Artificial Intelligence Review (2024) 57:99
https://doi.org/10.1007/s10462-024-10721-6

A review of convolutional neural networks in computer vision

Xia Zhao[1] · Limin Wang[1] · Yufei Zhang[2] · Xuming Han[3] · Muhammet Deveci[4,5,6] · Milan Parmar[7]

Accepted: 4 February 2024 / Published online: 23 March 2024

Abstract
In computer vision, a series of exemplary advances have been made in several areas involving image classification, semantic segmentation, object detection, and image super-resolution reconstruction with the rapid development of deep convolutional neural network (CNN). The CNN has superior features for autonomous learning and expression, and feature extraction from original input data can be realized by means of training CNN models that match practical applications. Due to the rapid progress in deep learning technology, the structure of CNN is becoming more and more complex and diverse. Consequently, it gradually replaces the traditional machine learning methods. This paper presents an elementary understanding of CNN components and their functions, including input layers, convolution layers, pooling layers, activation functions, batch normalization, dropout, fully connected layers, and output layers. On this basis, this paper gives a comprehensive overview of the past and current research status of the applications of CNN models in computer vision fields, e.g., image classification, object detection, and video prediction. In addition, we summarize the challenges and solutions of the deep CNN, and future research directions are also discussed.

Keywords Convolutional neural networks · Computer vision · Status quo review · Deep learning

1 Introduction

Computer vision is gaining popularity as a buzzword in the field of image processing. Human activity recognition (HAR), an established trend with numerous real-life applications including elderly care monitoring, rehabilitation activity tracking, posture correction analysis, and intrusion detection in security, is a prominent area of research in the field of computer vision (Singh and Vishwakarma 2019). Over the years, deep learning advances in computer vision have attracted the attention of many scholars in the field of human action recognition (Vishwakarma and Singh 2019; Singh and Vishwakarma 2021; Dhiman and Vishwakarma 2020). The convolutional neural network (CNN) is

Xia Zhao and Limin Wang have contributed equally to this work.

Extended author information available on the last page of the article used to construct the majority of computer vision algorithms. A convolutional neural network (Li et al. 2021), known for local connectivity of neurons, weight sharing, and down-sampling, is a deep feed-forward multilayered hierarchical network inspired by the receptive field mechanism in biology. As one of the deep learning models, a CNN can also achieve "end-to-end" learning. Through multiple layers of feature transformation, the underlying feature representation of the original data is gradually transformed into a higher-level feature representation, and the processed data is fed into a prediction function to settle the final classification or other tasks. The representation learned by the machine itself can generate good features, avoiding "feature engineering".

In 2006, Hinton et al. proposed several perspectives in their article, which was published in Science (Hinton and Salakhutdinov 2006), including (1) that artificial neural networks with multiple hidden layers have a robust feature learning capability and (2) that the difficulty of training deep neural networks can be greatly reduced by the "layer-by-layer initialization" method. Since then, deep learning has become a hot topic in both academia and industry, and it has made a splash in computer vision, speech recognition, machine translation, and other fields. Meanwhile, another learning boom in artificial neural networks (Yu et al. 2013) has kicked off. As a typical neural network model of deep learning, a CNN has also gained wide attention from all walks of life. One of the most widely concerned is AlexNet (Alom et al. 2018), which won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) in 2012 due to its excellent performance. With the improvement of AlexNet's accuracy on computer vision tasks such as image classification, researchers started to remedy the defects of the network models based on AlexNet in the expectation of further enhancing their performance. Significant advances have been made in model optimization, and some of the most representative neural network models are Visual Geometry Group (VGG) (Sengupta et al. 2019), GoogLeNet (Khan et al. 2019), Residual Network (ResNet) (Wightman et al. 2021), Squeeze and Excitation Network (SENet) (Jin et al. 2022), and MobileNet (Chen et al. 2022). With the development of these network architectures, neural network models tend to be deeper, wider, and more complex. Although this evolution can facilitate the networks to capture better feature representations, there is no guarantee that it can operate efficiently in all cases. Models still suffer from disadvantages such as the fact that the networks are more likely to fall into overfitting, and instead of decreasing, the error rate of the training set increases as the networks become deeper and more complex. To remedy the shortcomings of these models, many scholars have come up with various techniques to optimize the structure of CNN, e.g., network pruning (Yang et al. 2023), knowledge distilling (Guo et al. 2023), and tensor decomposition (Fernandes et al. 2021).

Despite the significant achievements of CNN in computer vision applications such as image classification (Chandra and Bedi 2021), object detection (Ma et al. 2023), speech recognition (Li et al. 2022), sentiment analysis (Chan et al. 2023), and video recognition (Yan et al. 2022), the field continues to face various challenges and opportunities. As computer vision tasks become increasingly complex, there is a pressing need for CNN models and algorithms that offer higher performance and efficiency. Moreover, current research focuses on addressing key issues such as knowledge sharing across different tasks, domain adaptation, and interpretability. Given these things into account, this paper aims to comprehensively summarize and analyze the applications of CNN in computer vision, with a particular emphasis on the latest advancements in tasks including image classification, object detection, and video prediction. The contributions of this survey paper are summarized below:

- A holistic literature review of CNN in computer vision, including image classification, object detection, and video prediction, is presented in this paper.
- A theoretical understanding of the CNN design principles and techniques, such as convolution, filter size, stride, down sampling, optimizer, etc., is explained in detail.
- The image classification and object detection performance obtained using the existing algorithms on the dataset of the domain to which they belong are compared, respectively.
- Classical architectures for deep learning and CNN-based visual models are highlighted.
- The current challenges involved and future research directions for CNN are identified and presented.

The remaining part of the paper proceeds as follows (shown in Fig. 1): Section 2 gives a basic introduction to the elementary components of CNN and their corresponding functions. Sections 3, 4, and 5 summarize the relevant research models and methods in three application directions, namely, image classification, object detection, and video prediction, respectively. In Sects. 6 and 7, through synthesizing the current research status, the issues of CNN are analyzed and summarized. In addition, an outlook on future research trends is provided.

2 Basic CNN components

Although there are numerous variations of CNN models, the overall architecture is essentially the same and adheres to a fixed paradigm, consisting of an input layer, alternate layers of convolution and pooling layers, one or more fully connected layers, activation functions, and an output layer at the end. The first half of the network comprises of a number of convolution and pooling layers stacked alternately to form a feature extractor, through which various operations can be performed to process the raw input data that is preprocessed into a more abstract and higher-level feature representation. Fully connected layers are used in combination with activation functions to execute tasks such as classification or regression on the extracted features. To maximize CNN performance, various regulatory units like batch normalization and dropout are also included in addition to various mapping functions (Bouvrie 2006). Fig. 2 shows various CNN components. The configuration of CNN components is essential to creating new architectures and, ultimately, to obtaining improved performance. It is crucial to comprehend various CNN components and their respective applications in order to learn about the developments in CNN architecture in computer vision (Bhatt et al. 2021). The role of these components in a CNN architecture is covered in brief in this section.

Before being input to CNN, the raw data needs to be preprocessed. The common processing methods include homogenization (Stepanov et al. 2023), normalization (Huang et al. 2023), and principal component analysis (PCA) (Uddin et al. 2021). To achieve homogenization, the average value calculated across the complete training set is subtracted to center each dimension of the input data at zero. Normalization is designed to normalize the data magnitude to the same range. By individually normalizing the input, dimension reduction with PCA can lessen the correlation between several data dimensions.

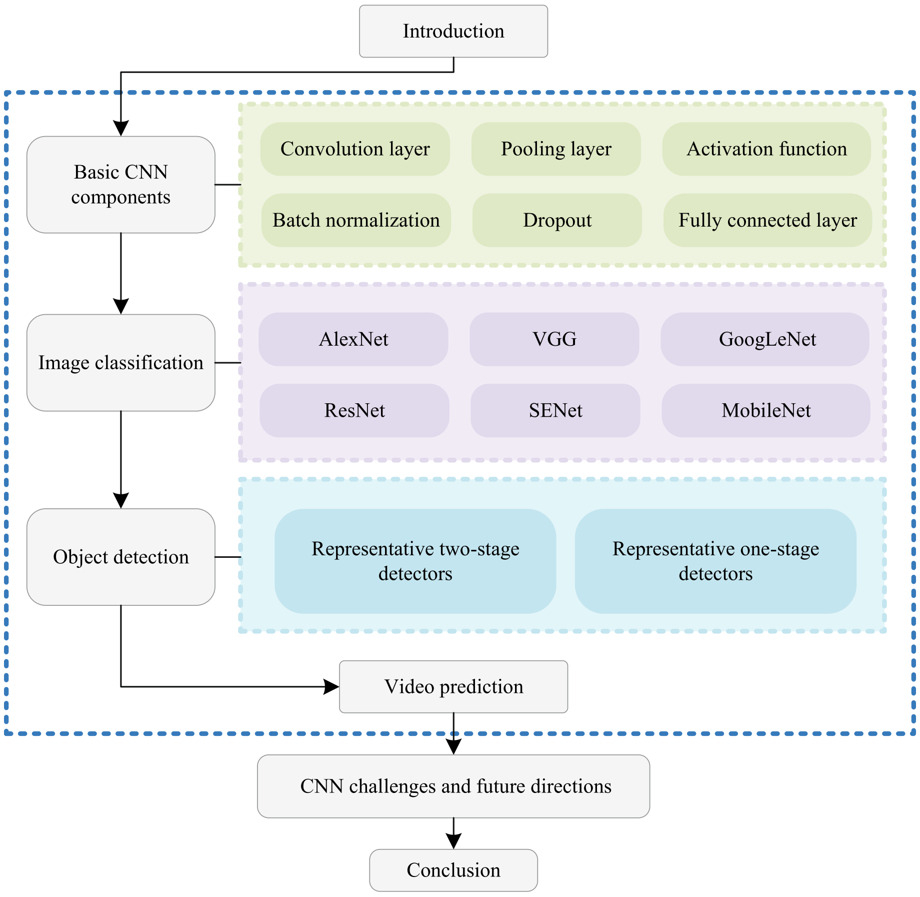


Fig. 1 Layout of the paper illustrating the overall process (This paper is reviewed in the order of introduction, basic CNN components, image classification, object detection, video prediction,CNN challenges and future directions, and conclusion. Among them, the convolution layer, pooling layer, activation function, batch normalization, dropout, and fully connected layer are introduced in the basic CNN compositions. An introduction to image classification includes AlexNet, VGG, GoogLeNet, ResNet, SENet, and MobileNet. We overview object detection according to two-stage and one-stage. Video prediction is a popular area of research in the field of CNN. This part presents the state-of-the-art models in video prediction. The conclusion summarizes the challenges related to CNN and outlines future research directions.)

2.1 Convolution layer

The convolution layer, which may extract various features from different local regions of the input data, is composed of a collection of convolution kernels, with each neuron acting as a kernel. Each convolution kernel has three dimensions: length (L), width (W), and depth (D). In the convolution layer of a CNN, the length and width of the convolution kernel are designed artificially, $L \times W$ is also known as the size of the convolution kernel. Commonly used sizes are 3×3, 5×5, etc. The number of channels, also known as the depth or the number of feature maps, is the number of feature maps output from each layer in the CNN, and the depth of the convolution kernel is the same as the number of

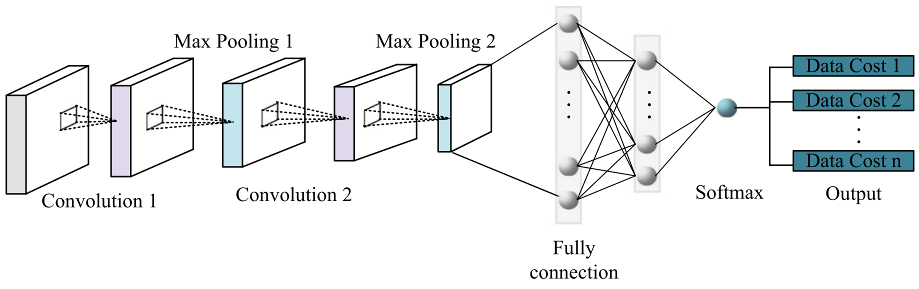


Fig. 2 Structure of CNN (Suppose this is an n-classification problem. The original data is convolved twice (Convolution 1, Convolution 2), pooled twice (Max Pooling 1, Max Pooling 2), and output to the fully connected layer (Fully connection), and finally the Softmax activation function compresses the output vectors of the full connection layer into (0, 1) and outputs them in the output layer. The Data Cost 1 represents the probability of belonging to the n categories; the larger the value, the greater the possibility of belonging to the category.)

sheets of the feature map. The number of channels directly affects the feature extraction ability and computational complexity of the CNN. By increasing the number of channels, the feature extraction ability of CNN can be enhanced, but it also increases the computational complexity. A convolution operation is the process of sliding a convolution kernel (filter) over the input image, multiplying the convolution kernel and the pixel values at the corresponding positions of the input image, and summing them to obtain a feature map. The convolution process is depicted in Fig. 3 using a single-channel original image 5×5 and a convolution kernel 3×3. Each pixel value of the feature map obtained by convolution is obtained by multiplying and summing the corresponding pixel values of the original image covered by the convolution kernel at the corresponding position. In Fig. 3, the -4 on the blue background in the feature map is calculated as follows: $-4 = (-1) \times 1 + 0 \times 0 + 1 \times 7 + (-1) \times 8 + 0 \times 2 + 1 \times 4 + (-1) \times 6 + 0 \times 5 + 1 \times 0$.

By convolving the original image with the filters and applying a nonlinear activation function to obtain new feature mappings, each feature mapping can be used as a class of extracted image features. To extract higher-level and more complete feature representations, the network model can stack multiple convolution layers. Convolutional operation's weight-sharing technique allows multiple sets of features within an image to be retrieved by sliding a kernel with the same set of weights on the image, making it more efficient and effective for CNN parameters than fully connected networks. Furthermore, it also allows

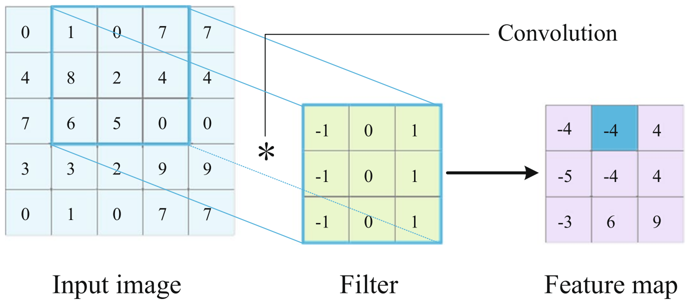


Fig. 3 Convolution procedure

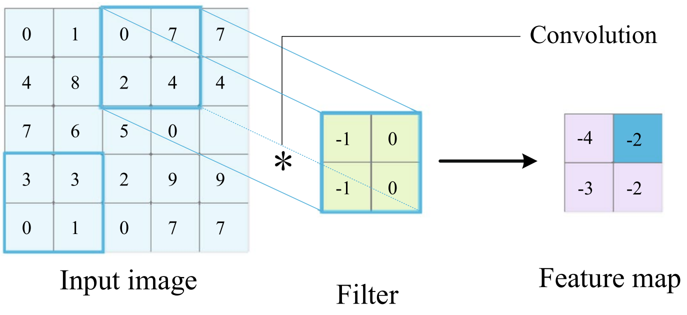


Fig. 4 Convolution procedure (stride=(2,3))

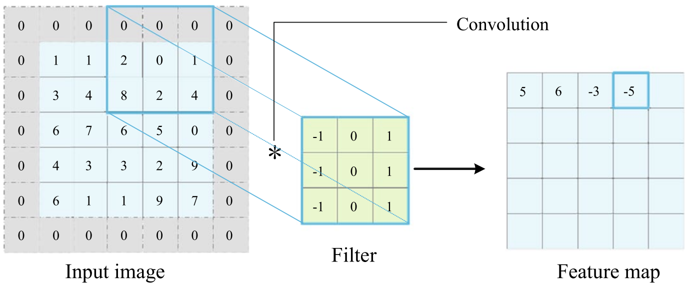


Fig. 5 Padding

the network to have fewer neuron connections and a simpler network architecture, which facilitates the training of the network.

