}^1, \dots, \mathbf{F}_{l+1}^k, \dots, \mathbf{F}_{l+1}^K]$ of l^{th} layer, used as input of $l+1^{\text{th}}$ layer. Xception makes computation easy by separately convolving each feature-map across spatial axes, which is followed by pointwise convolution (1×1 convolutions) to perform cross-channel correlation. In conventional CNN architectures; convolutional operation uses only one transformation segment, inception block uses three transformation segments, whereas in Xception number of transformation segments is equal to the number of feature-maps. Although, the transformation strategy adopted by Xception does not reduce the number of parameters, but it makes learning more efficient and results in improved performance.

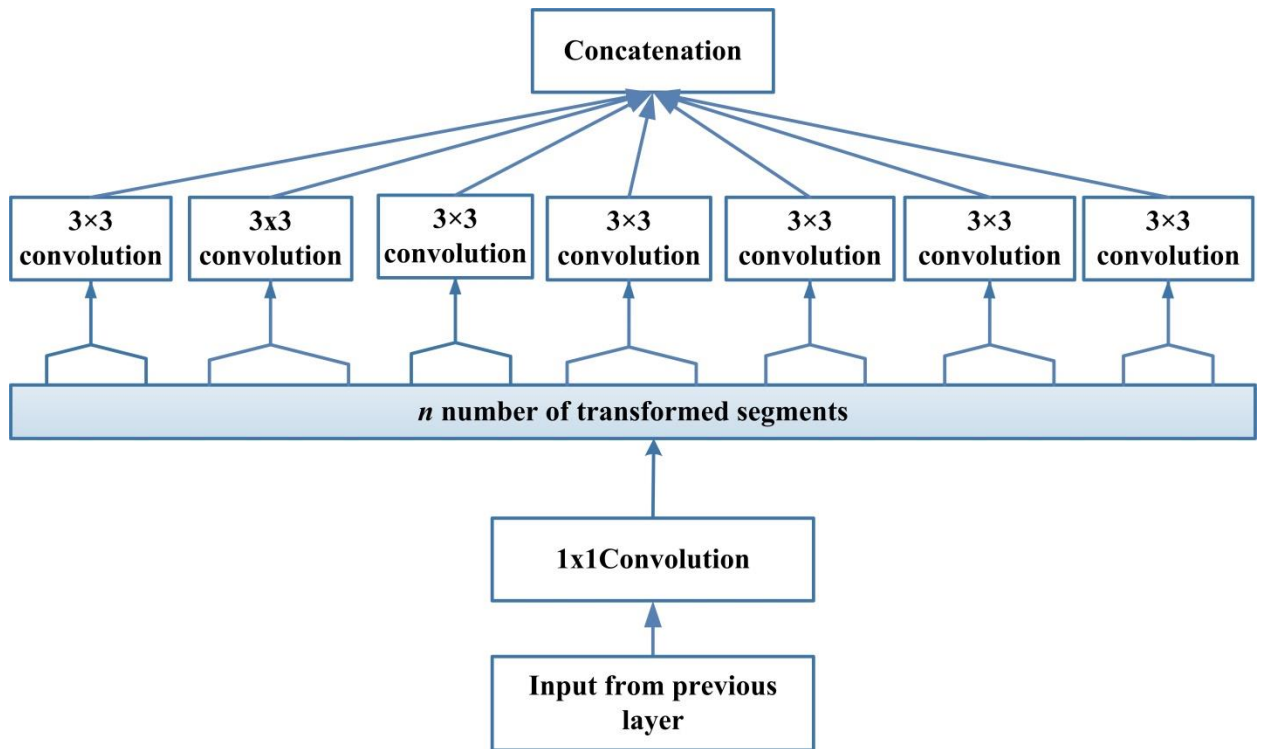


Fig. 8 Xception building block and its n sets of transformation.

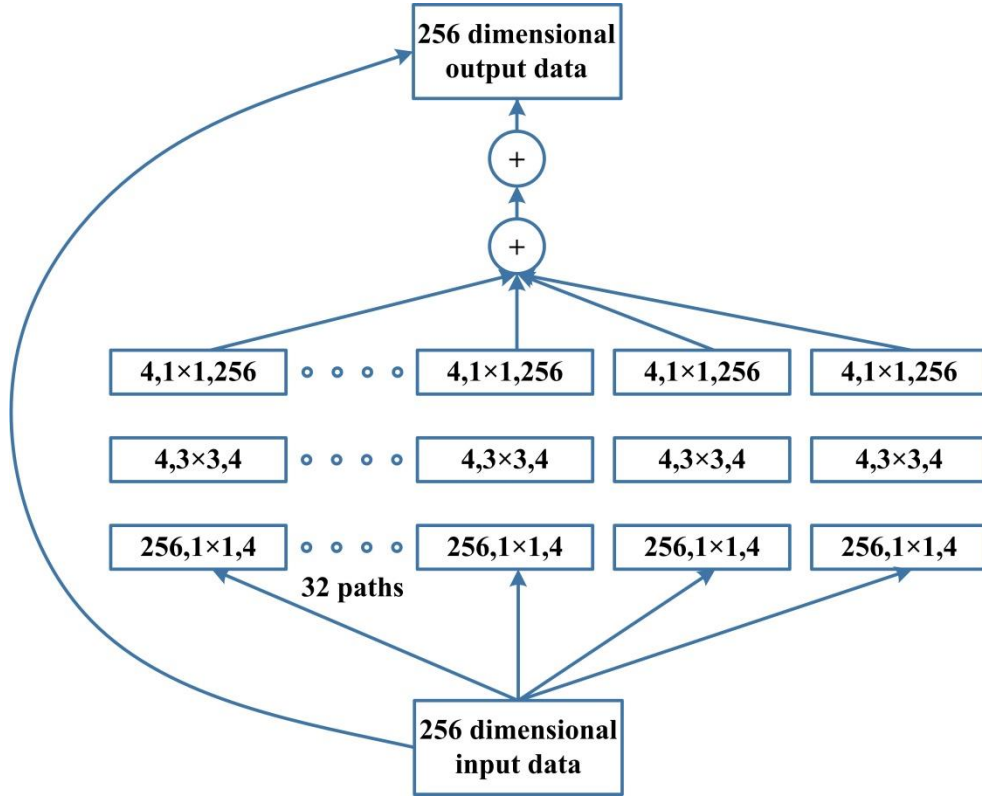


Fig. 9 ResNeXt building block showing the different paths of transformation.

4.4.4 ResNeXt

ResNeXt, also known as Aggregated Residual Transform Network, is an improvement over the Inception Network [34]. Xie et al. exploited the concept of the split, transform and merge in a powerful but simple way by introducing a new term; cardinality [32]. Cardinality is an additional dimension, which refers to the size of the set of transformations [163], [164]. Inception network has not only improved learning capability of conventional CNNs but also makes a network resource effective. However, due to the use of diverse spatial embedding's (such as use of 3×3 , 5×5 and 1×1 filter) in the transformation branch, each layer needs to be customized separately. ResNeXt utilized the deep homogenous topology of VGG and simplified GoogleNet architecture by fixing spatial resolution to 3×3 filters within the split, transform and merge block. Whereas, it used residual learning to improve the convergence of deep and wide network [31]–[33]. Building

block for ResNeXt is shown in Fig. 9. ResNeXt used multiple transformations within a split, transform and merge block and defined these transformations in terms of cardinality. Xie et al. (2017) showed that increase in cardinality significantly improves the performance. The complexity of ResNeXt was regulated by applying low embedding's (1x1 filters) before 3x3 convolution. Whereas training was optimized by using skip connections [128].

4.4.5 Inception Family

Inception family of CNNs also comes under the class of width based methods [32], [35], [126]. In Inception networks, within a layer, varying sizes of the filters were used which increased the output of the intermediate layers. The use of the different sizes of filters is helpful in capturing the diversity in high-level features. Salient characteristics of Inception family are discussed in section 4.1.5 and 4.2.3.

4.5 Feature-Map (Channel_{FMap}) Exploitation based CNNs

CNN became popular for MV tasks because of its hierarchical learning and automatic feature extraction ability [12]. Feature selection plays an important role in determining the performance of classification, segmentation, and detection modules. In CNN, features are dynamically selected by tuning the weights associated with a kernel (mask). Also, multiple stages of feature extraction are used, which can extract diverse types of features (known as feature-maps or channels in CNN). However, some of the feature-maps impart little or no role in object discrimination [116]. Enormous feature sets may create an effect of noise and thus lead to over-fitting of the network. This suggests that apart from network engineering, selection of feature-maps can play an important role in improving generalization of the network. In this section, feature-maps and channels will be interchangeably used as many researchers have used the word channels for the feature-maps.

4.5.1 Squeeze and Excitation Network

Squeeze and Excitation Network (SE-Network) was reported by Hu et al. [116]. They proposed a new block for the selection of feature-maps (commonly known as channels) relevant to object discrimination. This new block was named as SE-block (shown in Fig. 10), which suppresses the less important feature-maps, but gives high weightage to the class specifying feature-maps. SE-

Network reported record decrease in error on ImageNet dataset. SE-block is a processing unit that is designed in a generic way and therefore can be added in any CNN architecture before the convolution layer. The working of this block consists of two operations; squeeze and excitation. Convolution kernel captures information locally, but it ignores contextual relation of features (correlation) that are outside of this receptive field [46]. To obtain a global view of feature-maps, the squeeze block generates feature-map wise statistics (also known as feature-map motifs or descriptors) by suppressing spatial information of the convolved input. As global average pooling has the potential to learn the extent of target object effectively [61], [165], therefore, it is employed by the squeeze operation $g_{sq}(\cdot)$ using the following equation (15):

$$s_l^k = g_{sq}(\mathbf{F}_l^k) = \frac{1}{P \times Q} \sum_{p,q} f_l^k(p,q) \quad (15)$$

where, s_l^k represents a feature descriptor for k^{th} feature-map of l^{th} layer, and $P \times Q$ defines the spatial dimension of feature-map \mathbf{F}_l^k . Whereas output of squeeze operation $\mathbf{S}_l^K = [s_l^1, \dots, s_l^K]$ for K number of convolved feature-maps for l^{th} layer is assigned to the excitation operation $g_{ex}(\cdot)$, which models motif-wise interdependencies by exploiting gating mechanism. Excitation operation assigns weights to feature-maps using two layer feed forward NN, which is mathematically expressed in equation (16).

$$y_{l+1}^k = g_{ex}(\mathbf{S}_l^K) = g_{s_g}(\mathbf{w}_2, g_t(\mathbf{S}_l^K, \mathbf{w}_1)) \quad (16)$$

In equation (16), y_{l+1}^k denotes weightage for input feature-map \mathbf{F}_{l+1}^k of next layer $(l+1)$, where $g_t(\cdot)$ and $g_{s_g}(\cdot)$ apply the ReLU based non-linear transformation and sigmoid gate, respectively. Similarly, $\mathbf{Y}_{l+1}^K = [y_{l+1}^1, \dots, y_{l+1}^K]$ show the weightage for K number of convolved feature-maps that are used to rescale them before assigning to the $l+1^{\text{th}}$ layer. In excitation operation, \mathbf{w}_1 and \mathbf{w}_2 both are used as transformation weight vectors and regulating factors to limit the model complexity and aid the generalization [52], [53]. The output of first hidden transformation in NN is preceded by ReLU activation function, which inculcates non-linearity in motif responses. Gating mechanism is exploited in SE-block using sigmoid activation function, which models the non-linear responses of the feature-maps and assigns a weight based on feature-map relevance [166]. SE-block adaptively recalibrates each layer's feature-maps by multiplying convolved input with the motif responses.

