Using deep learning models in image processing

Xiong, Z.

Chapter 2

A Survey of Deep Learning in Image Processing: Developments, Applications and Challenges

Journal Artificial Intelligence Review (under review).

Chapter Summary

The rapid advancement of deep learning models (DLMs) in computer vision is driven by their capacity to unify feature representation and task inference into end-to-end automated frameworks, thereby surpassing manual feature engineering. This survey categorizes DLMs into Convolutional Neural Networks (CNNs), Visual Transformers (ViTs), and hybrid models based on their core components (convolutional vs. attention-based), systematically reviews their technical evolution, and critically analyzes their strengths and limitations. CNNs, the pioneering paradigm, leverage the locality of convolution kernels for efficient hierarchical pattern extraction. However, their reliance on multi-layer stacking to aggregate global context and rigidity to geometric variations has motivated innovations such as deformable convolutions to expand receptive fields. ViTs, emerging later, prioritize the globality of attention mechanisms to model long-range dependencies directly, but face computational bottlenecks and noise sensitivity, which are addressed by sparse attention and hierarchical tokenization. Hybrid models now dominate as optimal compromises, combining the locality of CNNs and globality of ViTs to balance efficiency and expressiveness. Moreover, we emphasize universal techniques—neural architecture search (NAS), model compression, and self-supervised learning—as indispensable tools to enhance robustness and adaptability across architectures. Despite progress, challenges persist in interpretability, data efficiency, and real-world robustness. We advocate prioritizing lightweight hybrid designs, explainable attention-convolution combinations, and cross-modal systems integrating more data beyond vision. This survey not only systematically reviews the technical evolution of DLMs but also proposes a forward-looking roadmap for adaptive, scalable, and trustworthy vision systems, grounded in current technological trajectories.

Keyword: Machine Learning Deep Learning Convolutional Neural Network Vision Transformers CNN-Transformer Architectures

2.1 Introduction

The rapid advancement of machine learning (ML) has profoundly reshaped modern society, embedding itself into nearly every aspect of daily life. From personalized recommendations in e-commerce and content moderation on social platforms to real-time language translation and autonomous decision-making systems, ML technologies have become indispensable tools for fully exploiting the value of vast amounts of data (big data) and unleashing their immense potential. At its core, the capacity of ML systems to automatically discover hidden patterns in multi-modal inputs (i.e., text, images, audio, and sensor data) has exceeded the limitations of traditional approaches and advanced the computational boundaries to complex, high-dimensional information. Among the diverse sub-fields, computer vision stands out as a cornerstone due to its direct relevance to automating the interpretation of visual data.

In computer vision, traditional systems relied heavily on handcrafted feature engineering, in which experts manually designed filters (e.g., edge detectors, texture descriptors) to extract structure-dependent representations for task inference. While friendly to interpretability, these methods lack adaptability to inherent variations and complexities present in real-world data, such as lighting variations, occlusions, and deformable objects, hindering generalization across data structures or distributions. For instance, conventional techniques for lesion detection or cellular segmentation in biomedical imaging demand laborious parameter tuning to accommodate diverse tissue structures and imaging modalities. By contrast, ML-based approaches, particularly deep learning (DL) methods, circumvent these limitations by automating the construction of feature representations directly from data. By training on large-scale datasets, DL networks have achieved superior performance across diverse tasks like object detection, image segmentation, data simulation, drug discovery, and pathological diagnosis.

The explosive growth of the literature on deep learning for computer vision poses a unique challenge: organizing decades of research into a clear, logical structure that captures the core contributions, advancements, and limitations in the field. Existing surveys often categorize works by application domain or chronology, but this approach risks obscuring the underlying evolution of model design principles. To address this, we propose a novel organizational perspective rooted in the mathematical foundations of core computational components, namely *convolution kernels* and *attention mechanisms*. By dissecting their formulations, we systematically reveal the inherent strengths and limitations of Convolutional Neural Networks (CNNs) and Visual Transformers (ViTs), as well as their interconnections, paving the way for the foreseeable emergence of hybrid

models that combine their advantages.

Convolutional operations, characterized by fixed receptive fields and weight sharing, excel at capturing localized spatial patterns through inductive biases like translation equivariance. These properties make CNNs highly efficient for processing complicated grid-structured data via a hierarchy of convolutional layers, while maintaining computational efficiency and good generalization performance, particularly on relatively small datasets. However, their rigid receptive fields and static kernel parameters hinder modeling of long-range dependencies and adaptation to complex geometric variations. Attention mechanisms, conversely, employ global receptive fields and compute long-range pairwise interactions across all input elements, thereby capturing the so-called global context. This flexibility allows ViTs to dynamically adjust weights based on the input and to benefit from global reasoning, such as in panoptic segmentation or cross-modal retrieval, but at the cost of quadratic computational complexity, sensitivity to noisy tokens, and data-hungry.

Compared to previous surveys, our survey proposes a novel organizational structure grounded in the mathematical and computational foundations of the models. The deep mathematical kinship naturally motivates the design principle of hybrid architectures that use flexible dynamic aggregation schemes across different feature scales, balancing convolutional locality with attentional globality for efficiency and expressiveness. For instance, deformable convolutions that learn adaptive yet sparse sampling grids can capture non-local representations from high-resolution feature maps with small overhead, whilst sparse attention mechanisms can model dynamic long-range correlations from low-resolution feature maps to support customized global inference for all input data. Such connections underscore the inevitability of hybrid designs, where complementary inductive bias of convolutional kernels enables efficient task-agnostic extraction of commonality (i.e., building a shared feature space), whilst the dynamic global contexts in attention mechanisms support flexible task-specific extraction of individuality (i.e., performing input-dependent task inferences), to tackle complex vision tasks.

In addition, this survey extends beyond mere taxonomy to encompass universal advancements, including neural architecture search, model compression, self-supervised learning, diffusion methods, and deep unfolding techniques. These can improve the overall performance of DL models, transcending architectural boundaries. Moreover, we illustrate the remarkable progress achieved in visual applications, including generic computer vision tasks (e.g., classification, detection, data generation) and a dedicated section 2.4.5 for medical data analysis. Given limited annotations, multi-modal data, and high-reliability requirements, it is worth paying attention to specialized archi-