Stride is the number of rows and columns that the convolution kernel slides over the input matrix in order from left to right and top to bottom, starting from the top left of the input matrix. For example, in Fig. 3, the stride is 1 in both the height and width directions. In addition, we can also use a larger stride. Fig. 4 illustrates a convolution operation with a stride of 3 in the vertical direction and 2 in the horizontal direction. At the output of the second element of the first column, the convolution window is slid down 3 rows, and the elements used for the calculation are: $(-1) \times 3 + 0 \times 3 + (-1) \times 0 + 0 \times 1 = -3$. The convolution window slides two columns to the right when the second element of the first row is output. The elements that were used in the calculation are: $(-1) \times 0 + 0 \times 7 + (-1) \times 2 + 0 \times 4 = -2$. Since the input elements cannot fill the convolution kernel window, no result is produced when the convolution window slides two more columns to the right on the input. The output data size, computational complexity, and feature extraction capability can all be impacted by the stride. The output data size reduces and the ability to extract features weakens as the stride increases, but the computation speed increases.

Padding is the process of adding a certain number of pixels to the edges of the input data so that the size of the output data can match the input data. As shown in Fig. 5, it is also known as padding some values on the boundary of the matrix to increase the size of the matrix, usually with 0 or copying the boundary pixels for padding. Padding is frequently used in CNN to prevent feature map sizes from shrinking at each layer. Furthermore,

Fig. 6 Max pooling

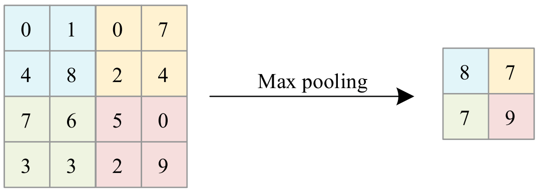
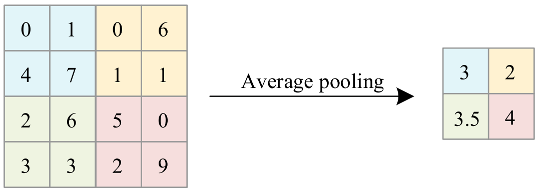


Fig. 7 Average pooling



padding makes it easier for the convolution kernel to learn the information surrounding the input image. For instance, when the $5 \times 5 \times 1$ image is reinforced into a $7 \times 7 \times 1$ and applied to the $3 \times 3 \times 1$ kernel over it, the complex matrix is shown to be of dimensions $5 \times 5 \times 1$. It demonstrates that the dimensions of the input and output images are the same. If the same procedure is done without padding, the output might have a smaller-sized image. Consequently, a $5 \times 5 \times 1$ image will be converted to a $3 \times 3 \times 1$ image (Bhatt et al. 2021).

2.2 Pooling layer

Upon acquiring the feature maps, a pooling (down sampling) layer must be added. The neurons in the pooling layer are connected to the local receptive domains of their input layer, i.e., the convolution layer, and the local receptive domains of different neurons do not overlap. The pooling procedure, like the convolution process, can be thought of as a pooling function without weights, in which the input feature mapping group is divided into many regions and each area is pooled to yield a value as a generalization of this region. Pooling functions that are commonly used are max pooling and average pooling.

For a region, max pooling selects the maximum activity value of all neurons as the representation of this region and extracts the most significant features from the input feature mapping, which is generally used for low-level feature extraction. In the case of max pooling (stride = 2), as shown in Fig. 6, a kernel of size 2×2 is moved across the matrix, and the maximum value is selected and put in the appropriate spot of the output matrix. For example, pooling the four numbers ’0, 1, 4, 8’ in the blue region yields 8, the maximum of these four numbers.

Average pooling takes the arithmetic mean of all elements in the region as the output result of the function, namely, the mean value of the local response of the extracted feature mapping.The average pooling results with filter = 2×2 and stride 2 are shown in Fig. 7. It is evident that the green region’s pooling result is $(2 + 6 + 3 + 3)/4 = 3.5$.

The introduction of a pooling layer not only effectively compresses the amount of data and parameters, reduces the feature map dimension, and minimizes overfitting, but also makes the network invariant to some small local morphological changes while having a larger perceptual field. Applying different pooling techniques also significantly shortens the time needed for model training and improves feature extraction and compression.

2.3 Activation function

An activation function is a different mathematical function that receives the filter's output. It plays an important role in neural networks, which strengthens the representational and learning capabilities of the network. Each layer's input and output in a neural network is a linear summation process, and the output of the next layer simply takes over the linear transformation of the previous layer's input function. On the contrary, with the introduction of the activation function, the neural network can approximate any other nonlinear function, making it applicable to a wider range of nonlinear models. In this section, we will introduce the most classical and widely used activation functions, including Sigmoid, Tanh, Softmax, ReLU, and Leaky ReLU.

The logistic function, also known as the sigmoid, has values between 0 and 1. As can be seen in Fig. 8, the sigmoid can be used to both normalize the output of each neuron and as a model that uses the predicted probabilities as the outputs. This is because the sigmoid maps any received vector to the (0,1) interval. The following is the expression for the sigmoid function.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

Figure 8 shows that the sigmoid gradient is smooth, preventing output values from jumping. Nevertheless, there are numerous issues with using Sigmoid. The next layer's neuron inputs confront bias shift as a result of the non-zero-centered output, which also slows down the gradient descent's convergence and decreases the weight update's efficiency. Secondly, the sigmoid function's rate of change flattens out as it gets closer to 0 and 1, meaning the sigmoid's gradient converges to 0. Neurons with outputs near 0 or 1 do not have their weights updated when the neural network is backpropagated using the sigmoid activation function because their gradients are convergent to 0. Furthermore, the weights of the neurons connected to such neurons are slowly updated and are prone to gradient

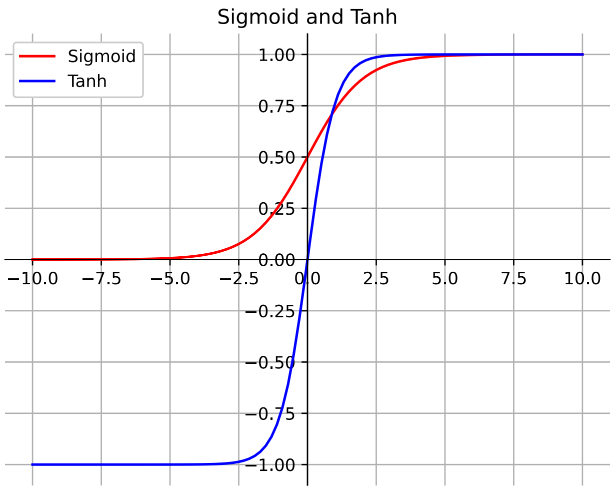


Fig. 8 Function curves of Sigmoid and Tanh

vanishing. Finally, the sigmoid function is an exponential operation, which lengthens the model's computation time.

Tanh, also known as the hyperbolic tangent activation function (HTAF), compresses the received vector into a range of − 1 to 1. Equation (2) and Fig. 8 show the function expression and curve, respectively.

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{2}$$

Figure 8 shows that the Tanh and sigmoid function curves are relatively similar and resemble an S-shaped curve. Furthermore, the Tanh function can be thought of as a zoomed and shifted sigmoid function. Tanh and Sigmoid have the following relationship:

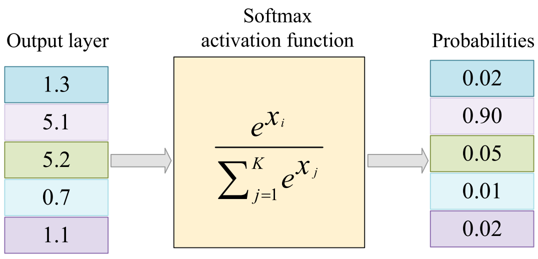
$$\mathrm{Tanh}(x) = 2Sigmoid(2x) - 1 \tag{3}$$

Tanh is used with a higher priority than Sigmoid in practice because it improves on Sigmoid and solves the problem of Sigmoid functions not centering the output at 0. However, like the sigmoid, when the input is large or small, the output is smooth and the gradient is small, which is inconvenient for weight updating.

Softmax is an activation function for multi-classification problems. For any real vector of length K, Softmax activation can compress it into a real vector of length K, with values in the range (0, 1) and vector elements summing to 1. In the K classification task, these values obtained by the activation function can be used to represent the predicted probability of each category, with larger values indicating a higher probability of belonging to that category. As shown in Fig. 9, this is a 5-classification problem. All the output layer vectors (left column) are given a number (right column) within (0, 1) after Softmax, where the probability of the second row is 0.90, indicating that the classification task belongs to the second category. SoftMax is formulated as follows:

$$Soft\max(x) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{4}$$

In contrast to the standard max function, which only returns the maximum value, Softmax ensures that smaller values have smaller probabilities and are not discarded outright. The denominator of the Softmax function combines all the factors of the original output value, which means that the various probabilities obtained by the Softmax function are correlated with each other. When the input is negative, the gradient is zero, which means that the weights for activation in that region will not be updated during backpropagation, resulting in dead neurons that never activate. Furthermore, Softmax has the issue of being non-trivial at zero. Fig. 10 depicts the Softmax function image.


Output layer
1.3
5.1
5.2
0.7
1.1
Softmax activation function
$\frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$
Probabilities
0.02
0.90
0.05
0.01
0.02
Fig. 9 Softmax schematic

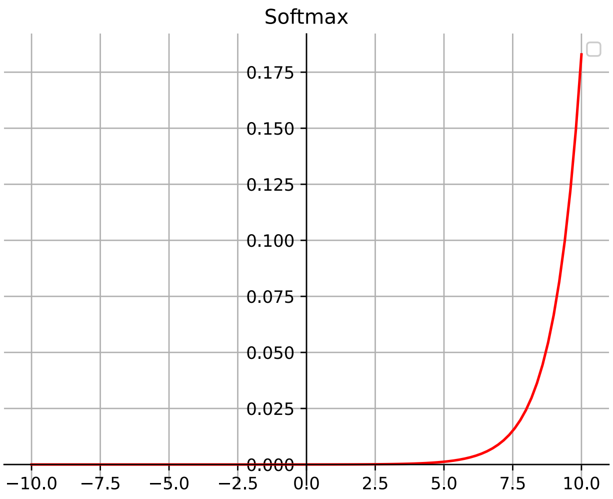


Fig. 10 Function curves of Softmax

Softmax and Sigmoid also have some similarities and differences in some aspects. Softmax can be regarded as an extension of sigmoid, and softmax regression degenerates to sigmoid regression when the number of categories $K = 2$. A sigmoid maps a real value to the interval (0,1) and is used for binary categorization. Softmax puts a K-dimensional vector of real values (**a1, a2, a3, a4....**) into (**b1, b2, b3, b4....**), where **bi** is a constant from 0 to 1. The multi-categorization task can then be performed based on the probability magnitude of **bi**. Although multiple sigmoid can also achieve the effect of multi-categorization by superposition, multi-categorization by softmax regression is mutually exclusive between classes, i.e., an input can only be categorized into one class; multi-categorization by multiple sigmoid regression is performed, and the classes of the output are not mutually exclusive.

ReLU, also known as Rectified Linear Unit, is a segmented linear function, as shown in Fig. 11. The ReLU function is essentially a ramp function with the following formula:

$$f(x) = \max(0, x) \tag{5}$$

To some extent, ReLU compensates for the lack of sigmoid and tanh. When the input is positive, the derivative is 1, which improves the gradient vanishing problem and speeds up gradient descent convergence. Second, because the ReLU function only has linear relationships, it is faster than the sigmoid and tanh functions. However, this activation function suffers from the Dead ReLU problem. (If the input is negative, the gradient will be exactly zero, and the ReLU neurons are more likely to "die" during training.) Similar to Sigmoid, the output of the ReLU function is not zero-centered, which introduces a bias offset to the neural network in the next layer, affecting the efficiency of gradient descent.

To solve the problem of the vanishing gradient in ReLU, when $x < 0$, we use Leaky ReLU, a function that tries to fix the Dead ReLU problem. The function expression is as follows:

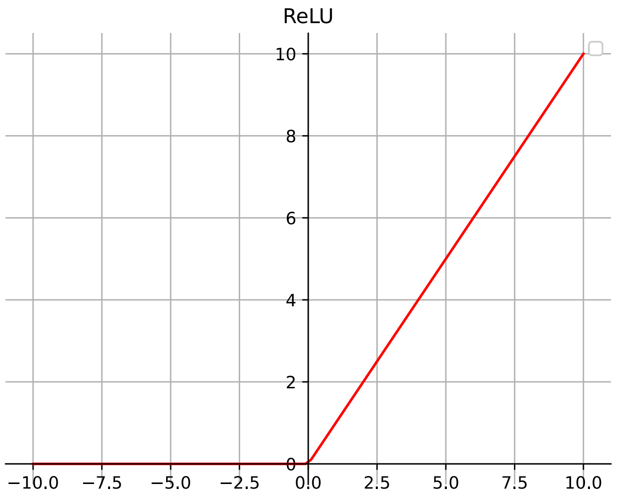


Fig. 11 Function curves of ReLU

$$f(x) = \begin{cases} x, x \geq 0 \\ ax, x < 0 \end{cases} \tag{6}$$

where a is a very tiny value, like 0.01, 0.1, etc. As in Fig. 12, let $a = 0.01$ be displayed here.

Leaky ReLU mitigates the Dead ReLU problem to some extent by giving very small linear components to the negative inputs to adjust for the zero gradients of the negatives, extending the range of ReLU. Although Leaky ReLU has all the features of ReLU, such as being computationally efficient, having fast convergence, and not saturating in positive

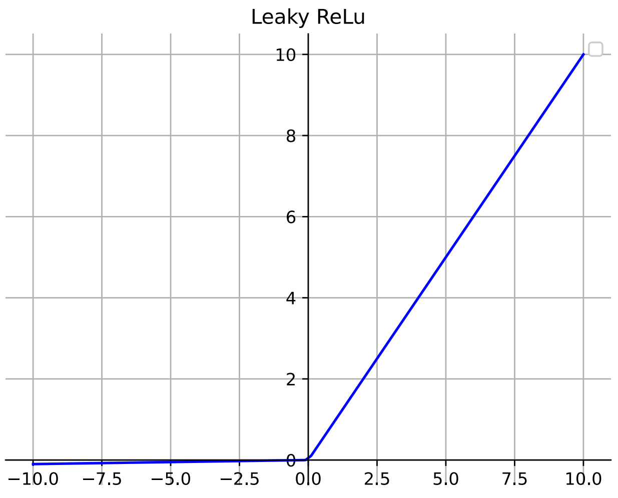


Fig. 12 Function curves of Leaky ReLU

regions, it has not been fully proven in practice that Leaky ReLU is always better than ReLU.