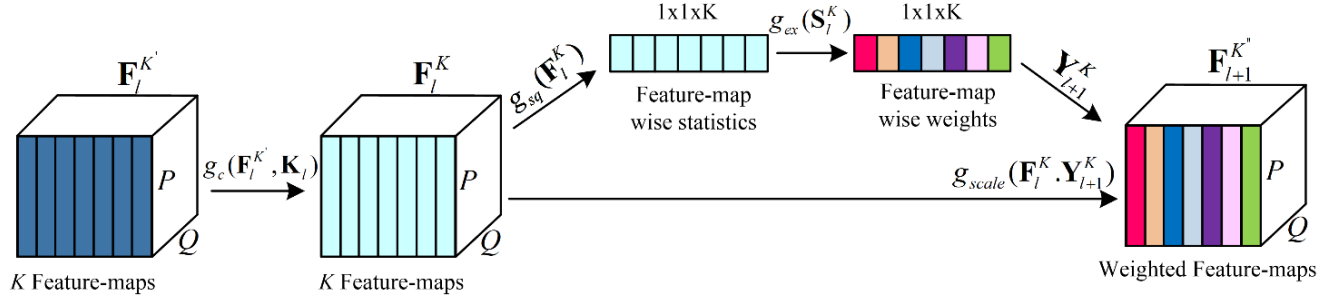


Fig. 10 Squeeze and Excitation block showing the computation of masks for the recalibration of feature-maps that are commonly known as channels in literature.

4.5.2 Competitive Squeeze and Excitation Networks

Competitive Inner-Imaging Squeeze and Excitation for Residual Network also known as CMPE-SE Network was proposed by Hu et al. in 2018 [130]. Hu et al. used the idea of SE-block to improve the learning of deep residual networks [116]. SE-Network recalibrates the feature-maps based upon their contribution in class discrimination. However, the main concern with SE-Net is that in ResNet, it only considers the residual information for determining the weight of each feature-map [116]. This minimizes the impact of SE-block and makes ResNet information redundant. Hu et al. addressed this problem by generating feature-map wise motifs (statistics) from both residual and identity mapping based feature-maps. In this regard, global representation of feature-maps is generated using global average pooling operation (equation (17)), whereas relevance of feature-maps is estimated by establishing competition between feature descriptors of residual and identity mappings. This phenomena is termed as inner imaging [130]. CMPE-SE block not only models the relationship between residual feature-maps but also maps their relation with identity feature-map. The mathematical expression for CMPE-SE block is represented using the following equation:

$$\mathbf{S}_l^k, \mathbf{S}_{m+1}^k = g_{sq}(\mathbf{F}_l^k), g_{sq}(\mathbf{F}_{m+1}^k) \quad (17)$$

$$\mathbf{Y}_{m+1}^k = g_{ex}(g_k(\mathbf{S}_l^k, \mathbf{S}_{m+1}^k)) \quad (18)$$

$$\mathbf{F}_{m+1}^k = \mathbf{Y}_{m+1}^k \mathbf{F}_{m+1}^{k'} \quad (19)$$

where \mathbf{F}_l^K and $\mathbf{F}_{m+1}^{K'}$ are the identity and residual mapping of input \mathbf{F}_l^K respectively. SE block is implemented by applying squeeze operation $g_{sq}(\cdot)$ both on residual and the identity feature-maps and their receptive output is used as joint input of excitation operation $g_{ex}(\cdot)$. Where $g_k(\cdot)$ represents the concatenation operation. The output masks of excitation operation (equation (18)) are multiplied with residual information (equation (19)) to rebuild each feature-map importance. The backpropagation algorithm thus tries to optimize the competition between identity and residual feature-maps and the relationship between all feature-maps in the residual block.

4.6 Channel_(Input) Exploitation based CNNs

Image representation plays an important role in determining the performance of the image processing algorithms including both conventional and deep learning algorithms. A good representation of the image is one that can define the salient features of an image from a compact code. In MV tasks, various types of conventional filters are applied to extract different levels of information for a single type of image [25], [167]. These diverse representations are then used as an input of the model to improve performance [168], [169]. Now CNN is an effective feature learner that can automatically extract discriminating features depending upon the problem [170]. However, the learning of CNN relies on input representation. The lack of diversity and absence of class discernable information in the input may affect CNN performance as a discriminator. For this purpose, the concept of channel boosting (input channel dimension) using auxiliary learners is introduced in CNNs to boost the representation of the network [39].

4.6.1 Channel Boosted CNN using TL

In 2018, Khan et al. proposed a new CNN architecture named as Channel boosted CNN (CB-CNN) based on the idea of boosting the number of input channels for improving the representational capacity of the network [39]. Block diagram of CB-CNN is shown in Fig. 11. Channel boosting is performed by artificially creating extra channels (known as auxiliary channels) through auxiliary deep generative models and then exploiting it through the deep discriminative models. CB-CNN is mathematically expressed in equation (20 & 21).

$$\mathbf{I}_B = g_k(\mathbf{I}_C, [\mathbf{A}_1, \dots, \mathbf{A}_M]) \quad (20)$$

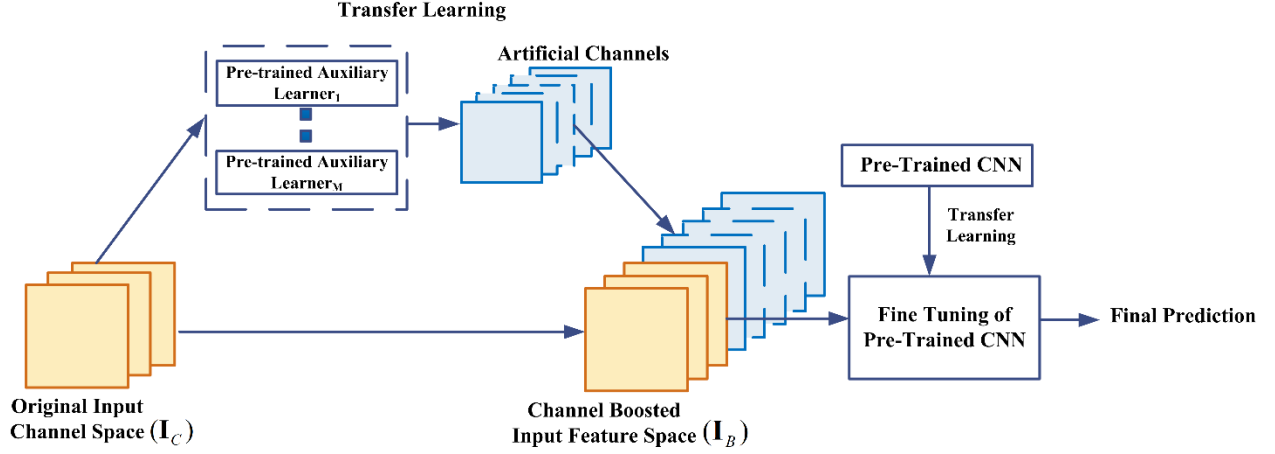
$$\mathbf{F}_l^k = g_c(\mathbf{I}_B, \mathbf{k}_l) \quad (21)$$

In equation (20), \mathbf{I}_C represents the original input channels, where \mathbf{A}_M is an artificial channel generated by M^{th} auxiliary learner. Whereas $g_k(.)$ is used as a combiner function that concatenates the original input channel with auxiliary channels to generate the channel boosted input \mathbf{I}_B for the discriminator. Equation (21) shows the k^{th} resultant feature-map \mathbf{F}_l^k , which is generated by convolving the boosted input \mathbf{I}_B with kernel \mathbf{k}_l of l^{th} layer.

Bengio et al. in 2013 empirically showed that data representation plays an important role in determining the performance of a classifier, as different representations may present different aspects of information [88]. For improving the representational potential of the data, Khan et al. exploited the power of TL and deep generative learners [29], [171], [172]. Generative learners attempt to characterize the data generating distribution during the learning phase. In CB-CNN, AEs are used as generative learners to learn explanatory factors of variation behind the data. The concept of inductive TL is used in a novel way to build a boosted input representation by augmenting learned distribution of the input data with the original channel space (input channels). CB-CNN encodes channel-boosting phase into a generic block, which is inserted at the start of a deep network. CB-CNN provides the concept that TL can be used at both generation and discrimination stages. The significance of the study is that multi-deep learners are used where generative learning models are used as auxiliary learners that enhance the representational capacity of deep CNN based discriminator. Although the potential of the channel boosting was only evaluated by inserting boosting block at the start, however, Khan et al. suggested that this idea can be extended by providing auxiliary channels at any layer in the deep architecture. CB-CNN has also been evaluated on medical image dataset, where it has shown improved results as shown in Table 4.

Table 4 Results of CNN and CB-CNN on mitosis dataset.