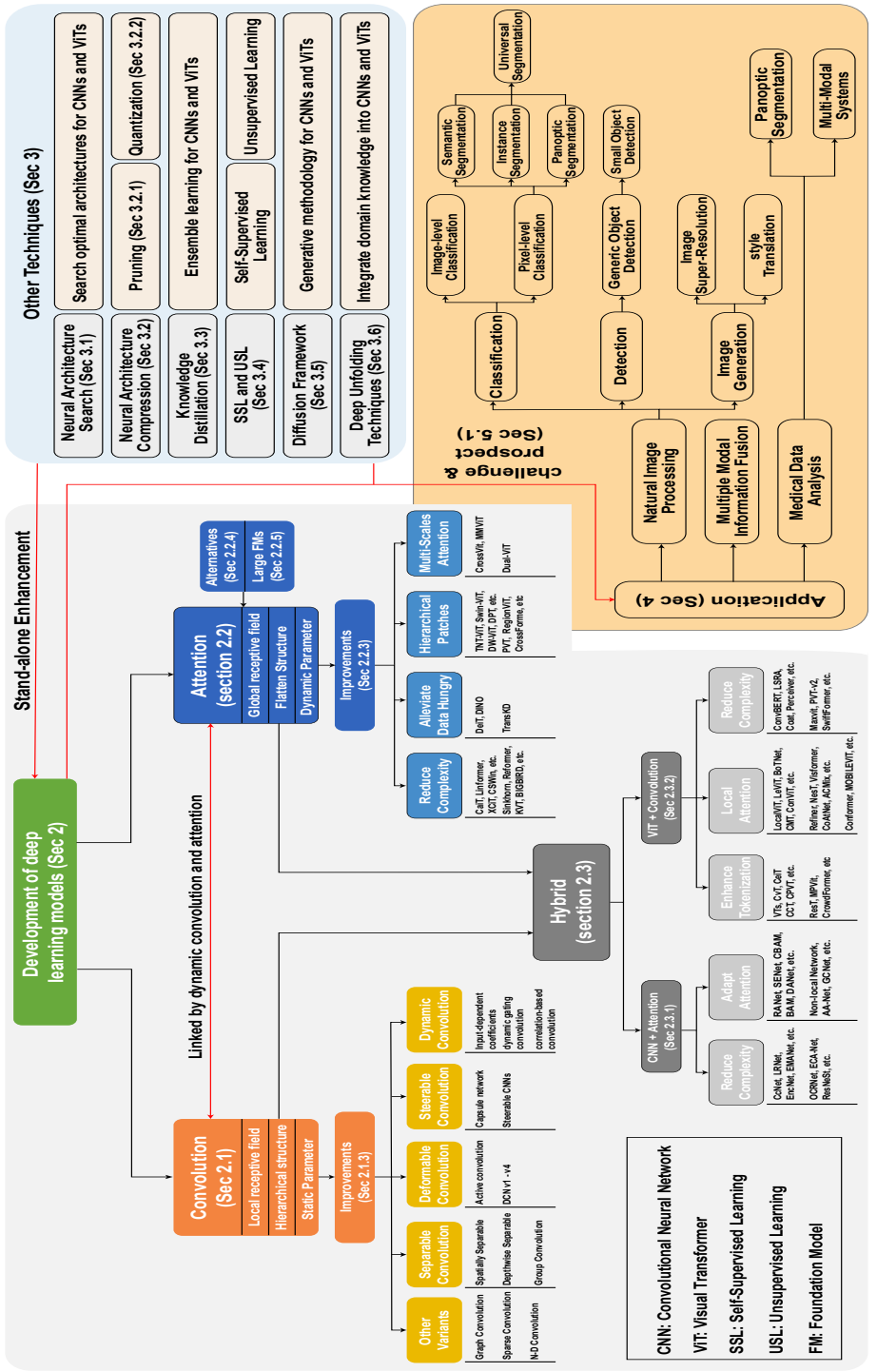


Figure 2.1: A simple visualization of the organization of the survey.

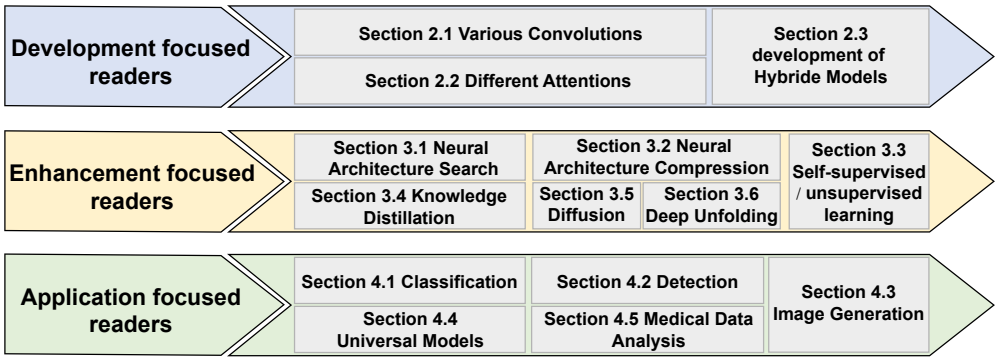


Figure 2.2: Text roadmap for each of the specific sections of the survey.

tectures and learning paradigms for medical vision tasks. Furthermore, we explore current challenges and future research directions. From the perspectives of lightweight model compression, interpretability enhancement, and multi-modal fusion, we delineate possible technical pathways for designing scalable, interpretable, and robust computer vision systems.

In conclusion, the survey presents a comprehensive architectural coverage. It provides a systematic review of the technical evolution, strengths, and limitations of the three major paradigms: CNNs, ViTs, and Hybrid models for organizing decades of research and revealing the underlying evolution of design principles, as depicted in Figure 2.1. To facilitate survey perusal, we have provided a text roadmap (Figure 2.2) to quickly locate the key sections.

2.2 Developments

Deep learning [257, 349] is a specific kind of machine learning. From a systematic perspective, a deep learning system typically has two phases: training and testing. At the training stage, a system consists of four components - a dataset that feeds valid raw data into the system, a representation model extracting machine-oriented discriminative features, a task-specific estimation metric (also known as loss/cost/object function) to measure the error from machine predictions to human expectations, and an optimization algorithm for automatic parameter tuning - while at the testing stage, it freezes all hidden trainable parameters and is ready to use. For improved representational capabilities, extracting features across multiple scopes and fusing information at different scales are essential. In fact, the evolution of deep architectures

in computer vision is fundamentally driven by the pursuit of an optimal balance between local and global feature aggregation, and between static and dynamic parameterization. Currently, there are three main architectures for image processing in computer vision: Convolutional Neural Networks (CNNs) established the initial paradigm, leveraging local, static kernels for efficient hierarchical feature extraction; Visual Transformers (ViTs) shifted the paradigm by introducing global, dynamic attention to capture long-range dependencies directly; Subsequently, Hybrid architectures emerged as a synthesis, strategically combining convolutional locality with attentional globality to achieve a more efficient and expressive balance. In this section, we elaborate on the progression and explicitly connect each architectural innovation to its underlying mathematical principles.