The essence of deep learning lies in continuously updating weights to find values that minimize loss. When dealing with complex tasks, deep networks outperform shallow ones. However, in deep neural networks, gradients are unstable, either vanishing or exploding, caused by the compounding effect of multiplication in gradient backpropagation. For example, the backpropagation (BP) algorithm, based on gradient descent, adjusts parameters in the negative gradient direction of the objective. Gradient calculation involves the derivative of the activation function. If the derivative is greater than 1, as network layers increase, the computed gradient update grows exponentially, leading to a gradient explosion. This results in significant updates to network weights, making the network unstable. If the derivative is less than 1, the gradient update information decays exponentially with increasing layers, causing the vanishing gradient problem. This prevents the model from learning effectively from training data, even with prolonged training.

Choosing the appropriate activation function can effectively alleviate the issues of gradients vanishing and exploding. If the derivative of the activation function is 1, there is no problem of gradients vanishing or exploding, and each layer of the network can update at the same rate. Sigmoid and Tanh are two classic activation functions, but Sigmoid has a drawback: when x is large or small, the derivative is close to 0, and the maximum value of the Sigmoid function's derivative is 0.25. If Sigmoid is used as the activation function, its gradient cannot exceed 0.25. Consequently, gradient vanishing is likely to occur after the chain rule in backpropagation. Similar to Sigmoid, using Tanh as an activation function may still lead to the issue of gradient vanishing; although its derivative is better than Sigmoid, it remains less than 1. Therefore, Sigmoid and Tanh are generally not suitable for neural networks. The derivative of ReLU is constantly 1 in the positive part, so using ReLU as the activation function avoids the problems of gradients vanishing and exploding. By allowing positive gradients to remain unchanged and setting negative values to zero, ReLU ensures that only positive gradients contribute to weight updates, mitigating the problem of gradients vanishing. Additionally, ReLU can prevent gradient explosion by truncating large gradient values. Other activation functions, such as Sigmoid or Tanh, can also partially alleviate the problem of gradient explosion to some extent. The activation function acts as a decision function and aids in the learning of complex patterns. Choosing an appropriate activation function can hasten the learning process. Different activation functions are appropriate for various application scenarios. ReLU and its variants, on the other hand, are preferred because they aid in overcoming the vanishing gradient problem (Nwankpa et al. 2018).

2.4 Batch normalization

Gradient descent is a very versatile optimization algorithm that is well suited to solving a range of problems. The whole idea of gradient descent is to minimize the objective function by iteratively updating the parameters in the opposite direction of the gradient of the objective function. The gradient is the representation of the directional derivative of a function at that point along which the function achieves its maximum value. The gradient descent algorithm is shown in Fig. 13, where a random initial value is chosen, the gradient at that point is calculated, and then the independent variables are updated in the direction of the gradient until the value of the function changes very little or the minimum number of iterations is reached. The formula is as follows:

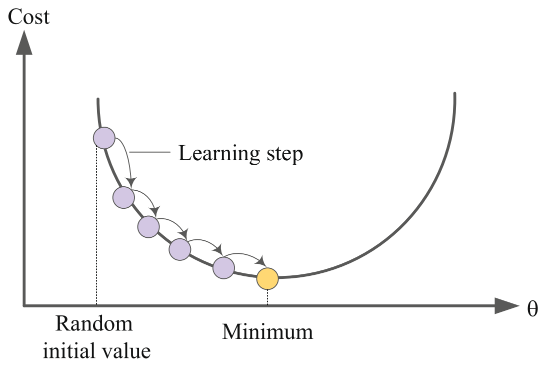


Fig. 13 Gradient descent algorithm

$$\theta = \theta - \alpha \frac{\partial J(\theta)}{\partial \theta} \tag{7}$$

where, θ is the parameter to be solved, α the learning rate represents the learning step for each optimization, and $J(\theta)$ is the objective function.

The learning step is an important parameter of gradient descent that determines just how far to try to advance on the objective function in order to find the minima point. There are two extremes that can occur with the setting of the learning step, as shown in Fig. 14: (a) If the learning step is too small, it will have to go through many iterations before the algorithm can converge, which is very time-consuming. (b) On the other hand, if the learning step is too large, the minimum point will be skipped or may not even be found.

However, not all objective functions resemble a standard bowl. They can be holes, ridges, plateaus, or any other irregular terrain that makes convergence difficult. Fig. 15 depicts the two main gradient descent challenges: If a random initial value is chosen on the image's left side, it will converge to a local minimum that is greater than the global minimum. It will take a long time to cross the plateau if it starts on the right, and if it stops training earlier than necessary, it will never reach the global minimum.

To address the various problems of gradient descent algorithms, the Google team proposed the idea of batch normalization (BN) (Ioffe and Szegedy 2015). BN is a neural network regularization technique that unifies the distribution of feature-map values by setting them to zero mean and unit variance. Furthermore, the BN layer helps to alleviate the problem of gradient vanishing and gradient explosion, improves the network's adaptability to different input data, speeds up the neural network's training process, and improves the

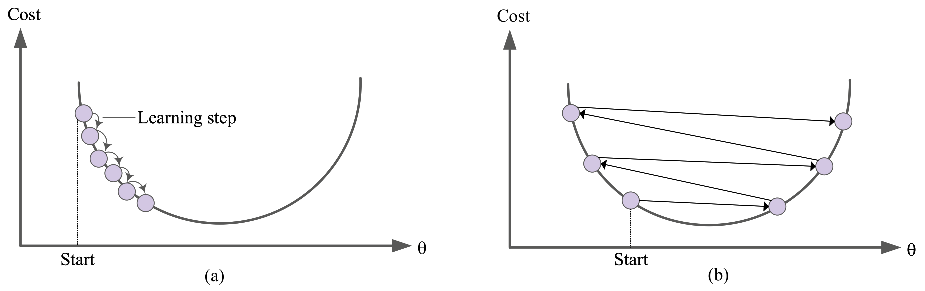


Fig. 14 Algorithms for gradient descent with excessively small or large learning steps

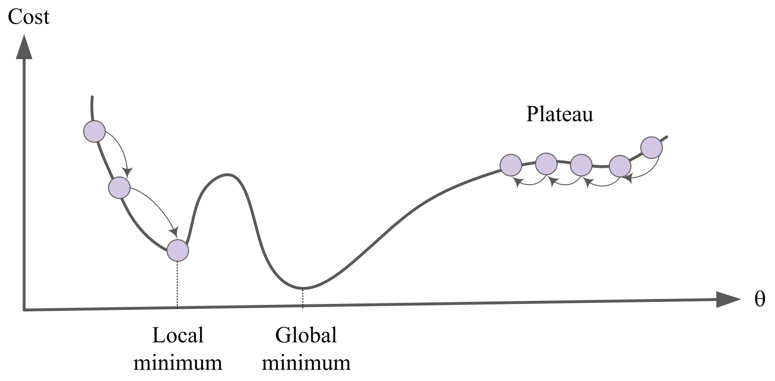


Fig. 15 Two main gradient descent challenges

network's generalization. It also avoids the problem of data death in the ReLU and makes weight initialization easier.

2.5 Dropout

Dropout facilitates regularization in the network by randomly omitting some units or connections with a predetermined probability, which eventually enhances generalization. This random dropping of some connections or units results in several thinned network architectures, from which one representative network is chosen with low weights. This chosen architecture is then regarded as an approximation of all proposed networks (Srivastava et al. 2014). Fig. 16 depicts the distinction between a fully connected layer and a dropout layer.

2.6 Fully connected layer

A fully connected layer is a global operation, as opposed to convolution and pooling, and is typically employed at the network's conclusion for classification. Like a multilayer perceptron neural network (MLP) (Isabona et al. 2022), each neuron in the fully connected layer is connected one by one with all the neurons in its preceding layers. Once the feature mapping obtained after several convolution and pooling operations is sufficient to recognize the features of the image, the next thing to consider is how to

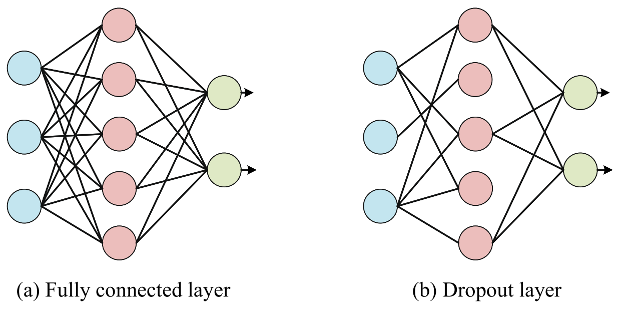


Fig. 16 Distinction between a fully connected layer and a dropout layer

perform the classification. Generally, the CNN will pull the multiple feature mappings that are finally obtained at the end into a long vector and send it to the fully connected layer, followed by the output layer, for classification. For example, when it comes to an image triple classification problem, the output layer of a CNN will have three neurons. In addition, the fully connected layer can integrate local information that is class-distinctive in the convolution or pooling layers (Sainath et al. 2013).

3 Image classification

3.1 Subtask explanation

Image classification (Chandra and Bedi 2021), which seeks to differentiate between distinct classes of objects, such as flowers, figures, and vehicles, based on various properties reflected in the image, is one of the fundamental challenges in computer vision. In other words, a computer can identify the class to which the objects in an image or video belong. The main process of image classification includes preprocessing the original image, extracting image features, and classifying the image using a pre-trained classifier, in which the extraction of image features plays a pivotal role. The data flow diagram for image classification is shown in Fig. 17. Traditional image classification algorithms can achieve the expected results in simple classification tasks. However, their performance in complex classification tasks is not satisfactory. CNN uses convolution kernels to extract features from the original input and automatically learns feature representations from massive sample data, giving the trained models stronger generalization abilities when compared to conventional image classification algorithms that manually extract features.

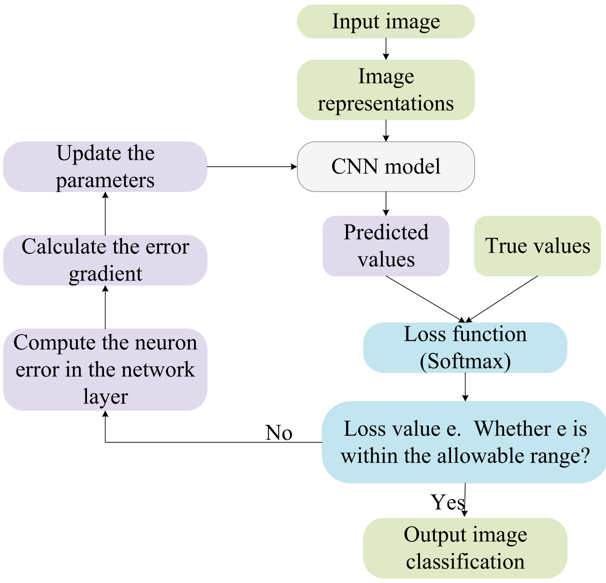

Input image
Image representations
Update the parameters
CNN model
Predicted values
True values
Calculate the error gradient
Loss function (Softmax)
Compute the neuron error in the network layer
No
Loss value e. Whether e is within the allowable range?
Yes
Output image classification

Fig. 17 Data flow diagram for image classification

3.2 AlexNet

LeNet was proposed by LeCun in 1998 (LeCun et al. 1998). LeNet is a feed-forward neural network consisting of two fully connected layers after five alternating layers of pooling and convolution. LeNet-5 is a LeNet extension and improvement that adds more convolution and fully connected layers. As shown in Fig. 18, the LeNet-5 network model has seven layers. LeNet-5 can share convolution kernels, reduce network parameters, perform well on the small-scale MNIST dataset, and achieve more than 98 % accuracy. CNN was first used for image recognition tasks thanks to the work of LeNet and LeNet-5, which also offered crucial lessons and insights for the later creation of deeper neural networks.

The concepts presented by David et al. in their 1968 seminal paper served as the foundation for the idea that LeCun and his colleagues implemented (Hubel and Wiesel 1968). The study on the striate cortex in monkeys categorized cells as simple, complex, or hypercomplex. It found smaller receptive fields, increased sensitivity to stimulus orientation, and a minority of cells with color-coding abilities. The evidence supports two vertical column systems in the studied cortex. The first type features columns with cells sharing receptive-field orientations, akin to cat orientation columns but likely smaller. The second system organizes cells into columns based on eye preference, with larger ocular dominance columns. The boundaries of the two systems appear to be independent. The cortex exhibits dual organization patterns: a vertical system aligns cells with common features along a line, mapping stimulus dimensions independently in superimposed mosaics. The horizontal system segregates cells hierarchically in layers, with lower orders (monocularly driven simple cells) near layer IV and higher orders in the upper and lower layers. These findings not only address the organizational aspects of receptive fields and functional structure but also provide a crucial foundation for further research into information processing in the cortical region of the brain.

However, due to the low performance of the hardware and the insufficiently rich dataset at that time, LeNet was not suitable for complex problems. In 2012, Krizhevsky et al. proposed AlexNet (Alom et al. 2018), which consists of five convolution layers and three fully connected layers. Each convolution layer contains a convolution kernel, a bias term, a ReLU activation function, and a local response normalization (LRN) module. The first convolution layer convolves the $224 \times 224 \times 3$ input image using 96 convolution kernels of size $11 \times 11 \times 3$ and stride 4. The second convolution layer takes the output of the first convolution layer as input and filters it with $5 \times 5 \times 48$ kernels. The third, fourth, and fifth

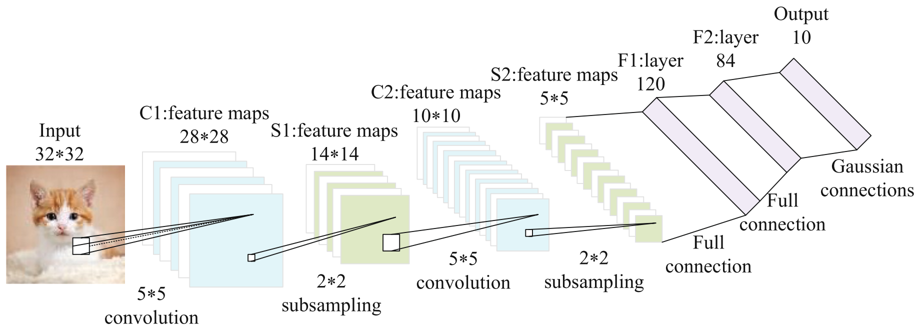


Fig. 18 Architecture of LeNet-5

convolution layers are connected to each other, with no pooling layer in between. The kernels of the second, fourth, and fifth convolution layers are only connected to those kernel maps of the previous convolution layer that are also located on the same GPU. The kernels of the third convolution layer are connected to all the kernel mappings of the second convolution layer. The neurons in the fully connected layer are connected to all the neurons in the previous layer. The response normalization layer follows the first and second convolution layers. The max pooling layer follows the response normalization layer and the fifth convolution layer. The image is convolved, fully connected, and finally fed into a Softmax classifier with 1000 nodes, which converts the output of the network into probabilistic values that can be used to predict the category of the image.