CNN Architecture	F-score
26 layers deep CNN	0.47
26 layers deep CB-CNN	0.53
VGG	0.55
CB-VGG	0.71
ResNet	0.44
CB-ResNet	0.54

**Fig. 11** Basic architecture of CB-CNN showing the deep auxiliary learners for creating artificial channels.

4.7 Attention based CNNs

Different levels of abstractions have an important role in defining discrimination power of the NN. In addition to learning of multiple hierarchies of abstractions, focusing on features relevant to the context also play a significant role in image localization and recognition. In human visual system, this phenomenon is referred as attention. Humans view the scene in a succession of partial glimpses and pay attention to context-relevant parts. This process not only serves to focus selected region but also deduces different interpretations of objects at that location and thus helps in capturing visual structure in a better way. A more or less similar kind of interpretability is added into RNN and LSTM [156], [157]. RNN and LSTM networks exploit attention module for

generation of sequential data and the new samples are weighted based on their occurrence in previous iterations. The concept of attention was incorporated into CNN, by various researchers to improve representation and overcome the computational limits. This idea of attention also helps in making CNN intelligent enough to recognize objects even from cluttered backgrounds and complex scenes.

4.7.1 Residual Attention Neural Network

Wang et al. proposed a Residual Attention Network (RAN) to improve feature representation of the network [41]. The motivation behind the incorporation of attention in CNN was to make network capable of learning object aware features. RAN is a feed forward CNN, which was built by stacking residual blocks with attention module. Attention module is branched off into trunk and mask branches that adopt bottom-up top-down learning strategy. The assembly of two different learning strategies into the attention module enables fast feed-forward processing and top-down attention feedback in a single feed-forward process. Bottom-up feed-forward structure produces low-resolution feature-maps with strong semantic information. Whereas, top-down architecture produces dense features in order to make an inference of each pixel. In the previously proposed studies, a top-down bottom-up learning strategy was used by Restricted Boltzmann Machines [173]. Similarly, Goh et al. exploited the top-down attention mechanism as a regularizing factor in Deep Boltzmann Machine during the reconstruction phase of the training. Top-down learning strategy globally optimizes network in such a way that gradually output the maps to input during the learning process [85], [173], [174]. Attention module in RAN generates object aware soft mask $g_{sm}(\cdot)$ at each layer for input feature-map \mathbf{F}_l^K [175]. Soft mask, $g_{sm}(\cdot)$ assigns attention towards object using equation (22) by recalibrating trunk branch $g_m(\mathbf{F}_l^K)$ output and thus, behaves like a control gate for every neuron output.

$$g_{am}(\mathbf{F}_l^K) = g_{sm}(\mathbf{F}_l^K) \cdot g_m(\mathbf{F}_l^K) \quad (22)$$

In one of the previous studies, Transformation network [176], [177] also exploited the idea of attention in a simple way by incorporating it with convolution block, but the main problem was that attention modules in Transformation network are fixed and cannot adapt to changing circumstances. RAN was made efficient towards recognition of cluttered, complex, and noisy images by stacking multiple attention modules. Hierarchical organization of RAN endowed the ability to adaptively assign weight to each feature-map based on their relevance in

the layers [41]. Learning of deep hierarchical structure was supported through residual units. Moreover, three different levels of attention: mixed, channel, and spatial attention were incorporated thus, leveraging the capability to capture object-aware features at different levels [41].

4.7.2 Convolutional Block Attention Module

The significance of attention mechanism and feature-map exploitation is validated through RAN and SE-Network [41], [116]. In this regard, Woo et al. came up with new attention based CNN; named as Convolutional Block Attention Module (CBAM) [40]. CBAM is simple in design and similar to SE-Network. SE-Network only considers the contribution of feature-maps in image classification, but it ignores the spatial locality of the object in images. Spatial location of the object has an important role in object detection. CBAM infers attention maps sequentially by first applying feature-map (channel) attention and then spatial attention, to find the refined feature-maps. In literature, generally, 1x1 convolution and pooling operations are used for spatial attention. Woo et al. showed that pooling of features along spatial axis generates an efficient feature descriptor. CBAM concatenates average pooling operation with max pooling, which generates a strong spatial attention map. Likewise, feature-map statistics were modeled using a combination of max pooling and global average pooling operation. Woo et al. showed that max pooling can provide the clue about distinctive object features, whereas use of global average pooling returns suboptimal inference of feature-map attention. Exploitation of both average pooling and max-pooling improves representational power of the network. These refined feature-maps not only focus on the important part but also increase the representational power of the selected feature-maps. Woo et al. empirically showed that formulation of 3D attention map via serial learning process helps in reduction of the parameters as well as computational cost. Due to the simplicity of CBAM, it can be integrated easily with any CNN architecture.

4.7.3 Concurrent Spatial and Channel Excitation Mechanism

In 2018, Roy et al. extended the work of Hu et al. by incorporating the effect of spatial information in combination with feature-map (channel) information to make it applicable to segmentation tasks [116], [117]. They introduced three different modules: (i) squeezing spatially and exciting feature-map information (cSE), (ii) squeezing feature-map and exciting spatial information (sSE) and (iii) concurrent squeeze and excitation of spatial and feature-map

information (scSE). In this work, AE based convolutional NN was used for segmentation, whereas proposed modules were inserted after the encoder and decoder layers. In cSE module, the same concept as that of SE-block is exploited. In this module, scaling factor is derived based on the combination of feature-maps used for object detection. As spatial information has an important role in segmentation, therefore in sSE module, spatial locality has been given more importance than feature-map information. For this purpose, different combinations of feature-maps are selected and exploited spatially to use them for segmentation. In the last module; scSE, attention to each feature-map is assigned by deriving scaling factor both from spatial and feature-map information and thus to selectively highlight the object specific feature-maps [117].

5 Applications of CNNs

CNNs have been successfully applied to different ML related tasks; namely object detection, recognition, classification, regression, segmentation, etc., [178]–[180]. However, CNN generally needs a large amount of data for learning. All of the aforementioned areas in which CNN has shown tremendous success have relatively abundant labeled data, such as traffic sign recognition, segmentation of medical images, and the detection of faces, text, pedestrians, and human in natural images. Some of the interesting applications of CNN are discussed below.

5.1 CNN based computer vision and related applications

Computer vision (CV) focuses to develop an artificial system that can process visual data including images and videos and can effectively understand and extract useful information from it. CV encompasses a number of application areas such as face recognition, pose estimation, activity recognition, etc.

Face recognition is one of the difficult tasks in CV. Face recognition systems have to cope with variations such as caused by illumination, change in pose, and different facial expressions. Farfade et al. (2015) proposed deep CNN for detecting faces from different poses as well as from occluded faces [181]. In another work, Zhang et al. performed face detection using a new type of multitask cascaded CNN [182]. Zhang’s technique showed good results, when compared against state-of-the-art techniques [183]–[185].

Human pose estimation is one of the challenging tasks related to CV because of the high variability in body pose. Li et al. (2014) proposed a heterogeneous deep CNN based pose

estimation related technique [186]. In Li's technique, empirical results have shown that the hidden neurons are able to learn the localized part of the body. Similarly, another cascade based CNN technique is proposed by Bulat et al. [187]. In their cascaded architecture, first heat maps are detected, whereas, in the second phase, regression is performed on the detected heat maps.

Action recognition is one of the important areas of activity recognition. The difficulties in developing an action recognition system are to solve the translations and distortions of features in different patterns, which belong to the same action class. Earlier approaches involved the construction of motion history images, use of Hidden Markov Models, action sketch generation, etc. Recently, Wang et al. [188] proposed a three dimensional CNN architecture in combination with LSTM for recognizing different actions from video frames. Experimental results have shown that Wang's technique outperforms other activity recognition based techniques [189]–[193]. Similarly, another three dimensional CNN based action recognition system is proposed by Ji et al. [194]. In Ji's work, three-dimensional CNN is used to extract features from multiple channels of input frames. The final action recognition based model is developed on combined extracted feature space. The proposed three dimensional CNN model is trained in a supervised way and is able to perform activity recognition in real world applications.

5.2 CNN based natural language processing

Natural Language Processing (NLP) converts language into a presentation that can easily be exploited by any computer. Although, RNNs are very suitable for NLP applications, however, CNN have also been utilized in NLP based applications such as language modeling, and analysis, etc. Especially, language modeling or sentence molding has taken a twist after the introduction of CNN as a new representation learning algorithm. Sentence modeling is performed to know semantics of the sentences and thus offer new and appealing applications according to customer requirements. Traditional methods of information retrieval analyze data, based on words or features, but ignore the core of the sentence. Kalchbrenner et al. (2014) proposed a dynamic CNN and dynamic *k-max* pooling during training. This approach finds the relations between words without taking into account any external source like parser or vocabulary [195]. In a similar way, Collobert et al. proposed CNN based architecture that can perform various MLP related tasks at the same time like chunking, language modeling, recognizing name-entity, and role modeling related to semantics [196]. In another work, Hu et al. proposed a generic CNN

based architecture that performs matching between two sentences and thus can be applied to different languages [197].

5.3 CNN based object detection and segmentation

Object detection focuses on identifying different objects in images. Recently, R-CNN has been widely used for object detection. Ren et al. (2015) proposed an improvement over R-CNN named as fast R-CNN for object detection [107]. In their work, fully connected convolutional neural network is used to extract feature space that can simultaneously detect boundary and score of object located at different positions. Similarly, Dai et al. (2016) proposed region-based object detection using fully connected CNN [198]. In Dai's work, results are reported on the PASCAL VOC image dataset. Another object detection technique is reported by Gidaris et al. [199], which is based on multi-region based deep CNN that helps to learn the semantic aware features. In Gidaris's approach, objects are detected with high accuracy on PASCAL VOC 2007 and 2012 dataset. Recently, autoencoder based CNN architectures have shown success in segmentation tasks. In this regard, various interesting CNN architectures have been reported for both semantic and instance based segmentation tasks such as FCNN, SegNet, Mask R-CNN, U-Net etc., [109], [200]–[202].