2.2.1 Convolutional Neural Networks

CNN is a feed-forward neural network that can extract features from data using convolution operators, as shown in Figure 2.3. Unlike traditional machine learning methods that rely on manual feature construction, CNN automates feature extraction by convolving data with a stack of differentiable composite functions. If each function denotes an artificial neuron, the extraction process is analogous to visual perception [349], where each neuron can respond to various features and transmit the information to the next neuron.

Foundations of CNNs

The basic module in a CNN comprises mainly four operations, with optional functions: a convolution operation, a normalization operation, an activation operation, and an optional pooling operation, along with additional residual connections or random dropouts.

Standard convolution

Standard convolution operations are the most fundamental elements in neurons, designed to process grid-structured data (e.g., images and videos). Originally developed from signal processing, convolutions were initially designed to detect patterns with predefined kernel weights. In deep learning, however, they have evolved into learnable feature extractors, in which kernel weights are optimized via backpropagation based on the input data. The essence of a convolution involves sliding a small filter over the input. The kernel performs pairwise multiplication between kernel weights and

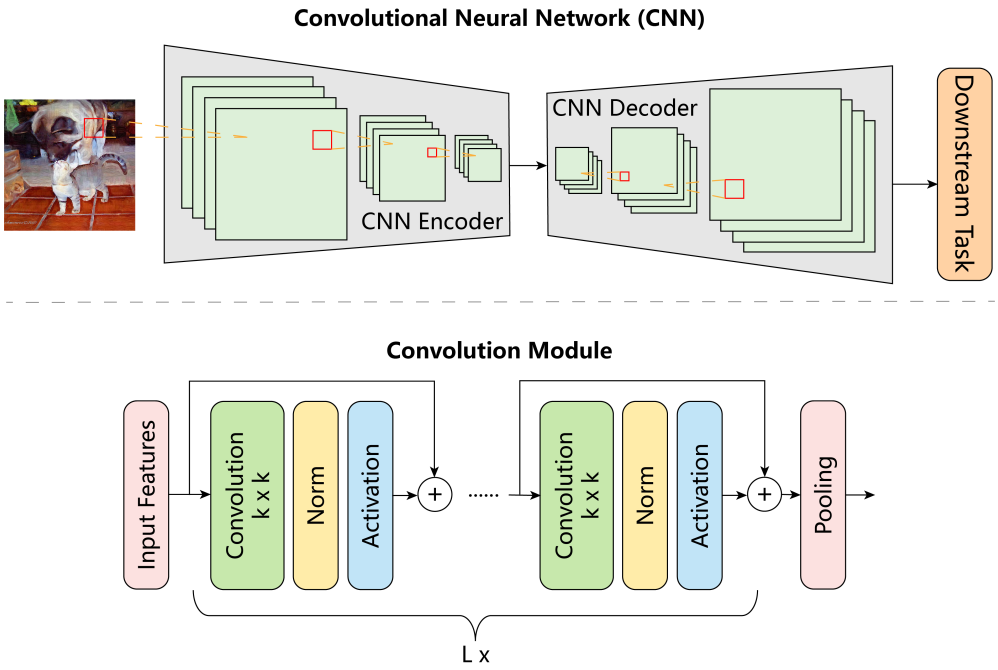


Figure 2.3: Schematics graphical representation of a generalized framework for a Convolutional Neural Network (CNN) with **L** blocks.

the corresponding input elements at each sliding position. This process captures local patterns and relationships within a local field, enabling the model to acquire essential features.

Mathematically, a generalized standard convolution [1] in machine learning works as a sliding-window operation over the input feature maps. For clarity, we begin with a special case of a generalized 2D convolution and then extend the definition to higher dimensions. Consider an input feature map $X \in \mathbb{R}^{H \times W \times C_{in}}$, where H and W are the spatial height and width, and C_{in} is the number of input channels. A generalized 2D convolution contains C_{out} learnable kernels and **independently** applies each kernel to produce one channel of the output feature map $Y \in \mathbb{R}^{H' \times W' \times C_{out}}$. Therefore, focusing on the dynamic process of the arbitrary kernel f_c benefits understanding its hyperparameters. Depending on different roles, all arguments can be divided into three groups: 1) stride s decides the sliding displacement from the previous window center to the next; 2) padding size p [2] controls the valid range of sliding window centers

[1] To show the geometrical characteristics of convolutions, dilated convolutions are also included as one type of standard convolutions in this paper.

[2] Although different padding methods depend on the various boundary conditions, we argue that

along the boundary to avoid discarding edge information; 3) kernel size k (usually odd), dilation rate $d \geq 1$, and input channels C_{in} jointly decide a 3D window for local operation.

a. Window Sliding with boundary condition

To preserve edge information, the input X is often extended by adding p rows/-columns of values (normally 0) around its borders. Then the window size k, step length s, and the padding size p jointly defines a function to map the sliding window centers (x, y) from the coordinates (x', y') on the original input:

$$x' = p + x \times s + (k-1)/2, \qquad y' = p + y \times s + (k-1)/2 \tag{2.1}$$

Likewise, considering the stride s and padding size p, the output size are given by:

$$H' = \lfloor \frac{H + 2p - d \times (k-1) - 1}{s} + 1 \rfloor, \qquad W' = \lfloor \frac{W + 2p - d \times (k-1) - 1}{s} + 1 \rfloor \tag{2.2}$$

Common padding schemes are valid when $p = 0$ and same (chosen p to get $H' = H$ and $W' = W$ when $s = 1$).

b. Local window operation

It is common to overlook that each 3D kernel defines only a 2D grid, since the grid is shared isotropically across all channels. Denote an arbitrary kernel as a 3D weight tensor $f_c \in \mathbb{R}^{k \times k \times C_{in}}$, then the window is defined by a regular sampling grid $\mathcal{R}$ that contains k^2 offsets stretching by dilation rate d:

$$\mathcal{R}_d^k = \left\{ (d \cdot i, d \cdot j) \middle| i, j \in \left\{ -\left\lfloor \frac{k}{2} \right\rfloor, \ldots, \left\lfloor \frac{k}{2} \right\rfloor \right\} \right\} \tag{2.3}$$

where $\lfloor \cdot \rfloor$ denotes a floor function. Let each element $(\Delta x, \Delta y) = (d \cdot i, d \cdot j) \in \mathcal{R}$ corresponds to one offset point, centered at $(0, 0)$. When $d = 1$, the offsets are the integer coordinates of a $k \times k$ square centered at $(0, 0)$, inducing a standard dense $k \times k$ grid. When $d > 1$, the kernel samples are spread out, producing a dilated (sparse) sampling pattern that effectively enlarges the receptive field without increasing the

they do not influence the mathematical representation.

number of parameters (e.g., in dilated convolution). Inside the window, the input values are weighted by the kernel weights and summed across all input channels:

$$Y(x, y, c_{out}) = \sum_{(\Delta x, \Delta y) \in \mathcal{R}_d^k} \sum_{c_{in}}^{C_{in}} X(x' + \Delta x, y' + \Delta y, c_{in}) \cdot f_c(\Delta x, \Delta y, c_{in}) \tag{2.4}$$

Notably, the kernel applies the same grid isotropically across the spatial dimensions: the same set of offsets is used at every window center, but the summation extends over all input channels. Thus, the kernel is inherently **a channel-wise aggregation followed by spatial sliding**, which can serve as the foundation for depthwise separable convolution.