The image classification task of the ILSVRC reflects the most notable breakthrough of deep CNN in this area. In the 2012 ILSVRC, AlexNet demonstrated the potential of deep learning and finally won the competition with a Top-5 classification error rate of 16.4%, surpassing the performance of the second-place algorithm that performed classification by traditional methods. This competition attracted the attention of many researchers, and since then, improved algorithms based on CNN have also obtained excellent results in the ImageNet competition. Meanwhile, AlexNet became the dividing line between traditional and deep learning algorithms and was the first deep CNN model in modern times. Distinguished from traditional algorithms, AlexNet adopts many modern technical methods of deep convolutional networks for the first time, including using dual GPU parallel convolution operations in training, which overcomes the limitation of hardware resources on the learning ability and thus accelerates the training of the model. In order to address the gradient disappearance issue and hasten the convergence of the network model, after convolution filtering, the output excitation of the convolution layer is obtained using the ReLU activation function, which is then output to the subsequent convolution layer after local response normalization and down-sampling operations. By utilizing dropout and data augmentation approaches, AlexNet also lessens the model's overfitting.

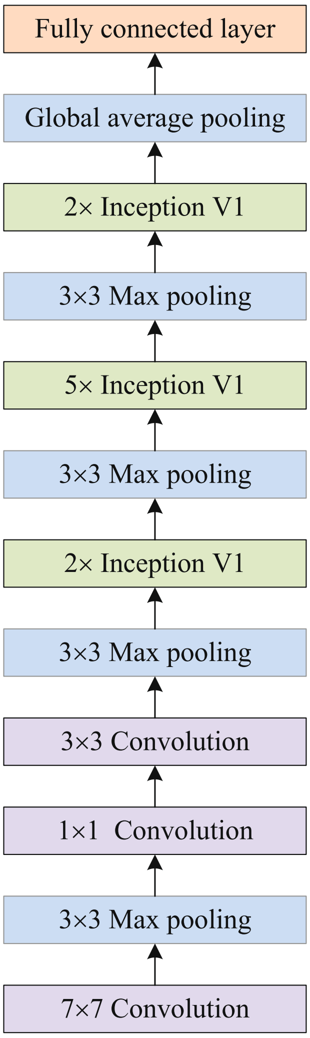
3.3 Visual geometry group

To examine the impact of a CNN's depth on its accuracy, Karen Sengupta et al. (2019) conducted a comprehensive evaluation of the performance of network models with increasing depth using small convolution filters (3×3) instead of the previous large convolution kernels (5×5) and proposed a series of Visual Geometry Group (VGG) models in 2014. With a classification error rate for the Top-5 of 7.3%, VGG finished as the second-place network in ILSVRC 2014. VGG made the following advancements in comparison to earlier neural network models: lowered the size of the convolution kernels while increasing the number of network layers. The modest size of the convolution kernels used in VGG, as opposed to the convolution kernels used in AlexNet, lowers the computational complexity and the number of training parameters. Simultaneously, the hypothesis that performance can be enhanced by continually deepening the network topology is also supported by VGG. To date, VGG-16 is still widely used in various tasks due to its simple structural features and its applicability in transfer learning.

3.4 GoogLeNet

The champion model in the 2014 ILSVRC is GoogLeNet (Khan et al. 2019). As shown in Fig. 19, GoogLeNet consists of nine Inception V1 modules, five down sampling layers, and

Fig. 19 Architecture of GoogLeNet



a number of other convolution and fully connected layers. Though GoogLeNet has deeper network layers, it still has a lesser number of parameters compared to VGG. Consequentially, when computer hardware resources are restricted, GoogLeNet is a superior solution for image classification. A GoogLeNet convolution layer has many convolution processes of varying sizes, allowing for the production of dense data while making optimal use of processing resources. Additionally, it makes use of sparse connections to eliminate redundant data and cut costs by skipping through pointless feature maps. Last but not least, the GoogLeNet reduces the connection density by adopting global average pooling rather than a fully connected layer.

By adding more hidden layers to CNN, the recognition accuracy and performance of deep neural networks can be enhanced (Szegedy et al. 2015), but it can lead to many issues. On the one hand, as the number of network layers rises, the network must learn more parameters, which easily leads to the model being overfitted to the training data set. On the other hand, networks with extra layers require robust hardware resources in order to maintain the required processing power. In order to overcome these problems, the research team at Google developed the concept of inception (Al Husaini et al. 2022), which aims to build the underlying neurons and a network topology for sparse high-performance computing. In in Fig. 20a, the original Inception structure is displayed. Based on experimental results, it is concluded that the structure's 5 × 5 convolution is the root cause of the excessive parameter issue. As a result, a new structure called Inception V1 is proposed. The structure of inception V1 is shown in Fig. 20(b). The main idea of inception V1 is to extract feature information from the preceding layers with three different-sized convolution kernels, fuse them, and pass them to the succeeding layers. The 1×1 convolution kernel is the most commonly utilized among them for data dimension reduction, which reduces convolution computation when passing to the next 3×3 and 5×5 convolution layers, avoiding the huge computation due to the increase in network size. The following layer can extract more valuable features from various scales by combining the features of the four channels.

Following Inception V1, Szegedy et al. proposed some optimizations to the Inception V1 structure and released the Inception V2 model in 2015 (Szegedy et al. 2016). To reduce the number of parameters and increase the discriminative nature of feature information, Inception V2 is improved by using two 3 × 3 convolution kernels instead of 5 × 5 convolution kernels, 1 × n convolution kernels, and n × 1 convolution kernels instead of n × n convolution kernels. Second, the pooling layer is optimized using a parallel structure to cut down on computation. Furthermore, by smoothing the probability distribution of labels, overfitting is minimized. Inception V3 is an improved version of Inception V1 and V2. The idea of Inception V3 was to reduce the computational cost of deep networks without affecting generalization. For this purpose, Szegedy et al. replaced large-size filters (5 × 5 and 7 × 7) with small and asymmetric filters (1 × 7 and 1 × 5) and used 1 × 1 convolution as a bottleneck before the large filters (Szegedy et al. 2017).

3.5 Residual network

A degradation problem emerges when deeper neural networks start to converge: accuracy increases to a saturation point and then rapidly declines as network depth increases. Nevertheless, the increase in layers that results in more training errors is what causes this degradation, rather than overfitting. Prior to residual network (ResNet Wightman et al. 2021), networks had relatively low layer counts; for example, the 2014 VGG network had only 19

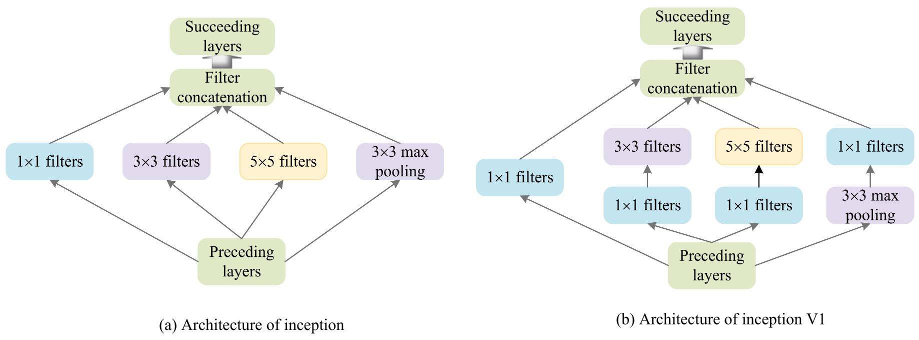


Fig. 20 Architecture of inception and inception V1

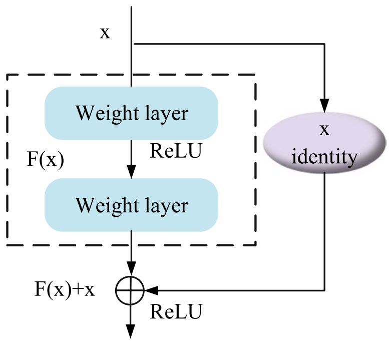
layers. ResNet, on the other hand, maintains greater accuracy while having 152 layers in its depth. ResNet alludes to the highway network concept Srivastava et al. (2015) and is composed of stacked residual blocks. The structure of a residual block is illustrated in Fig. 21. In addition to containing weighted layers, a residual block directly connects the input x to the output through a shortcut connection. The residual mapping is denoted as F(x), and the output is obtained by adding the residual mapping to the input, resulting in F(x) + x, representing the original mapping. The residual network encourages the stacked weighted layers to fit the residual mapping F(x) rather than the original mapping. Learning the residual mapping is simpler and more easily optimized compared to learning the original mapping. Furthermore, the shortcut connections enable the exchange of features between different layers, to some extent alleviating the problem of gradient vanishing. The Top-5 error rate of the residual network on the image classification task was reduced to 3.6%.

3.6 Squeeze and excitation network

In recent years, the attention mechanism has been another focus of CNN research. When the human eye scans an image, it first looks at the whole picture and then focuses its attention on a certain detail, concentrating its attention on the valuable part and ignoring the less valuable part. When we are designing neural network models, we hope that the models can have the same ability. Attention can be understood as selectively filtering out a small amount of important information from a large pool of data and focusing on these crucial details, while ignoring the majority of less significant information. The process of focusing is reflected in the calculation of weight coefficients, where larger weights indicate a stronger focus on the corresponding Value. In other words, the weights represent the importance of the information, and the Value is the corresponding piece of information. In this way, we can comprehend the attention mechanism (refer to Fig. 22). Imagine the constituent elements of Source as a series of <Key,Value> data pairs. Then, given an element Query in target, by calculating the similarity or correlation between Query and each Key, get the weight coefficients of the Value corresponding to each Key, and then weight and sum the Value, that is, we get the final Attention value. So the attention mechanism is essentially a weighted sum of the values of the elements in the Source, and the Query and Key are used to calculate the weight coefficients of the corresponding values.

Abstracting the specific calculations of the attention mechanism can be summarized into two processes: the first process involves calculating weight coefficients based on Query and Key, and the second process entails weighting and summing the values based on these weight

Fig. 21 A residual block



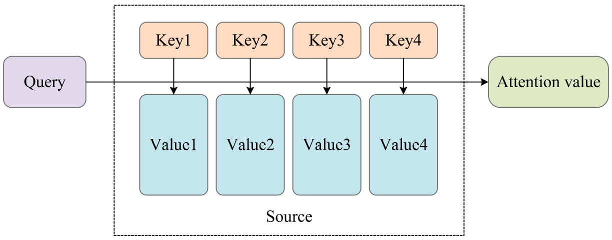


Fig. 22 The essential idea of attention

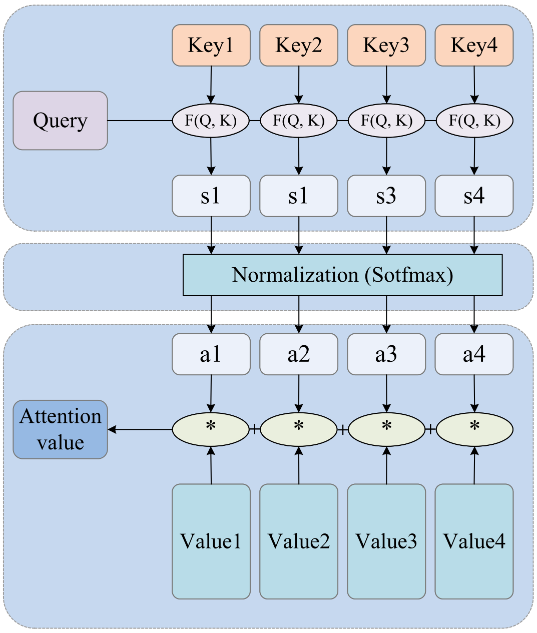
coefficients. The first process can be further divided into two stages: the first stage computes the similarity or correlation between Query and Key, and the second stage normalizes the raw scores obtained in the first stage. Fig. 23 illustrates the three-stage calculation process of attention.

In the first stage, various computation mechanisms can be introduced to calculate the similarity or correlation between Query and a given Key. The most common methods include computing the dot product of their vectors and calculating the cosine similarity, as illustrated below:

$$DotproductSim\left(Query, Key_i\right) = Query \cdot Key_i \tag{8}$$

$$\mathrm{Cosine}Sim\left(Query, Key_i\right) = \frac{Query \cdot Key_i}{|Query| \cdot |Key_i|} \tag{9}$$

Fig. 23 Three-stage process for computing attention



Due to the different methods used, the values produced in the first stage can have different ranges. In the second stage, a computation method similar to Softmax is introduced to transform the scores obtained in the first stage. On one hand, this normalization ensures that the original computed scores are unified into a probability distribution where the sum of all element weights is equal to 1. On the other hand, the intrinsic mechanism of Softmax helps emphasize the weights of important elements. Typically, the calculation is performed using the following formula:

$$\mathrm{a}_i = soft\max\left(Sim_i\right) = \frac{e^{Sim_i}}{\sum_{j=1}^{L_x} e^{Sim_j}} \tag{10}$$

where Lx=|$Source$| represents the length of the Source. The computed result a_i from the second stage represents the weight coefficient corresponding to $value_i$. Then, by performing a weighted sum, the attention value can be obtained:

$$Attention(Query, Source) = \sum_{i=1}^{L_x} a_i \cdot Value_i \tag{11}$$

Focusing on channel attention research, Hu proposed the squeeze-and-excitation block (SE block). The SE block explicitly models interdependencies between channels to recalibrate the feature responses within channels. This involves selectively enhancing useful channel features while suppressing irrelevant ones. Squeeze and excitation Networks (SENet Jin et al. 2022) won the 2017 ImageNet competition, similar to ResNet, both with a largely reduced error rate compared to previous models and low network complexity. The two primary parts of SENet are squeeze and excitation. A block of squeeze-and-excitation networks is shown in Fig. 24. The $\mathbf{f}_{tr}$ in the figure is the traditional convolution structure, $\mathbf{x}$ and $\mathbf{u}$ are the input and output of $\mathbf{f}_{tr}$, which are already present in the previous structures. The added part of SENet is the content after $\mathbf{u}$. In the image recognition task, the input image's dimensions are h, w, and c, where h stands for height, w for width, and c for channel count. The squeeze component is in charge of compressing the $h \times w \times c$ dimension into $1 \times 1 \times c$ dimension, which is the same as condensing $h \times w$ into a single dimension, and this is typically accomplished using global average pooling ($\mathbf{f}_{sq}$(.) in Fig. 24). The output $1 \times 1 \times c$ data is then fully concatenated ($\mathbf{f}_{ex}$(.) in Fig. 24, which is the excitation process), and finally, the self-gating technique is used to learn the excitation of each channel and scale this value to the c channels of $\mathbf{u}$ as the next level's input data. Controlling the scale size allows squeeze-and-excitation networks to strengthen critical channel properties while weakening non-important channel features, yielding good results and providing a novel notion for future research in this approach.