5.4 CNN based image classification

CNN has been widely used for image classification [203]–[205]. One of the major applications of CNN is in medical images especially, for diagnosis of cancer using histopathological images [206]. Recently, Spanhol et al. (2016) used CNN for the diagnosis of breast cancer images and results are compared against a network trained on a dataset containing handcrafted descriptors [207], [208]. Another recently proposed CNN based technique for breast cancer diagnosis is developed by Wahab et al. [209]. In Wahab's work, two phases are involved. In the first phase, hard non-mitosis examples are identified. In the second phase, data augmentation is performed to cope with the class skewness problem. Similarly, Ciresan et al. [210] used German benchmark dataset related to traffic sign signal. They designed CNN based architecture that performed traffic sign classification related task with good recognition rate.

5.5 CNN based speech recognition

Deep CNN is mostly considered as the best option to deal with image processing applications, however, recent studies have shown that it also performs well on speech recognition tasks. Hamid et al. reported a CNN based speaker independent speech recognition system [211]. Experimental results showed ten percent reduction in error rate in comparison to the earlier reported methods [212], [213]. In another work, various CNN architectures, which are either based on the full or limited number of weight sharing within the convolution layer, are explored [214]. Furthermore, the performance of a CNN is also evaluated after the initialization of whole network using pre-training phase [212]. Experimental results showed that almost all of the explored architectures yield good performance on phone and vocabulary recognition related tasks. Nowadays, utilization of CNNs for speech emotion recognition is also gaining attention. Huang et al. used CNN in combination with LSTM for recognizing emotions of speech. In Huang's approach, CNN was trained both on verbal and nonverbal segments of speech and CNN learned features were used by LSTM for recognizing speech emotions [215].

5.6 CNN based video processing

In video processing techniques, temporal and spatial information from videos is exploited. CNN has been used by many researchers for the purpose of solving various video processing related problems [188], [216]–[219]. Tong et al. proposed CNN based short boundary detection system. In Tong's approach, TAGs are generated using CNN [216]. During the experiment, merging of TAGs against one shot is performed in order to annotate video against that shot. Similarly, Wang et al. used 3-D CNN along with LSTM for the purpose of recognizing action within the video [188]. In another technique, Frizzi et al. used CNN for detecting smoke and fire within the video [217]. In Frizzi's approach, CNN architecture not only extracts salient features, but also performs classification task. In the field of action recognition, gathering of spatial and temporal information is considered as a tedious task. In order to overcome the deficiencies of traditional feature descriptors, Tian et al. [218] proposed a three stream based structure, which is capable of extracting spatial-temporal features along with short and long term motion within the video. Similarly, in another technique, CNN in combination with bi-directional LSTM is used for recognizing action from video [219]. Their approach comprises of two phases. In the first phase,

features are extracted from the sixth frame of the videos. In second phase, sequential information between features of the frame is exploited using bi-directional LSTM framework.

5.7 CNN for low resolution images

In the field of ML, different researchers have used CNN based image enhancement techniques for the purpose of enhancing the resolution of the images [220]–[222]. Peng et al. used deep CNN based approach, which categorizes the objects in images having low resolution [220]. Similarly, Chevalier et al. introduced LR-CNN for low-resolution image classification [221]. Another, deep learning based technique is reported by Kawashima et al., in which convolutional layers along with a layer of LSTM is used to recognize action from thermal images of low resolution [222].

5.8 CNN for resource limited systems

Despite of CNN's high computational cost, it has been successfully utilized in developing different ML based embedded systems [223]–[225]. Lee et al. developed number plate recognition system, which is capable of quickly recognizing the number on the license plate [223]. In Lee's technique, the deep learning based embedded recognition system comprises of simple AlexNet architecture. In order to address power efficiency and portability for embedded platforms, Bettoni et al. implemented CNN on FPGA platform [224]. In another technique, FPGA embedded platform is used for efficiently performing different CNN based ML tasks [225]. Similarly, resource limited CNN architectures such as MobileNet, ShuffleNet, ANT Nets, etc. are highly applicable for mobile devices [123]–[125]. Shakeel et al. developed a real time based driver drowsiness detection application for smart devices such as android phones. They used MobileNet architecture in combination with SSD to exploit the benefit of lightweight architecture of MobileNet that can be easily deploy on resource constrained hardware and can learn enriched representation from the incoming video [118].

5.9 CNN for 1D-Data

CNN has not only shown good performance on images but also on 1D-data. Use of 1D-CNN as compared to other ML methods is becoming popular because of its good feature extraction

ability. Vinayakumar et al. used 1D-CNN in combination with RNN, LSTM and gated recurrent units for intrusion detection in network traffic [226]. They evaluated performance of the proposed models on KDDCup 99 dataset consisting of network traffic of TCP/IP packets and showed that CNN significantly surpasses the performance of classical ML models. Abdeljaber et al. showed that 1D-CNN can be used for real-time structural damage detection problem [227]. They developed an end-to-end system that can automatically extract damage-sensitive features from accelerated signals for the detection purpose. Similarly, Yildirim, et al. showed the successful use of CNN for 1D biomedical dataset. Yildirim et al. developed 16 layers deep 1D-CNN based application for mobile devices and cloud based environment for detecting cardiac irregularity in ECG signals. They achieved 91.33% accuracy on MIT-BIH Arrhythmia database [228].

6 CNN challenges

Deep CNNs have achieved good performance on data that either is of the time series nature or follows a grid like topology. However, there are also some other challenges, where deep CNN architectures have been put to tasks. Major challenges associated with different CNN architectures are mentioned in Table 5a-g. Interesting discussions are made by the different researchers related to the performance of CNN on different ML tasks. Some of the challenges faced during the training of deep CNN models are given below:

- Deep CNNs are generally like a black box and thus may lack in interpretation and explanation. Therefore, sometimes it is difficult to verify them.
- Szegedy et al. showed that training of CNN on noisy image data can cause an increase of misclassification error [229]. The addition of the small quantity of random noise in the input image is capable to fool the network in such a way that the model will classify the original and its slightly perturbed version differently.
- Each layer of CNN automatically tries to extract better and problem specific features related to the task. However, for some tasks, it is important to know the nature of features extracted by the deep CNNs before classification. The idea of feature visualization in CNNs can help in this direction. Similarly, Hinton reported that lower layers should handover their knowledge only to the relevant neurons of the next layer. In this regard, Hinton proposed an interesting Capsule Network approach [230], [231].

- Deep CNNs are based on supervised learning mechanism, and therefore, availability of a large and annotated data is required for its proper learning. In contrast, humans have the ability to learn and generalize from a few examples.
- Hyper-parameter selection highly influences the performance of CNN. A little change in the hyper-parameter values can affect the overall performance of a CNN. That is why careful selection of hyper-parameters is a major design issue that needs to be addressed through some suitable optimization strategy.
- Efficient training of CNN demands powerful hardware resources such as GPUs. However, it is still needed to efficiently employ CNNs in embedded and smart devices. A few applications of deep learning in embedded systems are wound intensity correction, law enforcement in smart cities, etc., [232]–[234].
- In vision related tasks, one shortcoming of CNN is that it is generally, unable to show good performance, when used to estimate the pose, orientation, and location of an object. In 2012, AlexNet solved this problem to some extent by introducing the concept of data augmentation. Data augmentation can help CNN in learning diverse internal representations, which ultimately may lead to improved performance.

7 Future directions

The exploitation of different innovative ideas in CNN architectural design have changed the direction of research, especially in image processing and CV. Good performance of CNN on grid like topological data presents it as a powerful representational model for image data. Architectural design of CNN is a promising research field and in future, it is likely to be one of the most widely used AI techniques.

- Ensemble learning is one of the prospective areas of research in CNNs [235], [236]. The combination of multiple and diverse architectures can aid the model in improving generalization and robustness on diverse categories of images by extracting different levels of semantic representations. Similarly, concepts such as batch normalization, dropout, and new activation functions are also worth mentioning.
- The potential of a CNN as a generative learner is exploited in image segmentation tasks, where it has shown good results [237]. The exploitation of generative learning capabilities of CNNs at feature extraction stages can boost the representational power of

the model. Similarly, new paradigms are needed that can enhance the learning capacity of CNN by incorporating informative feature-maps that can be learnt using auxiliary learners at the intermediate stages of CNN [39].

- In human visual system, attention is one of the important mechanisms in capturing information from images. Attention mechanism operates in such a way that it not only extracts the essential information from image, but also stores its contextual relation with other components of image [238]. In future, research may be carried out in the direction that preserves the spatial relevance of objects along with their discriminating features at later stages of learning.
- The learning capacity of CNN is generally enhanced by increasing the size of the network and it can be done in reasonable time with the help of the current advanced hardware technology such as Nvidia DGX-2 supercomputer. However, the training of deep and high capacity architectures is still a significant overhead on memory usage and computational resources [239]–[241]. Consequently, we still require a lot of improvements in hardware technology that can accelerate research in CNNs. The main concern with CNNs is the run-time applicability. Moreover, use of CNN is hindered in small hardwares, especially in mobile devices because of its high computational cost. In this regard, different hardware accelerators are needed for reducing both execution time and power consumption [242]. Some of the very interesting accelerators are already proposed. For example, Application Specific Integrated Circuits, FPGA, and Eyeriss are well known [243]. Moreover, different operations have been performed to minimize the hardware resources in terms of chip area and energy requirement, by reducing floating point precision of operands and ternary quantization, or minimizing the number of matrix operations. Now it is also time to redirect research towards hardware-oriented approximation models [244].
- Deep CNN has a large number of hyper-parameters such as activation function, kernel size, number of neurons per layers, and arrangement of layers, etc. The selection of hyper-parameters and the evaluation time of a deep network, make parameter tuning quite a difficult job. Hyper-parameter tuning is a tedious and intuition driven task, which cannot be defined via explicit formulation. In this regard, Genetic algorithms can also be used to automatically optimize the hyper-parameters by performing search both in a random fashion as well as by directing search by utilizing previous results [245]–[247].