Other Components

Pooling operation is necessary for CNNs because it can expand the receptive field of the convolution without a significant computational overhead. A pooling (i.e., down-sampling) operation partitions the input feature map into multiple regions, and each region is pooled into a statistical summary. With fixed receptive fields, successive convolutional operations can effectively search over larger spatial scopes using reduced feature maps. The most commonly used pooling functions are max pooling [858] (selecting the mode of one region) and average pooling [41] (selecting the mean of one region). Therefore, pooling operations enable the hierarchical structure of CNNs and introduce a strong inductive bias that the learned feature maps must approximately satisfy the slow-variation assumption. Otherwise, a statistical summary (e.g., mode or mean) cannot precisely describe the sampling region. Applying pooling techniques can significantly reduce computational complexity and accelerate model training. There is some theoretical research [40, 306] that gave a detailed analysis of pooling operations.

Normalization operation is a crucial technique in CNNs that standardizes feature vector distributions to have zero mean and unit variance. Distribution standardization can reduce internal covariate shifts between layers induced by parameter updates. Consequently, each layer will operate under consistent conditions, stabilizing the training process. In addition, normalizing feature vector magnitudes to a more compact range can significantly reduce the influence of extreme-value outliers. Without extreme features, normalization can mitigate vanishing (too small) / exploding (too large) gradients and alleviate overfitting by preventing feature learning from relying too heavily on outlier features with extreme values. Therefore, these are crucial for fast convergence

and generalization, especially in deep architectures. Common normalization techniques in CNNs are designed to standardize along different dimensions for various application scenarios. Batch Normalization [288] computes the mean and variance within every mini-batch and its performance is dependent on the batch size; Layer Normalization [19] normalizes across all feature channels of one feature map independently of the batch; Instance Normalization [634] normalizes at one feature channel within one feature map independently; Group Normalization [701] normalizes groups of feature channels within one feature map, providing a balance between batch and instance normalization.

Activation operation plays a crucial role in introducing nonlinearity into CNNs. Without non-linear transformations, a stack of linear transformations would essentially act as a single linear transformation, limiting the ability to learn complex patterns and relationships in data [552, 553]. For this reason, activation functions are applied to each neuron's output, and non-linear transformations are introduced before the input of the next neuron. Similar to the function of a neuron in the human brain, an activation function acts as a stimulus between two neurons, controlling the transmission of information from one to the other. By stacking many convolutional layers and activation functions, deep neural networks can approximate any function, thereby greatly enhancing their ability to fit data. Within CNNs, there are mainly two types of typical activation function types: 1) map functions that non-linearly transform the input to different representation spaces, including sigmoid, tanh, and softmax; 2) gate functions that filter out irrelevant information include Rectified Linear Units (ReLU) [487] and its variants like Parametric ReLU [246], Randomized ReLU [730], Gaussian-error linear unit (GELU) [251], SiLU [478], Exponential Linear Unit (ELU)[118], Squareplus [28]. In summary, activation functions enable CNNs to model intricate patterns and decision boundaries through nonlinearity, thereby enhancing their performance across various tasks.

Residual connection, also known as a skip connection, is a shortcut that allows a layer to bypass multiple intermediate layers and connect directly to the computation of the destined layer. This simple modification has profound implications for addressing the degradation problem, where increasing the network depth leads to worse performance due to training difficulties. In very deep networks, gradients can become vanishingly small during backpropagation, making it difficult to train the early layers. Residual connections provide a direct path for gradients to flow, stabilizing the training process. Originating from ResNet [248], the residual connection enables the deep structures

in CNNs. Recently, residual connections have also been successfully applied to other types of neural networks, such as recurrent neural networks (RNNs) and transformers.

Dropout facilitates regularization in the network by randomly omitting some units or connections with a predetermined probability. This random dropping of connections or units yields different sub-network architectures, from which one combination can be interpreted as an approximation to the mixture-of-experts [594]. Each expert can generate weak predictions, and the ensemble of all experts can eventually enhance generalization and mitigate overfitting.

Characteristic of CNNs

Within a CNN, each standard convolutional layer extracts information from a neighborhood $\mathcal{R}$ of sampled locations and is responsible for detecting a distinct pattern in the input. During training, the model undergoes backpropagation and gradient descent to learn optimal convolutional filters. The essence enables the model to discern meaningful patterns in the data for feature representation automatically. From the perspective of extracting features at different levels, the mathematical definition of standard convolution forces the design of CNNs that must be characterized in the following three respects:

1. **Local receptive field**: Each neuron is connected only to a few neighboring neurons based on the sampling locations $\mathcal{R}$. This design promotes the extraction of short-term correlations because each neuron learns a specific pattern within its local scope. Local receptive fields are computationally efficient, focusing on small regions to capture local details.

2. **Hierarchical structure**: Extracting long-term dependencies by convolutions with increasing receptive field is computationally expensive. A rational alternative is to use a hierarchical structure that can gradually decrease the sizes of feature maps stage-wise via stride convolutions or pooling operations. With significant computational overhead, convolutions can sample from larger effective scopes, but at the cost of losing spatial information. Consequently, this architecture can exhibit different features from local to global, but it introduces a strong inductive bias: the learned feature maps must approximately satisfy the slow-variation assumption. Otherwise, a statistical summary (e.g., mode or mean) cannot precisely describe the sampling region.

3. **Static Parameter**: the weights in convolutional kernels are static because the

same kernel is applied across all sliding locations. This weight sharing reduces the number of parameters and accelerates computation, but introduces a strong inductive bias toward translation equivariance. The bias means that if the input feature is translated, the output changes accordingly. In other words, pattern detection is not related to spatial locations. Using static parameters enables training with less data.

Improvements of CNNs

However, standard convolutions face limitations in computational efficiency, geometric transformations, and adaptability to complex geometric variations. To address these limitations, recent works focus on three directions: (1) reducing computational complexity via separable convolutions, (2) enhancing geometric adaptability through deformable kernels, and (3) improving dynamic feature aggregation with steerable or dynamic convolutions. We organize these improvements by technical focus and highlight their potential application scenarios.