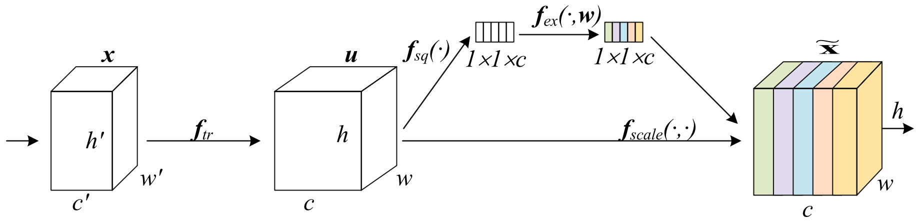


Fig. 24 A block of squeeze-and-excitation networks

3.7 MobileNet

In traditional CNN, the memory requirements and computational demands are substantial, making it impractical for running on mobile and embedded devices. Howard and his colleagues proposed a lightweight network, MobileNetV1 (Howard et al. 2017), tailored for mobile and embedded applications. Compared to traditional CNN, MobileNetV1 significantly reduces the model's parameters and computational workload while experiencing a minor decrease in accuracy. MobileNetV1 achieves 0.9% lower accuracy than VGG16, but with only 1/32 of the model's parameters. MobileNetV1 employs depthwise separable convolution layers, as illustrated in Fig. 25. This involves first applying depthwise convolution to each channel of the feature map, followed by pointwise 1×1 convolution, aiming to reduce computational load and model parameters. Two contraction hyperparameters, the width multiplier and the resolution multiplier, are introduced simultaneously to decrease computation, reduce volume, and improve accuracy. However, a drawback of this model is its low cost-effectiveness, as many convolution kernel parameters become zero during the training process. Subsequently, Google introduced MobileNetV2 (Sandler et al. 2018), which utilizes an inverted residual structure and a linear bottleneck structure. The inverted residual structure first employs a 1×1 convolution to increase dimensionality, deepening the channels to capture more feature information. It then applies a 3×3 depthwise convolution operation and concludes with a 1×1 convolution for dimensionality reduction, effectively reducing the number of parameters. One drawback of this model is the loss of diversity between layers, which cannot guarantee accuracy.

ImageNet (Deng et al. 2009), as one of the datasets for image classification tasks, has the characteristics of large-scale datasets and abundant image categories, and the trained model has good generalization ability, allowing it to obtain effective classification results on other image classification datasets such as CIFAR-10/100 Krizhevsky et al. (2009), Caltech-101 (Fei-Fei et al. 2004), and SUN (Xiao et al. 2010). Deep CNN models have improved in training thanks to the availability of a wide range of large-scale datasets, and models trained on these datasets have better generalization abilities. These generalization abilities can be used in practical applications to quickly learn the features of the datasets on their own and boost the effectiveness and efficiency of classification tasks. Performance comparisons of different architectures are shown in Table 1.

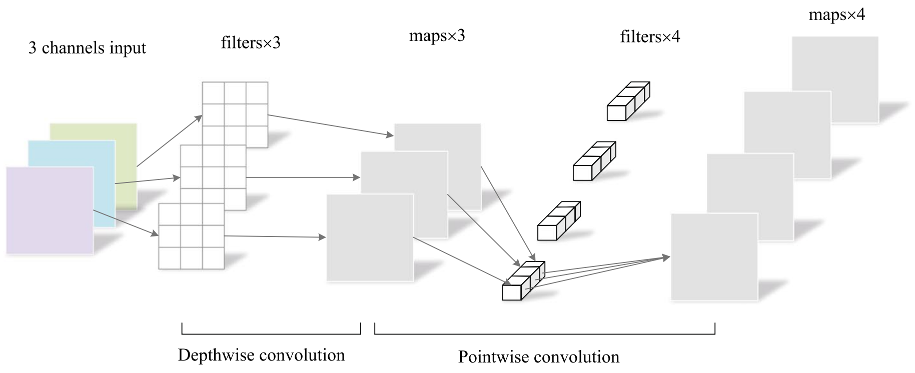


Fig. 25 Deep separable convolution layer structure

Table 1 Performance comparisons of different architectures

Models	Year	Depth	Parameters/M	Error rate/%
AlexNet Alom et al. (2018)	2012	8	60	16.4
VGG Sengupta et al. (2019)	2014	19	138	7.3
GoogLeNet Khan et al. (2019)	2015	22	4	6.7
ResNet Wightman et al. (2021)	2016	152	25.6	3.6
SENet Jin et al. (2022)	2017	152	27.5	2.3
MobileNetV1 Howard et al. (2017)	2017	–	4.2	5–10
MobileNetV2 Sandler et al. (2018)	2018	–	3.47	3–6

Top 5 error rate is reported for all architectures

- As shown in Table 1, from AlexNet to GoogLeNet, the accuracy of image classification increases progressively. This is attributed to the deeper architecture of the networks, which leads to more effective feature extraction.
- ResNet has a deeper network architecture compared to VGG, but the introduction of residual learning makes the network more easily optimized, mitigating gradient vanishing issues. Additionally, parameter sharing and reuse, along with a reduced parameter count, contribute to achieving higher performance with lower complexity and error rates.
- Deep neural networks incorporating attention mechanisms have achieved remarkable performance, as exemplified by the SE block proposed by Hu, which effectively models dependencies among channel features. Through these methods, it becomes evident that the core function of attention mechanisms is to emphasize useful components while disregarding those with relatively minor contributions to feature extraction. Consequently, integrating attention mechanisms into networks offers the advantage of enhancing model performance and improving the effective extraction of features.
- Although lightweight networks may not perform as well as classical deep CNNs in image classification on the ImageNet dataset, they significantly reduce the number of parameters. This indicates that lightweight networks effectively utilize model parameters by employing methods like depthwise separable convolution. This advantage is particularly valuable in resource-constrained environments, such as mobile devices or embedded systems, where they can still deliver relatively good performance while reducing model size. This makes them more suitable for practical deployment and operation.

Despite the fact that several CNN models have achieved outstanding performance in image classification, they have a number of drawbacks. Advanced CNN models frequently have intricate structures and a lot of parameters, requiring a lot of processing power and memory during training and deployment. The use of lightweight network topologies like MobileNet and EfficientNet, model pruning, and model compression, as well as other strategies to lessen model complexity and storage needs, can all be used to overcome this issue. The fact that many CNN models heavily rely on an enormous amount of labeled data to perform at their best is a huge hurdle. Large-scale annotated data can be expensive and time-consuming to acquire, though. Several techniques can be used to improve training data and lessen reliance on annotated data in order to address this difficulty. These techniques include transfer learning, semi-supervised learning, and data augmentation. Another problem is that typical CNN models may lose fine-grained information when used on small-sized images. Different strategies can be used to handle the limitations of small-sized photos in order to solve this issue. These methods involve leveraging shallow network designs, pyramid-style network structures, or smaller convolutional kernels. Recent years have seen the emergence of fresh study focuses and methodologies, like using transformer models for image categorization. Researchers have looked into replacing convolutional blocks with transformer model structures or implementing self-attention processes from transformers straight into CNN. As evidenced by models like DeiT, pyramid vision transformer, and swin transformer, these initiatives have yielded promising outcomes. A significant area for future research will be the combination of deep learning and reinforcement learning in image classification. The effectiveness of image classification models may be further improved by this fusion of methodologies.

4 Object detection

4.1 Subtask explanation

As a fundamental task in computer vision, object detection (Ma et al. 2023) is the key to solving more complex vision tasks such as image segmentation (Minaee et al. 2021), object tracking (Luo et al. 2021), behavior recognition (Hu et al. 2023), etc. The process of object detection and recognition typically consists of two steps: firstly, the prospective placement of each target object in a picture is localized, and secondly, the well-positioned objects are sorted into several categories. Compared with image classification, object detection focuses more on local regions of an image and specific sets of object classes. CNN has been used in object detection since the 1990's. However, because of a lack of training data and hardware resources such as computational power and storage devices, research on object detection using CNN received little attention and advanced slowly until 2012. The tremendous breakthrough of CNN in the ImageNet challenge in 2012 rekindled researchers' interest in deep CNN-based object detection, which led to a dramatic increase in object detection and recognition rates. At the same time, object detection has been widely applied in real-world scenarios, including autonomous driving (Zablocki et al. 2022), virtual reality (VR) (Xiong et al. 2021), intelligent video surveillance (Huang et al. 2021), etc.

Before the prosperity of deep learning, object detection algorithms depended on the traditional sliding window approach and were designed manually. The commonly used feature descriptors are Haar (Papageorgiou et al. 1998), Sift (Lowe 2004), Surf (Bay et al. 2006), etc., to train a unique shallow classifier for each class of target objects. Traditional object detection process is shown in Fig. 26. However, due to the factors of objects and the imaging environment, the method of manually designing features suffers from a lack of robustness, poor generalization, and low detection accuracy (Dicong et al. 2021). The bottlenecks of traditional object detection algorithms in practical applications are twofold. On the one hand, because the traditional object detection algorithm requires the designer to extract the features of the sample using prior knowledge, only a few parameters can appear in the feature design to lessen the difficulty of manually tuning the parameters. Shallow classifiers, on the other hand, require exponentially more parameters and training data in the face of tough detection tasks due to the lack of model depth. In response to the problem of manual parameter tuning of traditional object detection algorithms, the research boom in deep networks has brought new opportunities for the development of object detection. Compared with traditional

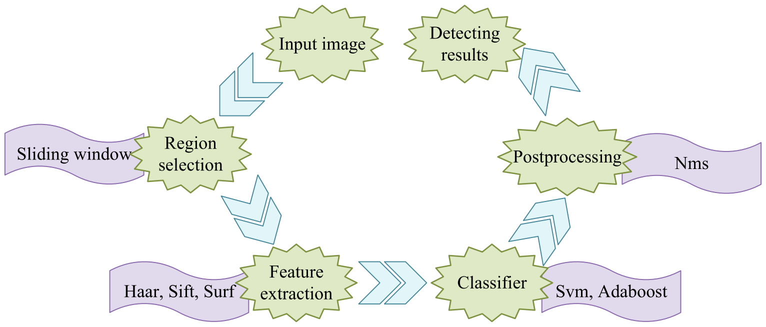


Fig. 26 Traditional object detection process

object detection algorithms, deep CNN can automatically learn feature representations of parameters from massive data sets and do not require additional training classifiers, which greatly improves the efficiency of the feature learning process.

In this paper, we outline known strategies for object detection from two perspectives: the region-based object detection algorithm (two-stage detectors) and the regression-based object detection algorithm (one-stage detectors). Fig. 27 depicts the basic procedure of two-stage detectors, which scan the whole image using multiple fixed-size sliding windows to generate a series of region proposal boxes, select the region proposal of the image, and then perform regression localization and classification of the targets that may exist in the region proposal to achieve object detection. One-stage detectors, in contrast, do not generate region proposals and combine feature extraction, object classification, and position regression into a single CNN to complete the process, simplifying the object detection process into a form of the end-to-end regression problem, as illustrated in Fig. 28.

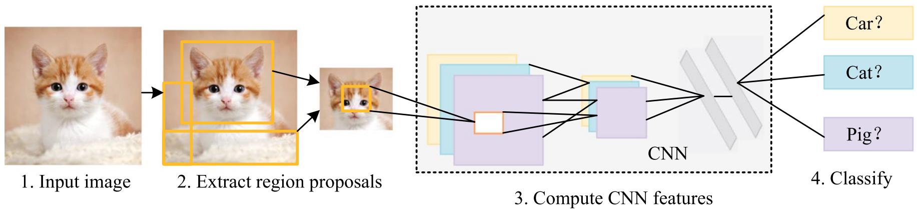


Fig. 27 Basic process of two-stage detectors

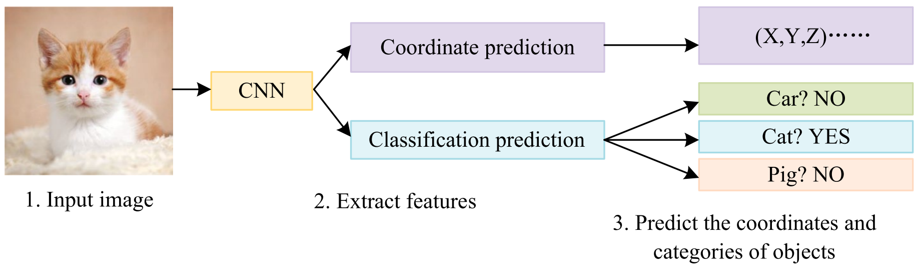


Fig. 28 Basic process of one-stage detectors

4.2 Representative two-stage detectors

Using CNN and region proposals, Girshick Girshick et al. (2014) introduced a deep learning object detection framework in 2014 called R-CNN. Initially, the model uses selective search (Ji et al. 2021), a non-deep learning algorithm, to propose candidate regions to be classified, and then feeds each candidate region into a CNN to extract features. Finally, these features are fed into a linear support vector machine for classification. To improve localization accuracy, a linear regression model is trained in R-CNN and used to correct the coordinates of the candidate region; this process is known as bounding box regression. On the PASCAL VOC object detection dataset, the model achieved an average correctness mean that was approximately 20% higher than the traditional algorithm, paving the way for the creation of two-stage detectors.

In R-CNN, approximately 2000 candidate regions are generated for each image, and each image's candidate regions must be feature extracted separately, making feature extraction a bottleneck in total test time. A Microsoft Research team applied SPP-Net (Ma et al. 2021), to object detection and elevated R-CNN's shortcoming. For the candidate regions generated by the selective search algorithm, SPP-Net projects the coordinates of these regions to the corresponding positions of the feature maps output by the highest convolution layer and then inputs the features corresponding to each candidate region into the spatial pyramid pooling layer to obtain a fixed-length feature representation. The subsequent stages keep similarities to R-CNN in that the fully connected layer receives these feature representations as input, a linear support vector machine uses the fully connected layer's feature output for classification, and bounding box regression is used to correct the candidate region coordinates. On the PASCAL VOC, the network achieved similar accuracy to the R-CNN, but the total time spent on the test was significantly reduced due to the fact that the time-consuming convolution operation was performed only once for each input image.