- In order to overcome hardware limitations, the concept of pipeline parallelism can be exploited to scale up deep CNN training. Google group has proposed a distributed ML library; GPipe [248] that offers model parallelism option for training. In future, the concept of pipelining can be used to accelerate the training of large models and to scale the performance without tuning hyper-parameters.
- In future, it is expected that the potential of cloud based platforms will be exploited for the development of computationally intensive CNN applications [249], [250]. Deep and wide CNNs present a critical challenge in implementing and executing them on resource-limited devices. Cloud computing not only allows dealing with huge amount of data but, also leverages a benefit of high computational efficiency at a negligible cost. World leading companies such as Amazon, Microsoft, Google, and IBM offer the public cloud computing facilities at high scalability, speed and flexibility to train resource hungry CNN architectures. Moreover, cloud environment makes it easy to configure libraries both for researchers and new practitioners.
- CNN are mostly used for image processing applications and therefore, the implementation of the state-of-the-art CNN architectures on sequential data requires the conversion of 1D-data into 2D-data. Due to the good feature extraction ability and efficient computations with limited number of parameters, the trend of using 1D-CNNs is being promoted for sequential data [226], [251].
- Recently high energy physicists at CERN have also been utilizing the learning capability of CNN for the analysis of particle collisions. It is expected that the use of ML and specifically that of deep CNN in high energy physics will grow [251], [252].

8 Conclusion

CNN has made remarkable progress, especially in image processing and vision related tasks and has thus revived the interest of scientists in ANNs. In this context, several research works have been carried out to improve the CNN's performance on such tasks. The advancements in CNNs can be categorized in different ways including activation, loss function, optimization, regularization, learning algorithms, and innovations in architecture. This paper reviews advancements in the CNN architectures, especially, based on the design patterns of the processing units, and thus has proposed the taxonomy for recent CNN architectures. In addition

to categorization of CNNs into different classes, this paper also covers the history of CNNs, its applications, challenges, and future directions.

Learning capacity of CNN is significantly improved over the years by exploiting depth and other structural modifications. It is observed in recent literature that the main boost in CNN performance has been achieved by replacing the conventional layer structure with blocks. Nowadays, one of the paradigms of research in CNN architectures is the development of new and effective block architectures. The role of these blocks in a network is that of an auxiliary learner, which by either exploiting spatial or feature-map information or boosting of input channels improves the overall performance. These blocks play a significant role in boosting of CNN performance by making problem aware learning. Moreover, block based architecture of CNN encourages learning in a modular fashion and thereby, making architecture more simple and understandable. The concept of block being a structural unit is going to persist and further enhance CNN performance. Additionally, the idea of attention and exploitation of channel information in addition to spatial information is expected to gain more importance.

Acknowledgments

The authors would like to thank Pattern Recognition lab at DCIS, and PIEAS for providing them computational facilities. The authors express their gratitude to M. Waleed Khan of PIEAS for the detailed discussion related to the Mathematical description of the different CNN architectures.

References

- [1] O. Chapelle, “Support vector machines for image classification,” *Stage deuxième année magistère d’informatique l’École Norm. Supérieure Lyon*, vol. 10, no. 5, pp. 1055–1064, 1998.
- [2] D. G. Lowe, “Object recognition from local scale-invariant features,” *Proc. Seventh IEEE Int. Conf. Comput. Vis.*, pp. 1150–1157 vol.2, 1999.
- [3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-Up Robust Features (SURF),” *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, 2008.
- [4] N. Dalal and W. Triggs, “Histograms of Oriented Gradients for Human Detection,” *2005 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. CVPR05*, vol. 1, no. 3, pp. 886–893, 2004.
- [5] T. Ojala, M. Pietikäinen, and D. Harwood, “A comparative study of texture measures with classification based on feature distributions,” *Pattern Recognit.*, vol. 29, no. 1, pp. 51–59, 1996.
- [6] M. Heikkilä, M. Pietikäinen, and C. Schmid, “Description of interest regions with local binary patterns,” *Pattern Recognit.*, vol. 42, no. 3, pp. 425–436, 2009.

- [7] Y. LeCun *et al.*, “Backpropagation applied to handwritten zip code recognition,” *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.
- [8] X. Liu, Z. Deng, and Y. Yang, “Recent progress in semantic image segmentation,” *Artif. Intell. Rev.*, vol. 52, no. 2, pp. 1089–1106, 2019.
- [9] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, “Deep neural networks segment neuronal membranes in electron microscopy images,” in *Advances in neural information processing systems*, 2012, pp. 2843–2851.
- [10] L. Deng, D. Yu, and B. — Delft, “Deep Learning: Methods and Applications Foundations and Trends R in Signal Processing,” *Signal Processing*, vol. 7, pp. 3–4, 2013.
- [11] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition? BT - Computer Vision, 2009 IEEE 12th International Conference on,” *Comput. Vision, 2009 ...*, pp. 2146–2153, 2009.
- [12] Y. LeCun, K. Kavukcuoglu, C. C. Farabet, and others, “Convolutional networks and applications in vision,” in *ISCAS*, 2010, vol. 2010, pp. 253–256.
- [13] D. Scherer, A. Müller, and S. Behnke, “Evaluation of pooling operations in convolutional architectures for object recognition,” in *Artificial Neural Networks--ICANN 2010*, Springer, 2010, pp. 92–101.
- [14] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, “Deep learning applications and challenges in big data analytics,” *J. Big Data*, vol. 2, no. 1, pp. 1–21, 2015.
- [15] Q. Abbas, M. E. A. Ibrahim, and M. A. Jaffar, “A comprehensive review of recent advances on deep vision systems,” *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 39–76, 2019.
- [16] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, “Deep learning for visual understanding: A review,” *Neurocomputing*, vol. 187, pp. 27–48, 2016.
- [17] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “A survey of deep neural network architectures and their applications,” *Neurocomputing*, vol. 234, no. October 2016, pp. 11–26, 2017.
- [18] Ian Goodfellow, Y. Bengio, and A. Courville, “Deep learning,” *Nat. Methods*, vol. 13, no. 1, p. 35, 2017.
- [19] D. H. Hubel and T. N. Wiesel, “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex,” *J. Physiol.*, vol. 160, no. 1, pp. 106–154, Jan. 1962.
- [20] D. H. Hubel and T. N. Wiesel, “Receptive fields and functional architecture of monkey striate cortex,” *J. Physiol.*, vol. 195, no. 1, pp. 215–243, Mar. 1968.
- [21] M. N. U. Laskar, L. G. S. Giraldo, and O. Schwartz, “Correspondence of Deep Neural Networks and the Brain for Visual Textures,” pp. 1–17, 2018.
- [22] K. Grill-Spector, K. S. Weiner, J. Gomez, A. Stigliani, and V. S. Natu, “The functional neuroanatomy of face perception: From brain measurements to deep neural networks,” *Interface Focus*, vol. 8, no. 4, p. 20180013, Aug. 2018.
- [23] Y. Bengio, “Learning Deep Architectures for AI,” *Found. Trends® Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.
- [24] T. Ojala, M. PeitiKainen, and T. Maenpää, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Trans. Pattern Anal. and Mach. Intell.*, 2002.
- [25] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
- [26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2012.

- [27] A. S. Qureshi and A. Khan, "Adaptive Transfer Learning in Deep Neural Networks: Wind Power Prediction using Knowledge Transfer from Region to Region and Between Different Task Domains," *arXiv Prepr. arXiv1810.12611*, 2018.
- [28] A. S. Qureshi, A. Khan, A. Zameer, and A. Usman, "Wind power prediction using deep neural network based meta regression and transfer learning," *Appl. Soft Comput. J.*, vol. 58, pp. 742–755, 2017.
- [29] Qiang Yang, S. J. Pan, Q. Yang, and Q. Y. Fellow, "A Survey on Transfer Learning," *IEEE Trans. Knowl. Data Eng.*, vol. 1, no. 10, pp. 1–15, 2008.
- [30] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," *arXiv Prepr. arXiv1311.2901v3*, vol. 30, no. 1–2, pp. 225–231, 2013.
- [31] K. Simonyan and A. Zisserman, "VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION," *ICLR*, vol. 75, no. 6, pp. 398–406, 2015.
- [32] C. Szegedy *et al.*, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, vol. 57, no. 3, pp. 1–9.
- [33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Multimed. Tools Appl.*, vol. 77, no. 9, pp. 10437–10453, Dec. 2015.
- [34] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated Residual Transformations for Deep Neural Networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5987–5995.
- [35] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," *arXiv Prepr. arXiv1602.07261v2*, vol. 131, no. 2, pp. 262–263, 2016.
- [36] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," *Proceedings Br. Mach. Vis. Conf. 2016*, pp. 87.1–87.12, May 2016.
- [37] D. Han, J. Kim, and J. Kim, "Deep Pyramidal Residual Networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, vol. 2017-Janua, pp. 6307–6315.
- [38] X. Zhang, Z. Li, C. C. Loy, and D. Lin, "PolyNet: A pursuit of structural diversity in very deep networks," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, no. 1, pp. 3900–3908, 2017.
- [39] A. Khan, A. Sohail, and A. Ali, "A New Channel Boosted Convolutional Neural Network using Transfer Learning," *arXiv Prepr. arXiv1804.08528*, Apr. 2018.
- [40] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," 2018.
- [41] F. Wang *et al.*, "Residual Attention Network for Image Classification," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, vol. 2017-Janua, no. 1, pp. 6450–6458.
- [42] J. Gu *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354–377, 2018.
- [43] S. Srinivas, R. K. Sarvadevabhatla, K. R. Mopuri, N. Prabhu, S. S. S. Kruthiventi, and R. V. Babu, "A Taxonomy of Deep Convolutional Neural Nets for Computer Vision," *Front. Robot. AI*, vol. 2, no. January, pp. 1–13, 2016.
- [44] Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, and B. Yu, "Recent advances in convolutional neural network acceleration," *Neurocomputing*, vol. 323, pp. 37–51, 2019.
- [45] J. Bouvrie, "1 Introduction Notes on Convolutional Neural Networks," 2006.
- [46] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

- [47] C.-Y. Lee, P. W. Gallagher, and Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in *Artificial Intelligence and Statistics*, 2016, pp. 464–472.
- [48] F. J. Huang, Y.-L. Boureau, Y. LeCun, and others, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, 2007, pp. 1–8.
- [49] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, 2015.
- [50] T. Wang, D. J. D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," *ICPR, Int. Conf. Pattern Recognit.*, no. May, pp. 3304–3308, 2012.
- [51] Y. Boureau, "Icml2010B.Pdf," 2009.
- [52] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical Evaluation of Rectified Activations in Convolutional Network," *J. Foot Ankle Res.*, vol. 1, no. S1, p. O22, May 2015.
- [53] Y. LeCun, "Efficient BackProp," *J. Exp. Psychol. Gen.*, vol. 136, no. 1, pp. 23–42, 2007.
- [54] P. Ramachandran, B. Zoph, and Q. V. Le, "Swish: a Self-Gated Activation Function," *arXiv*, 2017.
- [55] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation Functions: Comparison of trends in Practice and Research for Deep Learning," *arXiv Prepr. arXiv1811.03378*, Nov. 2018.
- [56] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Uncertainty, Fuzziness Knowledge-Based Syst.*, vol. 6, no. 02, pp. 107–116, 1998.
- [57] D. Misra, "Mish: A Self Regularized Non-Monotonic Neural Activation Function," no. 1, 2019.
- [58] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," 2015.
- [59] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," pp. 1–18, 2012.
- [60] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfittin," *J. Mach. Learn. Res.*, vol. 1, no. 60, p. 11, 2014.
- [61] M. Lin, Q. Chen, and S. Yan, "Network In Network," pp. 1–10, 2013.
- [62] W. Rawat and Z. Wang, "Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review," vol. 61, no. 5–6, pp. 1120–1132, 2016.
- [63] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *J. Physiol.*, 1959.
- [64] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in *Competition and cooperation in neural nets*, Springer, 1982, pp. 267–285.
- [65] K. Fukushima, "Neocognitron: A hierarchical neural network capable of visual pattern recognition," *Neural networks*, vol. 1, no. 2, pp. 119–130, 1988.
- [66] S. Linnainmaa, "The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors," *Master's Thesis (in Finnish), Univ. Helsinki*, pp. 6–7, 1970.