Separable Convolution

Separable convolution is a convolutional operation in deep learning that aims to reduce computational complexity by decomposing the convolution into a product of simpler matrices. They are particularly well-suited to scenarios with limited computational resources, such as mobile devices or real-time applications. By factorizing larger kernels into multiple smaller kernels, a separable convolution offers a compelling trade-off: significantly reduced computational cost and model size while maintaining acceptable performance. According to different splitting schemes, there are two main types of separable convolutions: spatially separable and depthwise separable convolutions.

Spatially separable technique decomposes a single standard convolution into a series of lower-dimensional convolutions on different spatial dimensions. This tensor decomposition can be achieved using tensor decomposition techniques, like Singular Value Decomposition (SVD) [142, 291, 143, 824, 392], Tucker decomposition [325, 412, 337], CP decomposition [347, 14, 852], and other decomposition methods [497, 662, 765, 569, 353, 651]. Given a set of decomposed components, a standard kernel can be expressed as the tensor product of multiple smaller kernels. For instance, a spatially separable 2D convolution kernel with spatial dimensions $F \times F$ can be split into two 1D kernels: $F \times 1$ and $1 \times F$. By applying these two filters sequentially, the spatially separable convolution achieves the effect of a full $F \times F$ convolution but

with significantly lower complexity: reducing from $\mathcal{O}(F^2)$ multiplications down to $\mathcal{O}(F)$. The main advantage is that there is no performance degradation after kernel separation, unless the matrix is decomposable. However, spatial separable convolutions have limited application in CNNs because not all convolution kernels are separable. Therefore, this decomposition may narrow the search for all possible kernels during training, leading to a performance trade-off. Hence, the applications of spatial separable convolution are significantly limited.

Depthwise separable technique is an efficient alternative that splits tensor computations of multiple convolutions into two steps: depthwise convolutions and pointwise convolutions. Depthwise convolutions apply a separate kernel to each input channel, capturing spatial patterns independently for each channel. Pointwise convolutions then use 1×1 convolutions to combine the output channels from the depthwise step, effectively aggregating the information. Depthwise separable convolutions significantly reduce the number of parameters and computation while maintaining model performance, making them popular in mobile and embedded applications. Therefore, the decomposition does not rely on the special properties of kernel functions. By decoupling the spatial filtering from cross-channel filtering, depthwise convolution achieves higher computational efficiency, reducing complexity from $\mathcal{O}(F^2 \cdot C_{\text{out}})$ to $\mathcal{O}(F^2 + C_{\text{out}})$. F is the kernel size and C_{out} denotes the channel number of the output feature maps. Depthwise separable convolution [211, 612, 417, 132], therefore, is well-suited for resource-constrained environments. The variants of MobileNet [265, 563, 264], ShuffleNets [823, 457], and Xception [112] are popular CNN architectures that use depthwise convolution to reduce model size and improve inference speed without significantly compromising performance.

Group convolution is another solution to the depthwise separable technique that divides depth channels into a group of smaller filters based on the assumption that each group of filters is mutually independent. Therefore, it simplifies the original full-matrix multiplication to a diagonal block-matrix multiplication along the depth dimension. Therefore, sparsity in the off-diagonal blocks improves efficiency and reduces the complexity by a factor of g, where g is the number of groups. Group convolution was first applied to AlexNet [340] to reduce the number of parameters. Then, ResNeXt[716] overlaps smaller convolution blocks to approximate ResNet's convolution blocks. Moreover, CondenseNet [276] introduced sparsity-inducing regularization and divided group convolution training into several steps. However, previous methods require manual group selection. GroupNet [834] combines network architecture search (NAS) with group

convolution, enabling automatic learning of the number of groups via an end-to-end approach.

Deformable Convolution

Deformable convolution is an advanced variant of standard convolution that extends the receptive field at low computational cost. In segmentation or detection tasks, it is crucial to pay closer attention to the contents within semantic / object regions. In other words, not all locations in a receptive field contribute equally to an output response. Locations within regions are much more important for sampling than those outside the region. Therefore, uneven sampling requires convolution operations to account for geometric shapes. Constrained to the regular structure of the receptive field, standard convolution evenly samples at fixed locations and, therefore, cannot efficiently tackle geometrical variations. That means a stack of standard convolutions can linearly expand their theoretical receptive field scope to cover arbitrary geometrical shapes with increasing computational complexity. In contrast, deformable convolution enhances geometric modeling by learning input-dependent sampling offsets. The core idea is to transform the shape of receptive fields from a regular grid to arbitrary shapes. Specifically, deformable convolutions define sub-networks to learn additive disturbances $(\Delta p_x, \Delta p_y)$ alongside the static offset $(\Delta x, \Delta y)$ in a regular grid (refer to Eq.2.3). Then adaptive offsets can be generated as: $\Delta x' = \Delta x + \Delta p_x$ and $\Delta y' = \Delta y + \Delta p_y$. Therefore, a static convolution kernel has been replaced with spatially variant operators that use input-dependent geometric information to adaptively sample at more important locations. Under the assumption of local continuity in feature representation, the deformable convolutions have dynamic, receptive fields which are equivalent to larger receptive fields with sparse non-zero elements in standard convolutions. Compared to standard convolution with kernel size k, deformable convolution incurs an additional cost for offset prediction (equivalent to one extra $k \times k$ convolution). It requires at least bilinear interpolation for sampling at non-integer positions. The total complexity is approximately 2 times that of a standard convolution. However, by enabling adaptive receptive fields, it often achieves higher accuracy with fewer layers, offering favorable trade-offs between efficiency and accuracy in tasks that require geometric flexibility, such as object detection and segmentation. Active convolution [300] was an early work to augment sampling locations with learnable offsets. However, the offsets were static model parameters shared across different spatial locations, making them less general and less adaptive. DCNv1 [127] extends the sampling offsets as a dynamic output from

other sub-networks, allowing offsets to be adaptive to different geometrical shapes at different locations. Subsequently, DCNv2 [869] introduced an extra learnable modulation scalar for each offset, allowing it to adjust the importance of the sampled location, thereby providing additional flexibility. To reduce computational cost, DCNv3 [672] introduced depthwise separable convolution to compute modulation scalars. Besides, it splits deformable convolution into groups and normalizes the modulation scalars at each sampling point using the softmax function. Each group has its own sampling offsets and modulation scalars, and can have different spatial aggregation patterns. Therefore, it can enhance feature extraction by ensembling different patterns with normalized modulation scalars. Recently, DCNv4 [723] aims to speed up the computation by removing softmax normalization and optimizing memory access to minimize redundant operations.