Like R-CNN, SPP-Net has certain limitations: the multi-stage training process of region proposal creation, feature extraction, and object classification is challenging, and it needs a lot of storage space for the derived features. Additionally, SPP-Net ignores the parameters of the network model's other layers and only adjusts the fully connected layer. To solve these problems, Fast R-CNN (Girshick 2015) was available in 2015; its structure is shown in Fig. 29. Compared with the CNN in R-CNN, Fast R-CNN improves on the last pooling layer by proposing the Region of Interest (RoI) pooling layer. The role of this layer is similar to that of the spatial pyramid pooling layer used in SPP-Net, which is to output

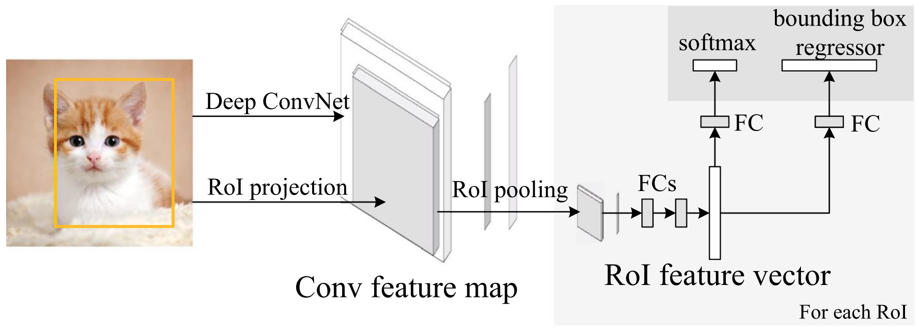


Fig. 29 Fast R-CNN architecture

a fixed-dimensional feature vector for any size of input, except that only a single level of spatial block partitioning is performed in the RoI pooling layer. This improvement allows Fast R-CNN, like SPP-Net, to input the whole input image together with the coordinates of the candidate regions generated by the selective search algorithm into a CNN and then perform RoI pooling on the feature maps of the output of the last convolution layer for the features corresponding to each candidate region. RoI pooling is performed on the output feature mapping of the last convolution layer, thus eliminating the need to perform a separate convolution computation for each candidate region. In addition, Fast R-CNN replaces the last softmax classification layer of the CNN with two side-by-side fully connected layers, one of which is still a softmax classification layer, and the other is a bounding box regressor, which is used for correcting the coordinate information of the candidate regions. During the training process, Fast R-CNN designs a multi-task loss function to train the two fully connected layers for classification and correction of candidate region coordinate information simultaneously. This training approach achieves better detection results on the PASCAL VOC dataset than the network obtained from the staged training previously used for R-CNN, thus eliminating the need for additional training of SVM classifiers in Fast R-CNN and realizing the integration of the process from extracting image features to completing detection.

These models have made improvements in the training process and the structure of CNN, however, they all use traditional algorithms to propose candidate regions, and these algorithms are implemented on CPUs, which makes the time of calculating candidate regions the bottleneck of the overall running time of the model. Therefore, in the Faster R-CNN model (Ren et al. 2015) designed by Ren Shaoqing et al., a candidate region network is proposed to improve this step, and its structure is shown in Fig. 30. Faster R-CNN improves on Fast R-CNN by setting a sliding window on the feature mapping output from the last convolution layer, which is fully connected to the candidate region network. For each position that the sliding window slides over, several anchor points with different scales and aspect ratios centered on the center of the sliding window are given in the model, and the candidate region network will compute a candidate region based on each anchor point

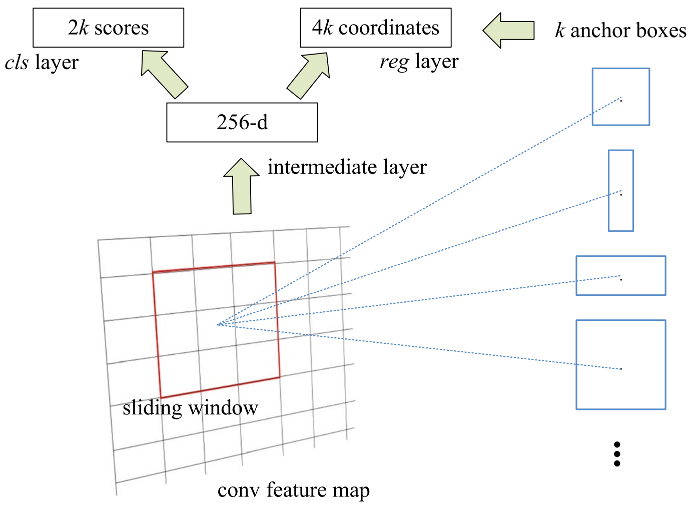


Fig. 30 Region proposal network (RPN)

accordingly. Since the process of proposing candidate regions by Faster R-CNN is based on the features extracted from the first few convolution layers of the Fast R-CNN used for detection and the candidate region network is also implemented on GPUs, the time overhead for proposing candidate regions is greatly reduced, the time required for detection is about 1/10th of the original, and the accuracy is improved, which suggests that the candidate region network is able to not only operate more efficiently but also improve the quality of the candidate regions produced.

Despite the fact that various applications have successfully recognized medium-size and large-size items in images with accuracy, small object detection remains problematic. Small objects are very difficult to recognize due to indistinguishable characteristics, complicated backdrops, low resolution, insufficient context information, and so on. As a result, there is considerable research being done in this field, and numerous deep-learning approaches have been developed recently with promising outcomes. In 2017, Lin et al. (2017) used the pyramidal hierarchy property of CNN to connect top-down lateralized high-level features with low-resolution, high-semantic information and low-level features with high-resolution, low-semantic information to construct Feature Pyramid Networks (FPNs) with high-level semantic information at different scales. The proposed FPN greatly improved the detection accuracy of the network and achieved state-of-the-art object detection, which will also become one of the important techniques to improve the accuracy of major networks in the future. Moreover, compared with other object detection models, the performance of FPN to improve classification accuracy in small object detection has achieved good results.

He et al. presented Mask R-CNN (He et al. 2017) in 2017, which integrates the concepts of Faster R-CNN and FCN. The feature extraction section uses a feature pyramid network (FPN) architecture and replaces the RoI pooling layer with a RoI align pooling layer, as well as a Mask prediction branch. The new FPN architecture improves the model's multi-scale feature extraction capacity and improves the recognition of small objects. However, the detection speed is the same as Faster R-CNN, which is insufficient for real-time monitoring applications.

Cao suggested a novel two-stage detector, D2Det (Cao et al. 2020), in 2020, that can handle the difficulties of precise localization and accurate classification at the same time. The model uses dense local regression to estimate the object's various dense frame offsets. Dense local regression is not confined to a fixed set of quantified key points but may also regress location-sensitive real dense offsets, allowing for more accurate localization. To improve classification accuracy, discriminative RoI pooling (DRP) is used to recover accurate object feature areas from the first and second phases, respectively. Table 2 compares the performance of two-stage detectors.

Table 2 Performance comparison of two-stage detectors

Models	Backbone	Detection speed(f/s)	AP
R-CNN Girshick et al. (2014)	AlexNet	0.03	–
SPPNet Ma et al. (2021)	ZF-5	2.0	–
Fast R-CNN Girshick (2015)	VGG-16	3.0	19.7
Faster R-CNN (Ren et al. (2015)	VGG-16	6.0	36.2
Mask R-CNN He et al. (2017)	ResNeXt-101	11.0	39.8
D2Det Cao et al. (2020)	ResNet-101	–	50.1

4.3 Representative one-stage detectors

One-stage detectors separated input photos into a number of cells, and each cell was used to forecast the item's center falling into the cell, as opposed to using pre-defined anchors for the object region. After just one stage, which has a quicker detection speed, the class and location of the object can be determined. However, compared to two-stage detectors, the detection accuracy is less accurate. The YOLO (you only look once) algorithm is a typical example of such an algorithm. The first one-stage object detection algorithm is YOLO (Redmon et al. 2016). The fundamental concept behind YOLO is to break the image up into multiple cells, predict the bounding box coordinates, the objects inside the boxes, and their corresponding confidence levels for each cell, and then remove the overlapping boxes using a non-maximal value algorithm to get the desired predicted boxes to achieve object detection. For instance, if the center of an object that needs to be recognized falls within one of the image's divided cells, the cell is in charge of determining the type and location of the target object.

When compared to two-stage detectors, the real-time object detector YOLO was incredibly quick. However, it struggles to accurately forecast bounding box scales and ratios, especially for small item detection, which leads to relatively low localization and classification accuracy. It also performs poorly on objects that are partially situated in one cell. In 2017, Redmon proposed YOLOv2 (Redmon and Farhadi 2017). YOLOv2 adds a batch normalization layer to all convolution layers to accelerate model learning, employs DarkNet-19 (Al-Haija et al. 2021) as the backbone, and employs a classification network, namely, a high-resolution classifier (Anuj and Gopalakrishna 2020), which pre-trains the model on high-resolution ImageNet datasets and then fine-tunes it using target datasets to improve model training stability. All of the strategies significantly increased detection accuracy while remaining fast.

DarkNet-53 served as the backbone for YOLOv3's extraction of picture characteristics, and logistics was employed in place of softmax for classification. The prediction was performed using the FPN network, and the previous frames were chosen using k-means clustering. In YOLOv3 (Redmon and Farhadi 2018), nine preceding frames were chosen, and three feature maps with various sensory fields were chosen to identify objects of various sizes.

YOLOv4 (Bochkovskiy et al. 2020) introduced mosaic data enhancement on the input images. In feature extraction, YOLOv4 integrated numerous novel techniques, including CSPDarkNet53 and the mish activation function. Instead of FPN, SPP and PAN were employed to extend the perceptual field and conduct feature fusion. Overall, YOLOv4 is a significant improvement over YOLOv3 and has considerable technical value since it introduces the most recent research methods within the realm of deep learning for validation testing. The network topology of YOLOv5 can be broken down into four sections: input, backbone, neck, and prediction. This makes it quite similar to YOLOv4. On the input photos, YOLOv5 applies adaptive image scaling, adaptive anchor frame computation, and mosaic data enhancement. A YOLOv5 invention, the backbone section employs a mix of focus structure and CSP structure, and the key is the slicing operation. Although YOLOv5 presently employs the same structure as YOLOv4, when it was launched, only the FPN structure was in use. The PAN structure was later introduced, and other network components were also modified. Although YOLOv4 already has a high level of detection precision, YOLOv5's numerous network architectures are more adaptable in real-world experiments.

Accuracy and speed are two critical performance characteristics in object identification, and how to balance them is critical in actual applications in industry. YOLOv6 (Li et al. 2022), designed for industrial applications, was released in 2022, and it supports the entire chain of industrial application requirements, such as model training, inference, and multi-platform deployment, as well as making several improvements and optimizations at the network structure, training strategy, and other algorithm levels. In terms of backbone, neck, head, and training approach, YOLOv6 outperforms earlier models. Li created a re-parameterizable and more efficient backbone network based on the RepVGG architecture, inspired by the notion of hardware-aware neural network design (Ding et al. 2021). The anchor-free paradigm is employed as the training approach, and to further increase the detection accuracy, the SimOTA (Ge et al. 2021) label assignment technique and SIoU (Gevorgyan 2022) bounding box regression loss are included. In terms of accuracy and speed, YOLOv6 surpasses other methods of the same volume on the COCO datasets.

Tan proposed EfficienDet (Tan et al. 2020), which is based on EfficienNet (Tan and Le 2019), in order to establish a model that balances detection speed and accuracy. This model introduces a collaborative scaling strategy while enabling quick multi-scale feature fusion using EfficienNet as the backbone and a bi-directional feature pyramid network as the feature network. Additionally, the concept of weighting is used. Joint scaling may evenly scale the depth, breadth, and resolution of the frame-class prediction network, the feature network, and the backbone network to produce the best outcomes.

Dong introduced CentripetalNet (Dong et al. 2020) to address the issue that key point-based detectors are prone to matching mistakes. This approach matches the corner points more precisely than the conventional embedding method, and this model can anticipate the corner point location and centripetal displacement of the item and match its corresponding corner. In the meantime, the cross-star-shaped variability convolution is proposed to maximize the learning of the cross-star feature in the partial feature map created after the corner pooling layer. On the COCO datasets, the model experimentally outperforms all other object detectors without anchor frames. The datasets are an important measure for the training and evaluation of different supervised algorithms. The two datasets that are most often utilized for object detection tasks are PASCAL VOC (Shetty 2016) and Microsoft COCO (Lin et al. 2014). Table 3 compares the performance of one-stage detectors.

The accuracy of the method is improved by optimizing the network structure by making the model more complicated, but this decreases the training and detection speeds, making it challenging to satisfy the requirement for real-time detection. Therefore, concentrating on the combination of accuracy and speed will be the direction of future study. We

Table 3 Performance comparison of one-stage detectors

Models	Backbone	Detection speed(f/s)	AP
YOLO Redmon et al. (2016)	VGG-16	45.0	–
YOLO-v2 Redmon and Farhadi (2017)	DarkNet-53	40.0	33.0
YOLO-v3 Redmon and Farhadi (2018)	DarkNet-19	20.0	21.6
YOLO-v4 Bochkovskiy et al. (2020)	CSPDarkNet-53	31	43.0
YOLO-v5	Focus+CSP	140.0	–
EfficienDet-D7 Tan et al. (2020)	EfficienNet-B7	–	52.2
CentripetalNet Dong et al. (2020)	Hourglass-104	–	48.0

simultaneously increase the accuracy and speed of object recognition to establish a balance between precision and speed that would satisfy the real demand. This is done by combining the high accuracy of region-based algorithms with the high speed of regression-based algorithms. An individual detection algorithm may perform SOTA on task A but may not perform as well on other tasks due to factors such as complex object backgrounds with substantial noise interference, and low contrast between object and background colors, which makes it challenging for the network to extract discriminate features, and small object sizes, which are challenging to detect. Therefore, a specific analysis of the difficulties of each detection task is beneficial to designing techniques that perform SOTA on a specific task.

In the past several years, object detectors based on CNN have entered the fast track of development, during which certain results have been achieved, but there is still room for further development. The following provides the frontier issues and research directions in this field to promote the research and improvement of subsequent object detectors.

(1) Weakly supervised and small sample detection: At this stage, the object detection model is trained by large-scale instance-labeled data, and data labeling is a time-consuming and labor-intensive project. Weakly supervised object detection reduces the cost of data annotation and efficiently trains the network with a little quantity of annotated data. The labeled data can be transferred from related domains through transfer learning and then trained with a small amount of labeled data in the desired domain to improve object detection in the desired domain.
(2) Multi-modal detection: To overcome the problem of monolithic data set categories, data from multiple modalities such as RGB images, 3D images, etc. can be fused, which is crucial for fields such as autonomous driving and intelligent robotics. Therefore, how to fuse data from different modalities and train the relevant detection models to migrate to multi-modal data will be the focus of future research.
(3) Video detection: There are a good deal of issues in video detection involving redundant feature information, video focus disorder, and occlusion, and so on, resulting in lower computational redundancy and detection accuracy. Therefore, the study of object detection algorithms based on video sequences will become one of the future research directions.

5 Video prediction

5.1 Subtask explanation

Transformer (Vaswani et al. 2017), with its strong capabilities in long-range modeling and parallelized sequence processing, has gradually attracted the interest of researchers in the fields of image processing and computer vision. It has demonstrated excellent performance in applications such as object tracking, image generation, and image enhancement. Fig. 31 illustrates a simplified architecture diagram of the Transformer model.