- [67] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv Prepr. arXiv1502.01710*, 2015.
- [68] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [69] Y. LeCun *et al.*, "Learning algorithms for classification: A comparison on handwritten digit recognition," *Neural networks Stat. Mech. Perspect.*, vol. 261, p. 276, 1995.
- [70] J. Schmidhuber, "New millennium AI and the convergence of history," in *Challenges for computational intelligence*, Springer, 2007, pp. 15–35.
- [71] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*, 1998, pp. 137–142.
- [72] D. Decoste and B. Schölkopf, "Training invariant support vector machines," *Mach. Learn.*, vol. 46, no. 1–3, pp. 161–190, 2002.
- [73] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: benchmarking of state-of-the-art techniques," *Pattern Recognit.*, vol. 36, no. 10, pp. 2271–2285, 2003.
- [74] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *null*, 2003, p. 958.
- [75] L. Deng, "The MNIST database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, 2012.
- [76] K. Chellapilla, S. Puri, and P. Simard, "High performance convolutional neural networks for document processing," in *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.
- [77] A. Abdulkader, "Two-tier approach for Arabic offline handwriting recognition," in *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.
- [78] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep , Big , Simple Neural Nets for Handwritten," *Neural Comput.*, vol. 22, no. 12, pp. 3207–3220, 2010.
- [79] A. Frome *et al.*, "Large-scale privacy protection in Google Street View," in *Proceedings of the IEEE International Conference on Computer Vision*, 2009.
- [80] C. Garcia and M. Delakis, "Convolutional face finder: A neural architecture for fast and robust face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2004.
- [81] M. Matsugu, K. Mori, M. Ishii, and Y. Mitarai, "Convolutional spiking neural network model for robust face detection," in *Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, 2002, vol. 2, pp. 660–664.
- [82] Y.-N. Chen, C.-C. Han, C.-T. Wang, B.-S. Jeng, and K.-C. Fan, "The application of a convolution neural network on face and license plate detection," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, 2006, vol. 3, pp. 552–555.
- [83] B. Fasel, "Facial expression analysis using shape and motion information extracted by convolutional neural networks," in *Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on*, 2002, pp. 607–616.
- [84] A. Khan, A. Zameer, T. Jamal, and A. Raza, "Deep Belief Networks Based Feature Generation and Regression for Predicting Wind Power," *arXiv Prepr. arXiv1807.11682*, 2018.
- [85] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
- [86] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

- [87] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in neural information processing systems*, 2007, no. 1, pp. 153–160.
- [88] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.
- [89] A. Giusti, D. C. Cireşan, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Fast image scanning with deep max-pooling convolutional neural networks," in *2013 IEEE International Conference on Image Processing*, 2013, pp. 4034–4038.
- [90] G. Nguyen *et al.*, "Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey," *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 77–124, 2019.
- [91] D. Strigl, K. Kofler, and S. Podlipnig, "Performance and scalability of GPU-based convolutional neural networks," in *Parallel, Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on*, 2010, pp. 317–324.
- [92] K.-S. Oh and K. Jung, "GPU implementation of neural networks," *Pattern Recognit.*, vol. 37, no. 6, pp. 1311–1314, 2004.
- [93] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "High-Performance Neural Networks for Visual Object Classification," *arXiv Prepr. arXiv1102.0183*, Feb. 2011.
- [94] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with CUDA," in *ACM SIGGRAPH 2008 classes on - SIGGRAPH '08*, 2008, p. 1.
- [95] E. Lindholm, J. Nickolls, S. Oberman, and J. Montrym, "NVIDIA Tesla: A Unified Graphics and Computing Architecture," *IEEE Micro*, vol. 28, no. 2, pp. 39–55, Mar. 2008.
- [96] D. C. Cireşan, D. C. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [97] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, 2015.
- [98] A. Berg, J. Deng, and L. Fei-Fei, "Large scale visual recognition challenge 2010." 2010.
- [99] T. Sinha, B. Verma, and A. Haidar, "Optimization of convolutional neural network parameters for image classification," *2017 IEEE Symp. Ser. Comput. Intell. SSCI 2017 - Proc.*, vol. 2018-Janua, pp. 1–7, 2018.
- [100] M. Amer and T. Maul, "A review of modularization techniques in artificial neural networks," *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 527–561, 2019.
- [101] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 2261–2269, 2017.
- [102] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway Networks," 2015.
- [103] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra R-CNN: Towards Balanced Learning for Object Detection," 2020.
- [104] Z. Cai and N. Vasconcelos, "Cascade R-CNN: High Quality Object Detection and Instance Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.
- [105] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017.
- [106] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and Tell: Lessons Learned from the 2015 MSCOCO Image Captioning Challenge," *IEEE Trans. Pattern Anal. Mach.*

- Intell.*, 2017.
- [107] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” pp. 1–9, Jun. 2015.
 - [108] R. Girshick, “Fast R-CNN,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015.
 - [109] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.
 - [110] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.
 - [111] E. Lv, X. Wang, Y. Cheng, and Q. Yu, “Deep ensemble network based on multi-path fusion,” *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 151–168, 2019.
 - [112] Y. Yamada, M. Iwamura, and K. Kise, “Deep pyramidal residual networks with separated stochastic depth,” *arXiv Prepr. arXiv1612.01230*, 2016.
 - [113] J. Kuen, X. Kong, G. Wang, Y.-P. Tan, and A. Group, “DelugeNets: Deep Networks with Efficient and Flexible Cross-layer Information Inflows,” in *Computer Vision Workshop (ICCVW), 2017 IEEE International Conference on*, 2017, pp. 958–966.
 - [114] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, “Deep networks with stochastic depth,” in *European Conference on Computer Vision*, 2016, pp. 646–661.
 - [115] S. Targ, D. Almeida, and K. Lyman, “Resnet in Resnet: generalizing residual architectures,” *arXiv Prepr. arXiv1603.08029*, 2016.
 - [116] J. Hu, L. Shen, and G. Sun, “Squeeze-and-Excitation Networks,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.
 - [117] A. G. Roy, N. Navab, and C. Wachinger, “Concurrent spatial and channel ‘squeeze & excitation’ in fully convolutional networks,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11070 LNCS, pp. 421–429, 2018.
 - [118] M. F. Shakeel, N. A. Bajwa, A. M. Anwaar, A. Sohail, A. Khan, and Haroon-ur-Rashid, “Detecting Driver Drowsiness in Real Time Through Deep Learning Based Object Detection,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019.
 - [119] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, “Quantized convolutional neural networks for mobile devices,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016.
 - [120] W. Chen, J. T. Wilson, S. Tyree, K. Q. Weinberger, and Y. Chen, “Compressing neural networks with the hashing trick,” in *32nd International Conference on Machine Learning, ICML 2015*, 2015.
 - [121] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding,” in *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2016.
 - [122] N. Frosst and G. rey Hinton, “Distilling a neural network into a soft decision tree,” in *CEUR Workshop Proceedings*, 2018.
 - [123] X. Zhang, X. Zhou, M. Lin, and J. Sun, “ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.
 - [124] Y. Xiong, H. J. Kim, and V. Hedau, “ANTNets: Mobile Convolutional Neural Networks for Resource Efficient Image Classification,” 2019.

- [125] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” Apr. 2017.
- [126] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-Decem, pp. 2818–2826.
- [127] J. Kuen, X. Kong, G. Wang, and Y. P. Tan, “DelugeNets: Deep Networks with Efficient and Flexible Cross-Layer Information Inflows,” *Proc. - 2017 IEEE Int. Conf. Comput. Vis. Work. ICCVW 2017*, vol. 2018-Janua, pp. 958–966, 2018.
- [128] G. Larsson, M. Maire, and G. Shakhnarovich, “Fractalnet: Ultra-deep neural networks without residuals,” *arXiv Prepr. arXiv1605.07648*, pp. 1–11, May 2016.
- [129] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” *arXiv Prepr.*, pp. 1610–2357, 2017.
- [130] Y. Hu, G. Wen, M. Luo, D. Dai, J. Ma, and Z. Yu, “Competitive Inner-Imaging Squeeze and Excitation for Residual Network,” Jul. 2018.
- [131] H.-C. C. Shin *et al.*, “Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning,” *IEEE Trans. Med. Imaging*, vol. 35, no. 5, pp. 1285–1298, May 2016.
- [132] M. Kafi, M. Maleki, and N. Davoodian, “Functional histology of the ovarian follicles as determined by follicular fluid concentrations of steroids and IGF-1 in *Camelus dromedarius*,” *Res. Vet. Sci.*, vol. 99, pp. 37–40, 2015.
- [133] S. Potluri, A. Fasih, L. K. Vutukuru, F. Al Machot, and K. Kyamakya, “CNN based high performance computing for real time image processing on GPU,” in *Proceedings of the Joint INDS’11 & ISTET’11*, 2011, pp. 1–7.
- [134] M. W. Gardner and S. R. Dorling, “Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences,” *Atmos. Environ.*, vol. 32, no. 14–15, pp. 2627–2636, 1998.
- [135] G. E. Dahl, T. N. Sainath, and G. E. Hinton, “Improving deep neural networks for LVCSR using rectified linear units and dropout,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2013, pp. 8609–8613.
- [136] V. Nair and G. E. Hinton, “Rectified linear units improve Restricted Boltzmann machines,” in *ICML 2010 - Proceedings, 27th International Conference on Machine Learning*, 2010.
- [137] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, “Visualizing higher-layer features of a deep network,” *Univ. Montr.*, vol. 1341, no. 3, p. 1, 2009.
- [138] Q. V. Le *et al.*, “Building high-level features using large scale unsupervised learning,” *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, pp. 8595–8598, Dec. 2011.
- [139] F. Grün, C. Rupprecht, N. Navab, and F. Tombari, “A Taxonomy and Library for Visualizing Learned Features in Convolutional Neural Networks,” vol. 48, 2016.
- [140] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps,” pp. 1–8, 2013.
- [141] Y. N. Dauphin, H. De Vries, and Y. Bengio, “Equilibrated adaptive learning rates for non-convex optimization,” *Adv. Neural Inf. Process. Syst.*, vol. 2015-Janua, pp. 1504–1512, 2015.
- [142] Y. Bengio, “Deep learning of representations: Looking forward,” in *International Conference on Statistical Language and Speech Processing*, 2013, pp. 1–37.