Steerable Convolution

Steerable convolution is a convolutional operation used in computer vision to handle geometric transformations, such as rotations, efficiently. Unlike standard convolutions, which must use large amounts of static filters to enumerate all possible orientations, steerable convolutions aim to construct a vector space to represent arbitrary orientations. Inspired by the mixture-of-experts' architecture [255] in capsule networks [256, 2], steerable convolutions also encode the rotation of a pattern in the input as a linear combination of basis kernels. Normally, the combinatorial coefficients that are a set of learnable parameters can be interpreted as dynamic routines [559, 220]. At the same time, the disentangled basis filters can be understood as expert sub-networks dedicated to different geometrical transformations. Therefore, the steer convolutions define a representation space in which each orientation or transformation can be decomposed into (normalized) coordinates of basic pattern units. Cohen et al. [119] proposed steerable convolutional operations based on group representation theory. They designed equivariant filter banks and represented geometrical variations as linear transformations in a group. The constructed space increases the flexibility of equivariance at low computational complexity. Based on the 2-dimensional steerable convolution, Weiler et al. [690] proposed the 3-dimensional steerable convolution that maps rigid body transformations in 3-dimensional Euclidean space onto the representation space. Besides, the Euclidean group E(2) [689], and its subgroups are given as general solutions to the kernel space to adapt to the rotation and reflection of planar images. In summary, steerable convolutions are suitable for orientation / pose-sensitive scenarios, including

biomedical image analysis, 3D object recognition, and object detection.

Dynamic Convolution

Dynamic convolution is a more general concept in CNNs that enhances feature representation without significantly increasing computational cost. Instead of using a large set of static kernels to enumerate all possible feature representations, dynamic convolution defines generation routines of feature expression. It dynamically selects appropriate kernels for the input mini-batch. The dynamic characteristic implies constructing implicit representation spaces to handle more complicated contexts, thereby compressing redundancy in the set of static kernels. According to different generation methods, there are mainly three types of dynamic convolutions: 1) One of the most used types is to generate input-dependent aggregation coefficients over parallel fixed convolution kernels. This type of dynamic convolution still satisfies the **parameter sharing** constraint, i.e., it applies the same dynamic kernels to all locations. Inspired by the architectures of mixture-of-experts, the linear combination coefficients are obtained by different dynamic route functions running on the input [743, 97, 829]. Noted, capsule networks and steerable convolutions can also be regarded as special cases of dynamic convolution with dynamic combinatorial coefficients. 2) The second popular type is to generate gating information that can guide convolution to execute only on selected locations, extending the property of **parameter sharing**. Normally, gating information is generated by sub-networks and represented as binary masks, where non-zero values indicate the importance of each sampling location. Therefore, the gating masks induce sparsity in the data, thereby greatly reducing computational complexity. Spatially dynamic gating convolutions [180, 643] exploit the sparsity in the spatial domain and only execute conditionally on important selected regions. Likewise, depthwise dynamic gating convolutions [268, 598] account for the importance of different feature channels and selectively train only the relevant channels across various inputs. Consequently, combining spatial and depthwise gating information is possible for more generalized dynamic gating convolutions [381, 364]. Noted, deformable convolutions with modulation scalars can be categorized into the spatially dynamic gating convolution. 3) The last type is to directly use the output feature maps from other (sub-)networks as the kernel contents [305, 130]. Therefore, running the dynamic content convolutions over all locations of the input feature maps can be interpreted as collecting correlation information between two feature maps [460, 459]. Overall, dynamic convolution offers the most flexible design for convolution operations,

leveraging input-dependent adaptability to enhance model performance.

Other variants

Other variants include convolutions that emphasize application scope rather than improvements to the definition. We briefly list the following application directions of convolution operations:

Graph convolution is a generalized operation that extends the concept of convolution from Euclidean space to non-Euclidean space. Specifically, irregular data structures can be represented as graphs, where nodes represent entities and edges imply relationships. One of the most important advantages of graph structure is that it can efficiently represent complicated mutual correlations via an adjacency matrix. Therefore, graph convolution simplifies aggregating information from connected edges around each node and the learning of intuitive and scalable node representations at the graph level. Deep learning models that stack multiple graph convolution operators are called Graph Convolutional Networks (GCNs) [543]. GCNs can effectively discover inherent data structures and capture complex relationships among nodes, whereas traditional convolutional operations struggle to do so. GCNs have found applications in various fields, including social network analysis, drug discovery, and recommendation systems. The main challenges of GCNs are multiple folds: (a) Scalability, where large graphs result in high computational complexities. (b) Static nature that cannot adapt to evolving graph structures. (c) Limited expressiveness that hinders the capture of complex graph structures, especially degrading and over-smoothing node identifiability as graph convolution layers deepen. (d) Limited generalization that GCNs cannot transfer their information from one graph structure to another, especially between heterogeneous graphs.

Sparse Convolution is a specialized convolution operation designed to handle sparse data where most input values are zero efficiently. It allows computations to be performed only on non-zero data points, reducing computational and memory costs compared to traditional dense convolutional operations. Sparse convolution is typically implemented in three steps: (1) build an index system to store locations of non-zero values. (2) Build a RuleBook to indicate how to accumulate the values of pairwise multiplication between non-zero elements and corresponding kernel weights. (3) Execute all computations in parallel based on the instructions in the RuleBook. Therefore, sparse convolutions are often adopted for high-resolution images (i.e., medical & microscope images) or

3D data (i.e., point clouds). The main advantage of sparse convolutions is that they extract information from sparse data with high speed and low memory usage, since the overhead of the construction of the RuleBook can be ignored. SCNN [420] applied a two-stage decomposition over the channels and convolutional kernels and converted them into sparse matrix multiplication. However, the main disadvantage is their limited generalization, which imposes an ambiguous restriction on data sparsity. Therefore, the choice between sparse and traditional convolution depends on the specific application and the data characteristics.

Generalization to Higher Dimensions

The principles of 2D convolution naturally extend to higher dimensions, enabling deep learning models to process volumetric data such as 3D medical scans (CT, MRI), 3D/4D microscopy, video sequences, and point clouds. Let $X \in \mathbb{R}^{H \times W \times D \times C_{in}}$ denote a 3D input volume, where D denotes the depth (or temporal) dimension. A 3D convolution kernel is a 4D tensor $f_c \in \mathbb{R}^{k \times k \times k \times C_{in}}$ for each output channel c with kernel size k, and the sampling grid $\mathcal{G}$ expands to three spatial dimensions:

$$\mathcal{G}_d^k = \left\{ (d \cdot i, d \cdot j, d \cdot l) \middle| i, j, l \in \left\{ -\left\lfloor \frac{k}{2} \right\rfloor, \ldots, \left\lfloor \frac{k}{2} \right\rfloor \right\} \right\} \tag{2.5}$$

where $d \geq 1$ is the dilation rate (typically uniform across axes for isotropic data). The stride s and padding p are defined along all three dimensions. The coordinate- and size-mapping formulas are analogous. The output at location (x, y, z) is computed as:

$$Y(x, y, z, c_{out}) = \sum_{(\Delta x, \Delta y, \Delta z) \in \mathcal{G}_d^k} \sum_{c_{in}}^{C_{in}} X(x' + \Delta x, y' + \Delta y, z' + \Delta z, c_{in}) \cdot f_c(\Delta x, \Delta y, \Delta z, c_{in}) \tag{2.6}$$