The transformer consists of two parts: encoder and decoder, and the detailed composition of each encoder and decoder is depicted in Fig. 32. The encoder employs a multi-head self-attention mechanism (MHSA), where the input matrix is linearly mapped to a feature subspace composed of multiple independent attention heads for dot product operations. Subsequently, the feature vectors and linear mappings are concatenated to obtain the final

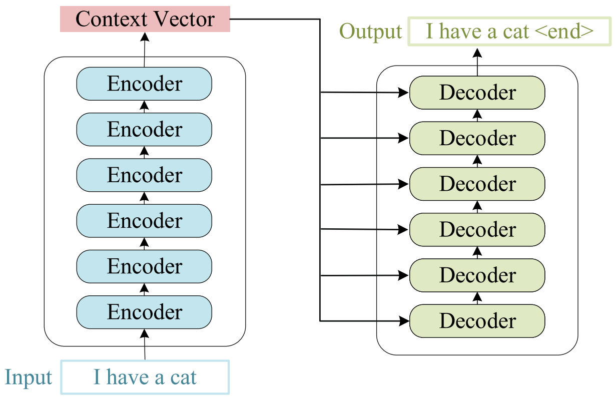


Fig. 31 Simplified architecture for transformer

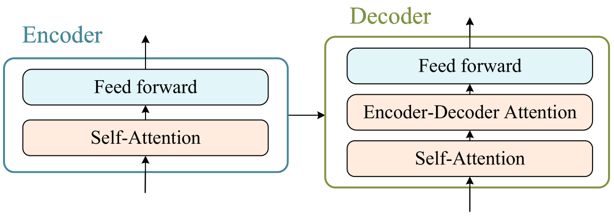


Fig. 32 Encoder and decoder

output, achieving the extraction of global information. Following this, a feedforward neural network (FFN), primarily consisting of two linear layers and a non-linear activation layer, is employed to transform dimensions and extract richer semantic information. The decoder is composed of self-attention, encoder-decoder attention, and feedforward components. For example, in Fig. 31, inputting the Chinese sentence 'I have a cat' goes through six encoders and produces something similar to a context vector. This can be understood as the encoder's understanding of the current input sentence. The obtained vector is then fed into the decoder. Each decoder performs self-attention on the output of the previous decoder, simultaneously applying encoder-decoder attention to the vector passed from the encoder. The result is then processed through a feedforward network, constituting one decoder. By stacking six decoders, the model learns and produces the final output.

Deep learning algorithms are mostly trained through a supervised approach, where model training is time-consuming and likely to be dependent on large amounts of labeled data. A key element we lack is predictive or unsupervised learning: the ability of a machine to simulate its environment, to predict future possibilities, and to understand how the world works through observation and engagement. Video prediction is a technique in which a computer learns spatio-temporal features inside a video frame and applies the learned features to the analysis and prediction of future frames. Since spatio-temporal information implies a large number of intrinsic laws of the real world and video prediction can be trained by a vast volume of unlabeled data, video prediction has attracted a lot of attention in academia, such as in human motion prediction (Liu et al. 2022), climate change (Ankrah et al. 2022), and traffic flow prediction (Gao et al. 2022). The goal of video prediction is to infer future frames from previous ones. Given a video sequence $\mathbf{X}_{t,T} = \{\mathbf{x}_i\}_{t-T+1}^{t}$, at time t with the past T frames, our goal is to forecast the sequence of events in the future $\mathbf{Y}_{t,T'} = \{\mathbf{x}_i\}_{t}^{t+T'}$, at time t that contains the next T' frames, where $\mathbf{x}_i \in \mathbb{R}^{C,H,W}$ is an image with channels C, height H, and width W. Formally, the predicting model is a mapping $\Gamma_\theta : \mathbf{X}_{t,T} \rightarrow \mathbf{Y}_{t,T'}$ with learnable parameters θ, optimized by:

$$\theta^* = \arg\min_{\theta} \Phi\left(\Gamma_\theta\left(\mathbf{X}_{t,T}\right), \mathbf{Y}_{t,T'}\right) \tag{12}$$

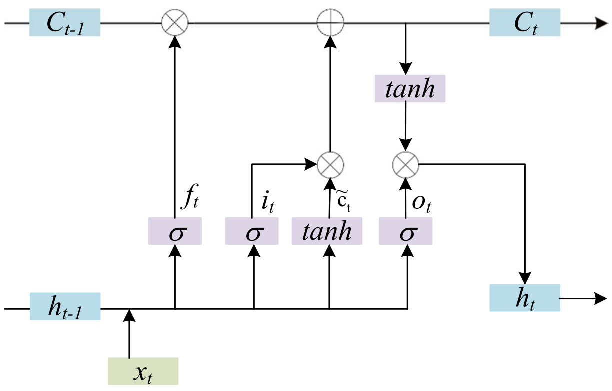
Where Φ can represent various loss functions; in our scenario, we specifically utilize Mean Squared Error (MSE) loss.

5.2 Deep learning applications

In response to the complexity and future uncertainty of video itself, scholars have achieved impressive results in a video in recent years by introducing various new neural operators, such as various RNNs (Wang et al. 2022), transformers (Ren et al. 2022), refinement structures (Chang et al. 2022), and applying different training strategies, involving adversarial training (Chan et al. 2022), etc. In order to assess the forecast's accuracy and the level of the predicted visuals, the procedure of creating video prediction models is also crucial. A majority of the current prominent video prediction models, such as self-encoders (Baldi 2012), recurrent neural networks (Medsker and Jain 2001), and generative adversarial networks (Creswell et al. 2018), are suggested based on deep learning, which has opened up new possibilities for video prediction.

Most video prediction models use self-encoders for video downscaling and generation since they can compress coding efficiently. Shi Shi et al. (2015) proposed the Convolutional LSTM (ConvLSTM) model that can solve the spatio-temporal sequence prediction problem after combining the sequence processing capability of LSTM and the spatial feature representation capability of CNN. Unlike various recurrent neural networks that acquire image features by using convolution operations on the input sequence images, which are one-dimensional word vector inputs when recurrent neural networks are applied to tasks such as translation, ConvLSTM acquires two-dimensional image inputs and can also input three-channel color images, i.e., three-dimensional inputs, depending on the task. ConvLSTM takes a single channel of 64×64 digital sequence images as input in the video frame prediction task. The ConvLSTM model, as illustrated

Fig. 33 Convolutional LSTM structure



in Fig. 33, contains the same three gate control units and one hidden layer as the LSTM model, namely an input gate, a forgetting gate, an output gate, and a hidden layer. The main distinction is that a single layer of convolution is computed after merging the input with the hidden layer at the present time, and this difference is critical for obtaining spatial structural information. Subsequently, Wang Wang et al. (2017) employed ConvLSTM units to develop an encoder that collected the spatio-temporal data contained in video frames and worked excellently in video prediction tasks. Lotter, inspired by "predictive coding" in neuroscience (Egner and Summerfield 2013), utilized ConvLSTM units to construct Prednet (Lotter et al. 2016), a multi-layer recurrent neural network that transmits the error caused by each layer of prediction to the next layer to assure the correctness of the network's final layer. ConvLSTM and an optical flow predictor are used in the spatio-temporal video auto-encoder (Patraucean et al. 2015) to record changes over time. ConvLSTM still struggles with the issue of producing ambiguous prediction frames, even though it partially resolves the issue of gathering and processing spatio-temporal information from video prediction frame sequences and increases prediction accuracy. Wang et al. (2019) presented a three-dimensional CNN architecture in conjunction with LSTM to distinguish various activities from video frames in order to address the issues of poor dynamic information obtained by the model, low prediction accuracy, and bad quality of the produced pictures. Wang's model outperforms others in terms of prediction accuracy, according to experimental data. The CNN encoder and RNN decoder are combined in a variable generation framework in conditional VRNN (Castrejon et al. 2019). According to CrevNet (Yu et al. 2020), the input for information-preserving feature transformation should be encoded and decoded using a CNN-based normalized stream module.

Contrary to popular belief, a completely CNN-based architecture is less widespread than the aforementioned models since its simplicity frequently necessitates the use of sophisticated modules and training techniques to increase novelty and performance, such as adversarial training (Yang et al. 2023), knowledge distillation (Feng et al. 2023), and optical flow approaches (Sui et al. 2022). Thus, Gao et al. introduced SimVP, a simple yet effective CNN video prediction model (Gao et al. 2022). SimVP is capable of achieving SOTA outcomes without the need for sophisticated modules, techniques, or tricks. Its minimal computing cost also makes it simple to scale to various scenarios. SimVP may function as a robust baseline and provide fresh perspectives for further study. Mean square error (MSE) loss is a way to train the model end-to-end, and it is totally constructed on top of CNN. The encoder, translator, and decoder of the SimVP model are all made entirely of CNN. Spatial feature extraction is done by the encoder, temporal evolution is learned by the translator, and spatial and temporal information are combined by the decoder to anticipate future frames.

More application possibilities for video prediction will be possible owing to improved model prediction performance and accuracy. First, deep learning-trained video prediction models have previously been used in areas like action identification and video interpretation. Additionally, for the self-driving industry, which has seen substantial progress in recent decades, if accurate future scene predictions can be made using the information currently available about the scene as it is being observed in real-time, driverless cars will be able to take the necessary precautions and, to the greatest extent possible, avoid risks. In the realm of computer vision, video prediction is an intriguing and challenging job. Most of the preceding models can forecast certain basic scenes successfully. Thus, future studies may begin with delicate circumstances, while the accuracy of prediction will be increased if the probability distribution of dynamic scenes can be modeled and predicted.

6 CNN challenges

Deep CNNs have demonstrated strong performance on data with a grid-like topology or that is time series in nature. However, in real-world applications, deep CNN architectures have run into additional difficulties. The different researchers have fascinating discussions about CNN's performance on different machine learning tasks. The following list includes some of the difficulties encountered when training deep CNN models:

- Given that deep CNNs are generally like black boxes, interpretation and explanation may be lacking. As a consequence, it can be difficult to verify them at times.
- The choice of hyper-parameters (for example, learning step, stride, and filter) has a substantial influence on CNN performance. However, because selecting optimal hyper-parameters involves a great deal of knowledge and talent and these hyper-parameters are tremendously internally dependent, any tiny alteration can have a significant influence on the final training outcomes. As a result, careful selection of hyper-parameters is a key design challenge that must be handled using an appropriate optimization technique. In this context, meta-heuristic algorithms may be utilized to automatically tune hyper-parameters by doing both random and directed searches based on prior findings.
- Deep CNN models for mobile devices are difficult to implement due to the necessity of maintaining model accuracy while taking into account the model's size and performance.
- Deep neural networks need a lot of data and processing power to be trained. Even when using the same set of datasets for various tasks, the data labeling varies, making the task of manually collecting large-scale and annotated datasets challenging. This results in a significant increase in labor and time costs for labeling the datasets created for a particular task. The success of the model training might also be significantly impacted by the datasets' annotation quality. Therefore, the creation of extensive and precisely labeled datasets has emerged as a critical issue for computer vision research. By using unsupervised learning approaches to extract hierarchical features, the requirement for a lot of labeled data may be reduced. Simultaneously, further research into how to construct effective and scalable parallel learning algorithms is imperative to accelerate the learning process.
- CNN deep models, when evaluated, need plenty of memory to hold numerous parameters and are highly time-consuming, making them unfeasible for deployment on resource-limited mobile platforms and other portable devices. As a result, it is critical to research ways to minimize the level of sophistication in neural networks while producing models that execute rapidly without sacrificing accuracy.
- Despite the outstanding performance of deep CNN in various applications, there is still a lack of theoretical and mathematical foundations. The neural network model is evolving toward deeper layers and a greater parameter scale as deep learning technology advances. Hence, discovering strategies to lower the computational complexity of the model is critical, which necessitates ongoing optimization in theory and experiment. Meanwhile, deep learning's mathematical theory is not flawless, and model optimization at this point is heavily reliant on the designer's prior knowledge, which is detrimental to the whole theoretical framework of deep learning. As a result, understanding what characteristics deep networks have learned and the basics of deep CNN to achieve high performance is an increasingly prominent study field.

7 Future directions

The incorporation of novel ideas into the design of CNN architectures has shifted the focus of research, particularly in the field of computer vision. Research on CNN architecture is very promising, and one of the most popular deep learning methods in the future is probably going to be related to it.

- One of the potential areas of CNN research is ensemble learning. By extracting different levels of semantic representations, the combination of multiple and diverse architectures can help the model improve its robustness and generalization to a variety of image categories.
- CNNs and their variations are extensively employed in diverse computer vision applications; however, the majority of CNN architectures are tailored to specific uses. Better-performing generic architectures are always needed (Patel et al. 2022).
- The key process by which the human visual system acquires information from images is attention. Furthermore, the attention mechanisms extract key information from images and store it in context with other visual components. The spatial relevance of objects and their distinguishing features can be maintained in subsequent learning stages in future research.
- CNN learning power is typically increased by increasing network size, which can be accomplished in a reasonable amount of time using the Nvidia DGX-2 supercomputer. However, in terms of memory usage and computational resources, training deep and high-volume architectures continues to be a significant overhead. As a result, numerous advancements in hardware technology are still needed to speed up CNN research.
- Deep CNNs have a large number of hyper-parameters, such as learning step, stride, filter, and so on. The selection of hyper-parameters and the evaluation time of deep networks make the parameter tuning task quite difficult. In this context, meta-heuristic algorithms can be used to automatically tune hyper-parameters by performing both random and directed searches based on previous results.
- The future of network design is neural architecture search, which has grown in popularity due to the time-intensive and labor-intensive nature of human network design. It does, however, have some requirements for the experimental environment due to its lengthy training period and significant memory resource consumption.
- Human activity recognition is a popular area of research in the field of CNN. The various CNN variants for human activity and pose recognition have been described in references (Vishwakarma and Singh 2019; Singh and Vishwakarma 2021; Dhiman and Vishwakarma 2020).

8 Conclusion

CNN has made impressive strides, particularly in image processing and video-related tasks, which has rekindled interest in deep learning among academics. Several studies have been done in this context to enhance CNN's performance, including activation, optimization, regularization, and innovations in architecture. This paper reviews the research progress of CNN architecture in computer vision, especially in image classification, target detection, and video prediction. In addition, this paper also covers the fundamental elements of CNN, its applications, challenges, and future directions. We have shown that CNN outperforms classical methods when it comes to classification, detection, and prediction. Through exploiting depth and other structural modifications, CNN's learning performance has dramatically improved over time. According to recent literature, the increase in CNN performance is primarily attributable to the replacement of the conventional layer structure with blocks. The function of an auxiliary learner can be performed by a block in a network. These additional learners leverage spatial or feature-map information or even enhance input channels to increase performance. Additionally, modular learning is supported by CNN's block-based design, which makes the structure easier to grasp.

As a review, this paper will inevitably suffer from the following shortcomings: First, it is limited by the scope of literature and time, resulting in the failure to comprehensively cover all relevant research work. Research on certain emerging areas or specific application scenarios may fail to be covered, and there are certain research blind spots. Second, considering the influence of subjectivity, we realize that the review may be influenced by the subjective judgment of the authors, which may have a certain impact on the objectivity of the research area. As a result, in future studies, we will need to sift through the relevant literature more thoroughly and deal with the subjective factors more cautiously in order to comprehend and investigate the application of CNN in computer vision in a more comprehensive and in-depth manner.

Acknowledgements This work was supported by the National Social Science Fund of China under Grant No. 22BTJ057.