- [143] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, “On the number of linear regions of deep neural networks,” in *Advances in neural information processing systems*, 2014, pp. 2924–2932.
- [144] B. Csáji, “Approximation with artificial neural networks,” *MSc. thesis*, p. 45, 2001.
- [145] O. Delalleau and Y. Bengio, “Shallow vs. deep sum-product networks,” in *Advances in Neural Information Processing Systems*, 2011, pp. 666–674.
- [146] H. Wang and B. Raj, “On the Origin of Deep Learning,” pp. 1–72, 2017.
- [147] Q. Nguyen, M. Mukkamala, and M. Hein, “Neural Networks Should Be Wide Enough to Learn Disconnected Decision Regions,” *arXiv Prepr. arXiv1803.00094*, 2018.
- [148] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, “Deep Networks with Stochastic Depth BT - Computer Vision – ECCV 2016,” in *European Conference on Computer Vision*, 2016, pp. 646–661.
- [149] A. Morar, F. Moldoveanu, and E. Gröller, “Image segmentation based on active contours without edges,” *Proc. - 2012 IEEE 8th Int. Conf. Intell. Comput. Commun. Process. ICCP 2012*, pp. 213–220, 2012.
- [150] T.-Y. Lin *et al.*, “Microsoft coco: Common objects in context,” in *European conference on computer vision*, 2014, pp. 740–755.
- [151] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, 2016.
- [152] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, “Language modeling with gated convolutional networks,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp. 933–941.
- [153] R. Pascanu, T. Mikolov, and Y. Bengio, “Understanding the exploding gradient problem,” *CoRR*, *abs/1211.5063*, 2012.
- [154] T. Tong, G. Li, X. Liu, and Q. Gao, “Image super-resolution using dense skip connections,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 4809–4817.
- [155] X. Mao, C. Shen, and Y.-B. Yang, “Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections,” in *Advances in neural information processing systems*, 2016, pp. 2802–2810.
- [156] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, “Recurrent neural network based language model,” in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.
- [157] M. Sundermeyer, R. Schlüter, and H. Ney, “LSTM neural networks for language modeling,” in *Thirteenth annual conference of the international speech communication association*, 2012.
- [158] K. Kawaguchi, J. Huang, and L. P. Kaelbling, “Effect of depth and width on local minima in deep learning,” *Neural Comput.*, vol. 31, no. 7, pp. 1462–1498, Nov. 2019.
- [159] B. Hanin and M. Sellke, “Approximating Continuous Functions by ReLU Nets of Minimal Width,” *arXiv Prepr. arXiv1710.11278*, 2017.
- [160] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, “The expressive power of neural networks: A view from the width,” in *Advances in Neural Information Processing Systems*, 2017, pp. 6231–6239.
- [161] R. K. Srivastava, K. Greff, and J. Schmidhuber, “Training very deep networks,” in *Advances in Neural Information Processing Systems*, 2015.
- [162] Y. Wang, L. Wang, H. Wang, and P. Li, “End-to-End Image Super-Resolution via Deep

- and Shallow Convolutional Networks,” *IEEE Access*, vol. 7, pp. 31959–31970, Jul. 2019.
- [163] A. Sharma and S. K. Muttou, “Spatial Image Steganalysis Based on ResNeXt,” *2018 IEEE 18th Int. Conf. Commun. Technol.*, pp. 1213–1216, 2018.
- [164] W. Han, R. Feng, L. Wang, and L. Gao, “Adaptive Spatial-Scale-Aware Deep Convolutional Neural Network for High-Resolution Remote Sensing Imagery Scene Classification,” *IGARSS 2018 - 2018 IEEE Int. Geosci. Remote Sens. Symp.*, pp. 4736–4739, 2018.
- [165] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.
- [166] H. Zheng, J. Fu, T. Mei, and J. Luo, “Learning multi-attention convolutional neural network for fine-grained image recognition,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 5219–5227.
- [167] P. Dollár, Z. Tu, P. Perona, and S. Belongie, “Integral channel features,” 2009.
- [168] M. N. Do and M. Vetterli, “The contourlet transform: an efficient directional multiresolution image representation,” *IEEE Trans. image Process.*, vol. 14, no. 12, pp. 2091–2106, 2005.
- [169] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014, vol. 111, no. 1, pp. 1717–1724.
- [170] J. Yang, W. Xiong, S. Li, and C. Xu, “Learning structured and non-redundant representations with deep neural networks,” *Pattern Recognit.*, vol. 86, pp. 224–235, 2019.
- [171] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.
- [172] P. Hamel and D. Eck, “Learning Features from Music Audio with Deep Belief Networks,” in *ISMIR*, 2010, vol. 10, pp. 339–344.
- [173] R. Salakhutdinov and H. Larochelle, “Efficient learning of deep Boltzmann machines,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 693–700.
- [174] H. Goh, N. Thome, M. Cord, and J.-H. Lim, “Top-down regularization of deep belief networks,” *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2013.
- [175] K. Xu *et al.*, “Show, attend and tell: Neural image caption generation with visual attention,” in *International conference on machine learning*, 2015, pp. 2048–2057.
- [176] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, “Spatial Transformer Networks,” pp. 1–15, 2015.
- [177] X. Li, L. Bing, W. Lam, and B. Shi, “Transformation Networks for Target-Oriented Sentiment Classification,” pp. 946–956, May 2018.
- [178] N. Wahab, A. Khan, and Y. S. Lee, “Transfer learning based deep CNN for segmentation and detection of mitoses in breast cancer histopathological images,” *Microscopy*, vol. 68, no. 3, pp. 216–233, 2019.
- [179] N. Chouhan and A. Khan, “Network anomaly detection using channel boosted and residual learning based deep convolutional neural network,” *Appl. Soft Comput.*, p. 105612, 2019.
- [180] Z. Batmaz, A. Yurekli, A. Bilge, and C. Kaleli, “A review on deep learning for

- recommender systems: challenges and remedies,” *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 1–37, 2019.
- [181] S. S. Farfade, M. J. Saberian, and L.-J. Li, “Multi-view Face Detection Using Deep Convolutional Neural Networks,” in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval - ICMR '15*, 2015, pp. 643–650.
 - [182] K. Zhang, Z. Zhang, Z. Li, Y. Q.-I. S. P. Letters, and undefined 2016, “Joint face detection and alignment using multitask cascaded convolutional networks,” *Ieeexplore.Ieee.Org*, vol. 23, no. 10, pp. 1499–1503, 2016.
 - [183] S. Yang, P. Luo, C.-C. Loy, and X. Tang, “From facial parts responses to face detection: A deep learning approach,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3676–3684.
 - [184] R. Ranjan, V. M. Patel, and R. Chellappa, “A deep pyramid deformable part model for face detection,” *arXiv Prepr. arXiv1508.04389*, 2015.
 - [185] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, “A convolutional neural network cascade for face detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5325–5334.
 - [186] S. Li, Z.-Q. Liu, and A. B. Chan, “Heterogeneous Multi-task Learning for Human Pose Estimation with Deep Convolutional Neural Network,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 488–495.
 - [187] A. Bulat and G. Tzimiropoulos, “Human Pose Estimation via Convolutional Part Heatmap Regression BT - Computer Vision – ECCV 2016,” 2016, pp. 717–732.
 - [188] X. Wang, L. Gao, J. Song, and H. Shen, “Beyond Frame-level CNN: Saliency-Aware 3-D CNN With LSTM for Video Action Recognition,” *IEEE Signal Process. Lett.*, vol. 24, no. 4, pp. 510–514, Apr. 2017.
 - [189] H. Wang and C. Schmid, “Action recognition with improved trajectories,” in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 3551–3558.
 - [190] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3d convolutional networks,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.
 - [191] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi, “Human action recognition using factorized spatio-temporal convolutional networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4597–4605.
 - [192] K. Simonyan and A. Zisserman, “Two-stream convolutional networks for action recognition in videos,” in *Advances in neural information processing systems*, 2014, pp. 568–576.
 - [193] J. Donahue *et al.*, “Long-term recurrent convolutional networks for visual recognition and description,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.
 - [194] S. Ji, M. Yang, K. Yu, and W. Xu, “3D convolutional neural networks for human action recognition,” *ICML, Int. Conf. Mach. Learn.*, vol. 35, no. 1, pp. 221–31, 2010.
 - [195] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, “A convolutional neural network for modelling sentences,” *arXiv Prepr. arXiv1404.2188*, 2014.
 - [196] R. Collobert and J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning,” in *Proceedings of the 25th international conference on Machine learning*, 2008, vol. 18, no. 6, pp. 160–167.
 - [197] B. Hu, Z. Lu, H. Li, and Q. Chen, “Topic modeling for named entity queries,” in *Proceedings of the 20th ACM international conference on Information and knowledge*