3D CNNs have become instrumental in medical image analysis and video understanding. For example, 3D U-Net [117] extended the successful U-Net architecture to volumetric segmentation by replacing 2D convolutions with 3D ones, enabling the model to leverage inter-slice context, which is crucial for accurate organ or lesion delineation. In video action recognition, C3D [628] demonstrated that straightforward 3D convolutional networks can learn powerful spatiotemporal features directly from raw video clips. However, processing 3D data with standard 3D CNNs is often computationally expensive. Several efficient architectures [717] have been developed for efficiency. Sparse 3D Convolutions [420, 204] mined spatial sparsity in the high-dimensional data and

computed operations only on non-zero voxels, dramatically reducing computation. Beyond dense volumetric grids, irregular 3D data, such as point clouds and meshes, require specialized architectures because of their non-Euclidean structure. Instead of voxelizing point clouds, Point Cloud Networks directly processed unordered point sets. PointNet [525] and PointNet++ [526] used shared multilayer perceptrons (MLPs) and symmetric pooling functions to achieve permutation invariance. 3D Graph Neural Networks [327] modeled point clouds as graphs, where edges represent local neighborhoods, and message passing aggregates information. Moreover, separable 3D Convolutions [717], inspired by 2D depthwise separable convolutions, have factorized the 4D kernel into a 3D spatial convolution (per channel) followed by a $1 \times 1 \times 1$ pointwise convolution, reducing parameters and computation. Likewise, to handle irregular geometries in volumetric data, deformable convolutions [127] have been extended to 3D, allowing the sampling grid to adapt to the data shape.

3D computer vision tasks often require tailored loss functions. The Dice loss [476] is widely used in medical image segmentation to handle class imbalance. For point cloud reconstruction, Chamfer distance [169] and Earth Mover's distance [557] measure the similarity between two point sets. In summary, extending deep learning to 3D and volumetric data introduces unique challenges in computational efficiency and representation. While 3D CNNs serve as a direct generalization of 2D convolutions to dense, grid-structured volumes, alternative architectures like GNNs and point-based networks are better suited to irregular 3D structures, enabling impactful applications across multiple domains. Future directions may include more efficient 3D attention mechanisms and hybrid models that combine the strengths of CNNs, GNNs, and Transformers for 3D data.

CNN Summary

CNNs are characterized by local receptive fields, a hierarchical structure, and static kernel parameters. These inductive biases grant them remarkable data efficiency and hardware-friendly computation. However, their fundamental limitations in modeling long-range interactions and adapting to complex geometric variations motivate the search for architectures with global receptive fields and dynamic aggregation—principles that form the foundation of the next architectural paradigm.

2.2.2 Visual Transformers

Transformer models originate from natural language processing (NLP) tasks and outperform recurrent neural networks (RNNs) [345] in modeling element-wise correlations within input sequences. Unlike the Long Short-Term Memory (LSTM) mechanism in RNNs, transformers can ignore spatial distances and directly model long-range pairwise interactions via self-attention. Owing to their strong representational capabilities, transformers have intrigued the vision community to adapt their architectures to computer vision tasks. The variants applied in computer vision have performed as well as, or better than, CNNs. To overcome the inherent locality and staticity of CNNs, Visual Transformers (ViTs) adopt a different design philosophy. By replacing convolution with a global self-attention mechanism, ViTs directly model global pairwise dependencies among all input tokens, without a hierarchical structure, enabling dynamic, content-dependent feature aggregation. As depicted in Figure 2.4, it consists of an encoder and a decoder with several transformer blocks of the same architecture. The encoder generates embeddings from the inputs, while the decoder uses all the embeddings and their incorporated contextual information to generate the output sequence. Each transformer block comprises a multi-head self-attention layer, a feed-forward neural network, a residual connection, and layer normalization. In the following, we describe how the shift from local/static to global/dynamic operations redefines architectural design while introducing new challenges in computational complexity and data efficiency.

Foundations of ViTs

In contrast to CNNs, a ViT uses a process called tokenization to partition an input image into a sequence of tokens. In addition, a basic ViT module comprises three components, with optional additional functions: multiple positional embeddings, an attention mechanism, and a normalization operation, along with residual connections and a Feed-Forward Network/Multilayer Perceptron (MLP).

Tokenization

Unlike CNNs, which directly process grid-structured data (e.g., images or videos), ViTs require tokenization to adapt self-attention mechanisms from NLP. Specifically, because treating individual pixels as tokens is computationally infeasible, ViTs partition the input image into patches. Each patch is then linearly projected into a vector embedding, and the sequence of these embeddings serves as the input to the transformer. Each

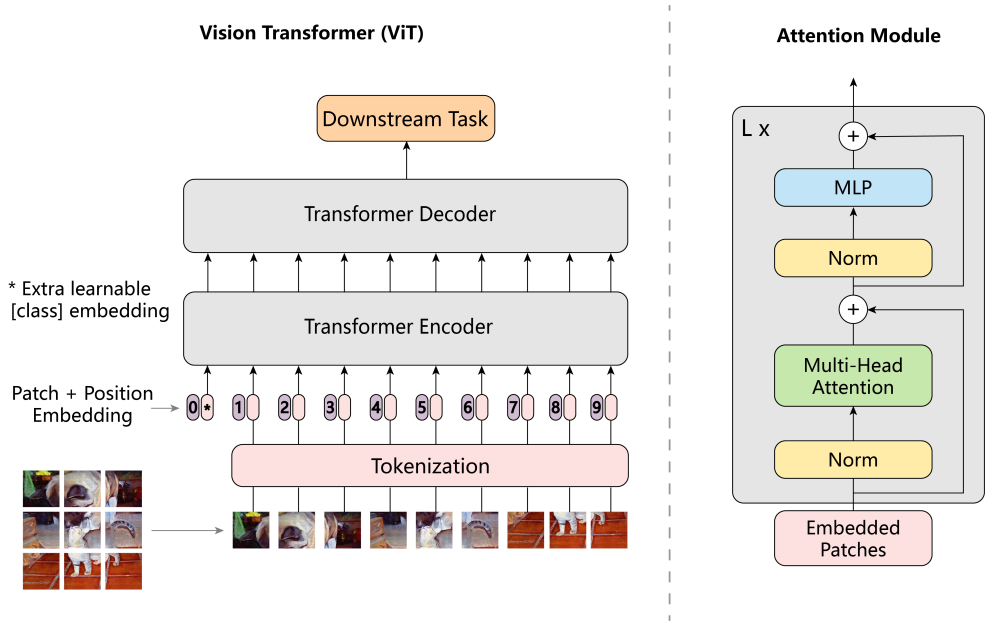


Figure 2.4: Schematic representation of a generalized framework for Visual Transformer (ViT) with **L** blocks.