Author contributions Xia Zhao: Conceptualization, Methodology, Investigation, Writing-original draft, Writing-review & editing. Limin Wang: Writing-review & editing, Project administration, Funding acquisition. Yufei Zhang: Writing-review & editing. Xuming Han: Writing-original draft, Investigation, Supervision. Muhammet Deveci: Writing-review & editing, Supervision. Milan Parmar: Conceptualization, Language polish.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

Al-Haija QA, Smadi M, Al-Bataineh OM (2021) Identifying phasic dopamine releases using darknet-19 convolutional neural network. In: 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), pp. 1–5.

Al Husaini MAS, Habaebi MH, Gunawan TS, Islam MR, Elsheikh EA, Suliman F (2022) Thermal-based early breast cancer detection using inception v3, inception v4 and modified inception mv4. Neural Comput Appl 34(1):333–348

Alom MZ, Taha TM, Yakopcic C, Westberg S, Sidike P, Nasrin MS, Van Esesn BC, Awwal AAS, Asari VK (2018) The history began from alexnet: A comprehensive survey on deep learning approaches. arXiv preprint arXiv:1803.01164

Ankrah J, Monteiro A, Madureira H (2022) Bibliometric analysis of data sources and tools for shoreline change analysis and detection. Sustainability 14(9):4895

Anuj L, Gopalakrishna M (2020) ResNet50-YOLOv2-convolutional neural network based hybrid deep structural learning for moving vehicle tracking under occlusion. Solid State Technol 63(6):3237–3258

Baldi P (2012) Autoencoders, unsupervised learning, and deep architectures. In: Proceedings of ICML Workshop on Unsupervised and Transfer Learning, pp. 37–49. JMLR Workshop and Conference Proceedings

Bay H, Tuytelaars T, Van Gool L (2006) Surf: speeded up robust features. Lecture Notes Comput Sci 3951:404–417

Bhatt D, Patel C, Talsania H, Patel J, Vaghela R, Pandya S, Modi K, Ghayvat H (2021) CNN variants for computer vision: history, architecture, application, challenges and future scope. Electronics 10(20):2470

Bochkovskiy A, Wang C-Y, Liao H-YM (2020) Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934

Bouvrie, J (2006) Introduction Notes on Convolutional Neural Networks," (1)

Cao J, Cholakkal H, Anwer RM, Khan FS, Pang Y, Shao L (2020) D2det: Towards high quality object detection and instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11485–11494

Castrejon L, Ballas N, Courville A (2019) Improved conditional vrnns for video prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 7608–7617

Chan ER, Lin CZ, Chan MA, Nagano K, Pan B, De Mello S, Gallo O, Guibas LJ., Tremblay J, Khamis S (2022) Efficient geometry-aware 3d generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16123–16133

Chan JY-L, Bea KT, Leow SMH, Phoong SW, Cheng WK (2023) State of the art: a review of sentiment analysis based on sequential transfer learning. Artif Intell Rev 56(1):749–780

Chandra MA, Bedi S (2021) Survey on SVM and their application in image classification. Int J Inf Technol 13:1–11

Chang Z, Zhang X, Wang S, Ma S, Gao W (2022) Stau: A spatiotemporal-aware unit for video prediction and beyond. arXiv preprint arXiv:2204.09456

Chen Y, Dai X, Chen D, Liu M, Dong X, Yuan L, Liu Z (2022) Mobile-former: Bridging mobilenet and transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5270–5279

Creswell A, White T, Dumoulin V, Arulkumaran K, Sengupta B, Bharath AA (2018) Generative adversarial networks: an overview. IEEE Signal Process Mag 35(1):53–65

Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255.

Dhiman C, Vishwakarma DK (2020) View-invariant deep architecture for human action recognition using two-stream motion and shape temporal dynamics. IEEE Trans Image Process 29:3835–3844

Dicong W, Chenshuai B, Kaijun W (2021) Survey of video object detection based on deep learning. J Front Comput Sci Technol 15(9):1563

Ding X, Zhang X, Ma N, Han J, Ding G, Sun J (2021) Repvgg: Making vgg-style convnets great again. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13733–13742

Dong Z, Li G, Liao Y, Wang F, Ren P, Qian C (2020) Centripetalnet: Pursuing high-quality keypoint pairs for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10519–10528

Egner T, Summerfield C (2013) Grounding predictive coding models in empirical neuroscience research. Behav Brain Sci 36(3):210–211

Fei-Fei L, Fergus R, Perona P (2004) Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: 2004 Conference on Computer Vision and Pattern Recognition Workshop, pp. 178–178.

Feng Z, Guo Y, Sun Y (2023) CEKD: Cross-modal edge-privileged knowledge distillation for semantic scene understanding using only thermal images. IEEE Robot Autom Lett 8(4):2205–2212

Fernandes S, Fanaee-T H, Gama J (2021) Tensor decomposition for analysing time-evolving social networks: an overview. Artif Intell Rev 54:2891–2916

Gao Z, Tan C, Wu L, Li SZ (2022) Simvp: Simpler yet better video prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3170–3180

Ge Z, Liu S, Wang F, Li Z, Sun J (2021) Yolox: Exceeding yolo series in 2021. arXiv preprint arXiv:2107.08430

Gevorgyan Z (2022) Siou loss: More powerful learning for bounding box regression. arXiv preprint arXiv:2205.12740

Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448

Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587

Guo G, Han L, Wang L, Zhang D, Han J (2023) Semantic-aware knowledge distillation with parameter-free feature uniformization. Visual Intell 1(1):6

He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969

Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507

Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861

Hu K, Jin J, Zheng F, Weng L, Ding Y (2023) Overview of behavior recognition based on deep learning. Artif Intell Rev 56(3):1833–1865

Huang C, Wu Z, Wen J, Xu Y, Jiang Q, Wang Y (2021) Abnormal event detection using deep contrastive learning for intelligent video surveillance system. IEEE Trans Industr Inform 18(8):5171–5179

Huang L, Qin J, Zhou Y, Zhu F, Liu L, Shao L (2023) Normalization techniques in training dnns: Methodology, analysis and application. IEEE Transactions on Pattern Analysis and Machine Intelligence

Hubel DH, Wiesel TN (1968) Receptive fields and functional architecture of monkey striate cortex. J Physiol 195(1):215–243

Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448–456. pmlr

Isabona J, Imoize AL, Ojo S, Karunwi O, Kim Y, Lee C-C, Li C-T (2022) Development of a multilayer perceptron neural network for optimal predictive modeling in urban microcellular radio environments. Appl Sci 12(11):5713

Ji X, Yan Q, Huang D, Wu B, Xu X, Zhang A, Liao G, Zhou J, Wu M (2021) Filtered selective search and evenly distributed convolutional neural networks for casting defects recognition. J Mater Process Technol 292:117064

Jin X, Xie Y, Wei X-S, Zhao B-R, Chen Z-M, Tan X (2022) Delving deep into spatial pooling for squeeze-and-excitation networks. Pattern Recognit 121:108159

Khan RU, Zhang X, Kumar R (2019) Analysis of ResNet and GoogleNet models for malware detection. J Comput Virol Hacking Tech 15:29–37

Krizhevsky A, Hinton G, et al (2009) Learning multiple layers of features from tiny images

LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324

Li Z, Liu F, Yang W, Peng S, Zhou J (2021) A survey of convolutional neural networks: analysis, applications, and prospects. IEEE transactions on neural networks and learning systems

Li J et al. (2022) Recent advances in end-to-end automatic speech recognition. APSIPA Transactions on Signal and Information Processing **11**(1)

Li C, Li L, Jiang H, Weng K, Geng Y, Li L, Ke Z, Li Q, Cheng M, Nie W, et al (2022) Yolov6: A single-stage object detection framework for industrial applications. arXiv preprint arXiv:2209.02976

Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, pp. 740–755. Springer

Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2117–2125

Liu Z, Wu S, Jin S, Ji S, Liu Q, Lu S, Cheng L (2022) Investigating pose representations and motion contexts modeling for 3d motion prediction. IEEE Transn Pattern Anal Mach Intell 45(1):681–697

Lotter W, Kreiman G, Cox D (2016) Deep predictive coding networks for video prediction and unsupervised learning. arXiv preprint arXiv:1605.08104

Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60:91–110

Luo W, Xing J, Milan A, Zhang X, Liu W, Kim T-K (2021) Multiple object tracking: a literature review. Artif intell 293:103448

Ma X, Guo J, Sansom A, McGuire M, Kalaani A, Chen Q, Tang S, Yang Q, Fu S (2021) Spatial pyramid attention for deep convolutional neural networks. IEEE Trans Multimedia 23:3048–3058

Ma P, Li C, Rahaman MM, Yao Y, Zhang J, Zou S, Zhao X, Grzegorzek M (2023) A state-of-the-art survey of object detection techniques in microorganism image analysis: from classical methods to deep learning approaches. Artif Intell Rev 56(2):1627–1698

Medsker LR, Jain L (2001) Recurrent neural networks. Des Appl 5:64–67

Minaee S, Boykov Y, Porikli F, Plaza A, Kehtarnavaz N, Terzopoulos D (2021) Image segmentation using deep learning: a survey. IEEE Trans Pattern Anal Mach Intell 44(7):3523–3542

Nwankpa C, Ijomah W, Gachagan A, Marshall S (2018) Activation functions: Comparison of trends in practice and research for deep learning. arXiv preprint arXiv:1811.03378

Papageorgiou CP, Oren M, Poggio T (1998) A general framework for object detection. In: Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271), pp. 555–562. IEEE

Patel C, Bhatt D, Sharma U, Patel R, Pandya S, Modi K, Cholli N, Patel A, Bhatt U, Khan MA (2022) DBGC: dimension-based generic convolution block for object recognition. Sensors 22(5):1780

Patraucean V, Handa A, Cipolla R (2015) Spatio-temporal video autoencoder with differentiable memory. arXiv preprint arXiv:1511.06309

Redmon J, Farhadi A (2017) Yolo9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7263–7271

Redmon J, Farhadi A (2018) Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767

Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788

Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems **28**

Ren J, Zheng Q, Zhao Y, Xu X, Li C (2022) Dlformer: Discrete latent transformer for video inpainting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3511–3520

Sainath TN, Kingsbury B, Mohamed A-r, Dahl GE, Saon G, Soltau H, Beran T, Aravkin AY, Ramabhadran B (2013) Improvements to deep convolutional neural networks for lvcsr. In: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, pp. 315–320. IEEE

Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L-C (2018) Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520

Sengupta A, Ye Y, Wang R, Liu C, Roy K (2019) Going deeper in spiking neural networks: VGG and residual architectures. Front Neurosci 13:95

Shetty S (2016) Application of convolutional neural network for image classification on pascal voc challenge 2012 dataset. arXiv preprint arXiv:1607.03785

Shi X, Chen Z, Wang H, Yeung D-Y, Wong W-K, Woo W-c (2015) Convolutional lstm network: A machine learning approach for precipitation nowcasting. Advances in neural information processing systems **28**

Singh T, Vishwakarma DK (2019) Video benchmarks of human action datasets: a review. Artif Intell Rev 52:1107–1154

Singh T, Vishwakarma DK (2021) A deeply coupled convnet for human activity recognition using dynamic and RGB images. Neural Comput Appl 33:469–485

Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958

Srivastava RK, Greff K, Schmidhuber J (2015) Highway networks. arXiv preprint arXiv:1505.00387

Stepanov S, Spiridonov D, Mai T (2023) Prediction of numerical homogenization using deep learning for the Richards equation. J Comput Appl Math 424:114980

Sui X, Li S, Geng X, Wu Y, Xu X, Liu Y, Goh R, Zhu H (2022) Craft: Cross-attentional flow transformer for robust optical flow. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 17602–17611

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9

Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826

Szegedy C, Ioffe S, Vanhoucke V, Alemi A (2017) Inception-v4, inception-resnet and the impact of residual connections on learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31

Tan M, Le Q (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp. 6105–6114.

Tan M, Pang R, Le QV (2020) Efficientdet: Scalable and efficient object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10781–10790

Uddin MP, Mamun MA, Hossain MA (2021) PCA-based feature reduction for hyperspectral remote sensing image classification. IETE Tech Rev 38(4):377–396

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. Advances in neural information processing systems **30**

Vishwakarma DK, Singh T (2019) A visual cognizance based multi-resolution descriptor for human action recognition using key pose. AEU-Int J Electron Commun 107:157–169

Wang Y, Long M, Wang J, Gao Z, Yu PS (2017) Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. Advances in neural information processing systems **30**

Wang Y, Jiang L, Yang M-H, Li L-J, Long M, Fei-Fei L (2019) Eidetic 3d lstm: A model for video prediction and beyond. In: International Conference on Learning Representations

Wang Y, Wu H, Zhang J, Gao Z, Wang J, Philip SY, Long M (2022) Predrnn: a recurrent neural network for spatiotemporal predictive learning. IEEE Trans Pattern Anal Mach Intell 45(2):2208–2225

Wightman R, Touvron H, Jégou H (2021) Resnet strikes back: An improved training procedure in timm. arXiv preprint arXiv:2110.00476

Xiao J, Hays J, Ehinger KA, Oliva A, Torralba A (2010) Sun database: Large-scale scene recognition from abbey to zoo. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3485–3492.

Xiong J, Hsiang E-L, He Z, Zhan T, Wu S-T (2021) Augmented reality and virtual reality displays: emerging technologies and future perspectives. Light Sci Appl 10(1):216

Yan S, Xiong X, Arnab A, Lu Z, Zhang M, Sun C, Schmid C (2022) Multiview transformers for video recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3333–3343

Yang J, Soltan AA, Eyre DW, Yang Y, Clifton DA (2023) An adversarial training framework for mitigating algorithmic biases in clinical machine learning. NPJ Digit Med 6(1):55

Yang W, Yu H, Cui B, Sui R, Gu T (2023) Deep neural network pruning method based on sensitive layers and reinforcement learning. Artif Intell Rev 56:1897–917

Yu K, Jia L, Chen Y, Xu W (2013) Deep learning: yesterday, today, and tomorrow. J Comput Res Dev 50(9):1799–1804

Yu W, Lu Y, Easterbrook S, Fidler S (2020) Efficient and information-preserving future frame prediction and beyond

Zablocki É, Ben-Younes H, Pérez P, Cord M (2022) Explainability of deep vision-based autonomous driving systems: review and challenges. Int J Comput Vision 130(10):2425–2452

Authors and Affiliations

Xia Zhao[1] · Limin Wang[1] · Yufei Zhang[2] · Xuming Han[3] · Muhammet Deveci[4,5,6] · Milan Parmar[7]

✉ Limin Wang
20211016@gdufe.edu.cn

✉ Xuming Han
hanxuming@jnu.edu.cn

Muhammet Deveci
muhammetdeveci@gmail.com

[1] School of Information Science, Guangdong University of Finance & Economics, Guangzhou 510320, China

[2] School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China

[3] School of Information Science and Technology, Jinan University, Guangzhou 510632, China

[4] Department of Industrial Engineering, Turkish Naval Academy, National Defence University, 34942 Tuzla, Istanbul, Turkey

[5] The Bartlett School of Sustainable Construction, University College London, 1-19 Torrington Place, London, WC1E 7HB, UK

[6] Department of Electrical and Computer Engineering, Lebanese American University, Byblos, Lebanon

[7] Department of Computer Science and Engineering, Mississippi State University, Starkville, MS 39762, USA