- management - CIKM '11*, 2011, vol. 116, no. c, p. 2009.
- [198] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object Detection via Region-based Fully Convolutional Networks," no. Nips, 2016.
 - [199] S. Gidaris and N. Komodakis, "Object detection via a multi-region and semantic segmentation-aware U model," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2015 Inter, no. 1, pp. 1134–1142, 2015.
 - [200] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015.
 - [201] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017.
 - [202] Y. Zhang, Z. Qiu, T. Yao, D. Liu, and T. Mei, "Fully Convolutional Adaptation Networks for Semantic Segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.
 - [203] P. Sermanet, S. Chintala, and Y. LeCun, "Convolutional Neural Networks Applied to House Numbers Digit Classification," *Proc. 21st Int. Conf. Pattern Recognit.*, no. Icpr, pp. 3288–3291, 2012.
 - [204] G. Levi and T. Hassner, "Sicherheit und Medien," *Sicherheit und Medien*, 2009.
 - [205] Z. M. Long, S. Q. Guo, G. J. Chen, and B. L. Yin, "Modeling and simulation for the articulated robotic arm test system of the combination drive," *2011 Int. Conf. Mechatronics Mater. Eng. ICMME 2011*, vol. 151, no. i, pp. 480–483, 2012.
 - [206] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Mitosis Detection in Breast Cancer Histology Images with Deep Neural Networks BT - Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013," in *Proceedings MICCAI*, 2013, pp. 411–418.
 - [207] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte, "Breast cancer histopathological image classification using Convolutional Neural Networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016, vol. 29, no. 1, pp. 2560–2567.
 - [208] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte, "A dataset for breast cancer histopathological image classification," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 7, pp. 1455–1462, 2016.
 - [209] N. Wahab, A. Khan, and Y. S. Lee, "Two-phase deep convolutional neural network for reducing class skewness in histopathological images based breast cancer detection," *Comput. Biol. Med.*, vol. 85, no. March, pp. 86–97, Jun. 2017.
 - [210] D. Cireşan, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," *Neural Networks*, vol. 32, pp. 333–338, Aug. 2012.
 - [211] O. Abdel-Hamid, A. R. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, no. May, pp. 4277–4280, 2012.
 - [212] A. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 20, no. 1, pp. 14–22, 2012.
 - [213] G. Dahl, A. Mohamed, and G. E. Hinton, "Phone recognition with the mean-covariance restricted Boltzmann machine," in *Advances in neural information processing systems*, 2010, pp. 469–477.
 - [214] O. Abdel-Hamid, L. Deng, and D. Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition," in *Interspeech*, 2013, vol. 2013, pp.

- 1173–1175.
- [215] K. Y. Huang, C. H. Wu, Q. B. Hong, M. H. Su, and Y. H. Chen, “Speech Emotion Recognition Using Deep Neural Network Considering Verbal and Nonverbal Speech Sounds,” in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2019.
 - [216] W. Tong, L. Song, X. Yang, H. Qu, and R. Xie, “CNN-based shot boundary detection and video annotation,” in *2015 IEEE international symposium on broadband multimedia systems and broadcasting*, 2015, pp. 1–5.
 - [217] S. Frizzi, R. Kaabi, M. Bouchouicha, J.-M. Ginoux, E. Moreau, and F. Fnaiech, “Convolutional neural network for video fire and smoke detection,” in *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 877–882.
 - [218] Y. Shi, Y. Tian, Y. Wang, and T. Huang, “Sequential deep trajectory descriptor for action recognition with three-stream CNN,” *IEEE Trans. Multimed.*, vol. 19, no. 7, pp. 1510–1520, 2017.
 - [219] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik, “Action recognition in video sequences using deep bi-directional LSTM with CNN features,” *IEEE Access*, vol. 6, pp. 1155–1166, 2017.
 - [220] X. Peng, J. Hoffman, S. X. Yu, and K. Saenko, “Fine-to-coarse knowledge transfer for low-res image classification,” in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3683–3687.
 - [221] M. Chevalier, N. Thome, M. Cord, J. Fournier, G. Henaff, and E. Dusch, “LR-CNN for fine-grained classification with varying resolution,” in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, no. 1, pp. 3101–3105.
 - [222] T. Kawashima, Y. Kawanishi, I. Ide, H. Murase, R. Corporate, and O. Corporation, “Action Recognition from Extremely Low-Resolution Thermal Image Sequence,” vol. 2.
 - [223] S. Lee, K. Son, H. Kim, and J. Park, “Car Plate Recognition Based on CNN Using Embedded System with GPU,” pp. 239–241, 2017.
 - [224] M. Bettoni, G. Urgese, Y. Kobayashi, E. Macii, and A. Acquaviva, “A Convolutional Neural Network Fully Implemented on FPGA for Embedded Platforms,” no. 3, pp. 49–52, 2017.
 - [225] W. Xie, C. Zhang, Y. Zhang, C. Hu, H. Jiang, and Z. Wang, “An Energy-Efficient FPGA-Based Embedded System for CNN Application,” in *2018 IEEE International Conference on Electron Devices and Solid State Circuits (EDSSC)*, 2018, pp. 1–2.
 - [226] R. Vinayakumar, K. P. Soman, and P. Poornachandran, “Applying convolutional neural network for network intrusion detection,” in *2017 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*, 2017.
 - [227] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman, “Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks,” *J. Sound Vib.*, 2017.
 - [228] Ö. Yıldırım, P. Pławiak, R. S. Tan, and U. R. Acharya, “Arrhythmia detection using deep convolutional neural network with long duration ECG signals,” *Comput. Biol. Med.*, 2018.
 - [229] C. Szegedy *et al.*, “Intriguing properties of neural networks,” Dec. 2013.
 - [230] G. E. Hinton, S. Sabour, and N. Frosst, “Matrix capsules with EM routing,” 2018.
 - [231] H. de Vries, R. Memisevic, and A. Courville, “Deep learning vector quantization,” in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2016.
 - [232] G. Hinton *et al.*, “Deep neural networks for acoustic modeling in speech recognition: The

- shared views of four research groups,” *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, 2012.
- [233] H. Lu *et al.*, “Wound intensity correction and segmentation with convolutional neural networks,” *Concurr. Comput. Pract. Exp.*, vol. 29, no. 6, p. e3927, 2017.
 - [234] G. E. Hinton, A. Krizhevsky, and S. D. Wang, “Transforming auto-encoders,” in *International Conference on Artificial Neural Networks*, 2011, pp. 44–51.
 - [235] D. Marmanis, J. D. Wegner, S. Galliani, K. Schindler, M. Datcu, and U. Stilla, “Semantic segmentation of aerial images with an ensemble of CNNs,” *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 3, p. 473, 2016.
 - [236] U. Ahmed, A. Khan, S. H. Khan, A. Basit, I. U. Haq, and Y. S. Lee, “Transfer Learning and Meta Classification Based Deep Churn Prediction System for Telecom Industry,” pp. 1–10, 2019.
 - [237] M. Kahng, N. Thorat, D. H. P. Chau, F. B. Viégas, and M. Wattenberg, “GAN Lab: Understanding Complex Deep Generative Models using Interactive Visual Experimentation,” *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 1, pp. 310–320, 2019.
 - [238] A. K. Bhunia, A. Konwer, A. K. Bhunia, A. Bhowmick, P. P. Roy, and U. Pal, “Script identification in natural scene image and video frames using an attention based Convolutional-LSTM network,” *Pattern Recognit.*, vol. 85, pp. 172–184, 2019.
 - [239] D. Justus, J. Brennan, S. Bonner, and A. S. McGough, “Predicting the Computational Cost of Deep Learning Models,” in *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, 2019.
 - [240] V. Sze, Y. H. Chen, T. J. Yang, and J. S. Emer, “Efficient Processing of Deep Neural Networks: A Tutorial and Survey,” *Proceedings of the IEEE*. 2017.
 - [241] G. Lacey, G. W. Taylor, and S. Areibi, “Deep Learning on FPGAs: Past, Present, and Future,” 2016.
 - [242] X. Geng, J. Lin, B. Zhao, Z. Wang, M. M. S. Aly, and V. Chandrasekhar, “Hardware-aware Exponential Approximation for Deep Neural Network,” 2018.
 - [243] B. Moons and M. Verhelst, “An energy-efficient precision-scalable ConvNet processor in 40-nm CMOS,” *IEEE J. Solid-State Circuits*, vol. 52, no. 4, pp. 903–914, 2017.
 - [244] X. Geng, J. Lin, B. Zhao, A. Kong, M. M. S. Aly, and V. Chandrasekhar, “Hardware-aware Softmax Approximation for Deep Neural Networks,” 2018.
 - [245] M. Suganuma, S. Shirakawa, and T. Nagao, “A genetic programming approach to designing convolutional neural network architectures,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2017, vol. 2018-July, pp. 497–504.
 - [246] S. R. Young *et al.*, “Optimizing deep learning hyper-parameters through an evolutionary algorithm,” in *Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments*, 2015, p. 4.
 - [247] A. Khan, A. S. Qureshi, M. Hussain, M. Y. Hamza, and others, “A Recent Survey on the Applications of Genetic Programming in Image Processing,” *arXiv Prepr. arXiv1901.07387*, 2019.
 - [248] Y. Huang *et al.*, “GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism,” vol. 2014, 2018.
 - [249] E. Akar, O. Marques, W. A. Andrews, and B. Furht, “Cloud-Based Skin Lesion Diagnosis System Using Convolutional Neural Networks,” in *Intelligent Computing-Proceedings of the Computing Conference*, 2019, pp. 982–1000.
 - [250] M. Stefanini, R. Lancellotti, L. Baraldi, and S. Calderara, “A Deep-learning-based approach to VM behavior Identification in Cloud Systems,” in *Proceedings of the 9th*

- International Conference on Cloud Computing and Services Science*, 2019, pp. 308–315.
- [251] C. F. Madrazo, I. Heredia, L. Lloret, and J. Marco de Lucas, “Application of a Convolutional Neural Network for image classification for the analysis of collisions in High Energy Physics,” *EPJ Web Conf.*, 2019.
- [252] A. Aurisano *et al.*, “A convolutional neural network neutrino event classifier,” *J. Instrum.*, 2016.