patch embedding serves as a "super-node" of the input and contains information about all pixels in a group. Typically, each patch embedding is referred to as a visual token, and the conversion of grid-structured data into sequential tokens is called tokenization. The vision transformer [156], which is the origin of pure ViTs, partitions an input into fixed-sized non-overlapping patches, flattens each patch into a 1-dimensional vector, and projects all vectors onto a higher-dimensional embedding feature space using linear layers:

$$X_{patch}^{N\times D} = \text{tokenize}(I^{L\times C}) \tag{2.7}$$

where I is a flattened input with length L and feature channels C; X_{patch} is a sequence of tokens with length N and feature channels D. Note that $N = \frac{L}{P}$ in which the L is divisible by the area of a patch P; otherwise, the input image will be padded with zero elements.

Positional embedding

Because the self-attention mechanism is designed for sequential data, grid-structured data (e.g., images or videos) is flattened into 1D vectors, losing spatial structure. In

addition, the global pairwise correlation in self-attention is invariant to the position of each element. That means spatial information cannot be integrated into computation. Therefore, extra positional embeddings $X_{position}$ are added to the sequential input to retain spatial information:

$$X = X_{patch} + X_{position} \tag{2.8}$$

Different position embeddings fall into three categories:

Absolute position embedding (APE) is a vector that represents the unique absolute embeddings to the position of one element in a sentence [53]. It can use element embeddings to encode positional information.

Relative position embedding (RPE) technique is primarily used to incorporate relative position information into the attention mechanism [696, 115]. The RPE technique is based on the idea that the spatial relationships between patches carry more weight than their absolute parameters. The relative distance between patches determines the lookup process. Although the RPE technique can be applied to sequences of varying lengths, it may increase training and testing time.

Convolutional position embedding (CPE) methods take into account the grid structure of the input and employ convolutional operations to gather position information with zero-padding [693, 674, 289].

Self-attention mechanism

The self-attention mechanism is the primary operation in ViTs and plays a crucial role in explicitly modeling relationships among entities in sequential data. It calculates pairwise correlations between each location and all others, aggregates all long-term dependencies into a single entity, and represents it as global contextual information [642]. The self-attention mechanism transforms the input sequence(s) into three different embedding vectors, namely query, key, and value, as named in NLP configurations. We introduce a more generalized definition of the self-attention mechanism, which can benefit in explaining the developments of other variants and drawing connections with convolutions:

$$Y_i = \frac{1}{C(X)} \sum_{\forall j} f\Big(g_q(X_i), g_k(X_j)\Big) g_v(X_j) \tag{2.9}$$

Where i is the index of the visual token in the output sequence, j is the index that enumerates all possible positions. $X \in \mathbb{R}^{L \times C}$ is the input sequence and $Y \in \mathbb{R}^{L \times C}$

is the output sequence. Note that L is the length of the sequences, and C is the number of channels of each vector. A pairwise function f computes a correlation scalar between i and all j. The unary functions g_q, g_k, g_v transform the input sequence X into embedding vectors of query, key, and value, respectively. The pairwise function $f(\cdot)$ is normalized by a factor $\mathcal{C}(X)$.

The standard single-head self-attention mechanism can be instantiated as follows:

$$\begin{aligned} &g_q(X) = \text{Linear}(X),\ g_k(X) = \text{Linear}(X),\ g_v(X) = \text{Linear}(X) \\ &\frac{1}{\mathcal{C}(X)} f\Big(g_q(X), g_k(X)\Big) = \text{Softmax}\left(\frac{g_q(X) \cdot g_k(X)^T}{\sqrt{d_k}}\right) \end{aligned} \tag{2.10}$$

where $\text{Linear}(\cdot)$ denotes linear projection function, $\text{Softmax}(\cdot)$ denotes the softmax normalization function, and $\frac{1}{\sqrt{d_k}}$ is the scaling factor with d_k as feature channels of the key vector. Then, the standard single-head self-attention mechanism can be reformulated into a popular formula:

$$\begin{aligned} &q_i = g_q(X_i),\ k_j = g_k(X_j),\ v_j = g_v(X_j), \\ &\text{Attention}(q_i, K^T, V) = \sum_{\forall j} \text{Softmax}\left(\frac{q_i \cdot k_j^T}{\sqrt{d_k}}\right) \cdot v_j \end{aligned} \tag{2.11}$$

where q_i, k_j, and v_j are the embedding vectors of query, key, and value at location i and j, respectively. In addition, K^T and V are the transposed key and value embedding matrices, respectively.

It is limited by the capacity of single-head self-attention to attend to all important positions. Multi-head self-attention (MHSA) [642] is utilized as the real component in ViTs. Like group convolution, MHSA assumes several mutually independent feature groups and decomposes the embedding spaces of the query, key, and value into multiple sub-spaces. Applying single-head self-attention across sub-spaces in parallel can not only reduce computational complexity within groups but also capture a diverse range of complex interactions across groups. MHSA comprises multiple self-attention blocks, each of which projects the concatenation of its outputs into the output space. The separation enables MHSA to focus on multiple portions and to capture the relationships

among them effectively:

$$\begin{aligned} \text{Head}_i^k &= \text{Attention}(q_i^k, K^k, V^k) \\ \text{MHSA}(q_i, K, V) &= \text{Concat}(\text{Head}_i^1, \text{Head}_i^2, \cdots, \text{Head}_i^k) \cdot W^{Cat} \end{aligned} \tag{2.12}$$

where Head_i^k is the k-th output of group-wise self-attention at position i, and W^{Cat} represents the learnable weights to transform the concatenation of all heads into the final output.

Other components

Feed-Forward Network (FFN) is applied after the self-attention modules to obtain more complex attributes from the input sequences. It consists of two linear project functions and a nonlinear activation function

$$\text{FFN}(X) = W_2 \cdot \sigma(W_1 X) \tag{2.13}$$

where W_1 and W_2 are the weight matrices of the two linear project layers, and $\sigma(\cdot)$ represents the non-linear activation functions, such as GELU [251]. The dimensionality of the hidden layer is $d_h = 2048$ by default.

Normalization with residual connection consists of a residual connection and a normalization operation after each self-attention module. This design can stabilize information flow and improve performance. The output of these operations can be described as

$$\text{Norm}\left(X + \text{Attention}(X)\right) \tag{2.14}$$

where X is the input of the self-attention module, and Norm is a normalization layer, like layer-normalization [19]. A variant pre-normalization [23, 664, 156] is also widely used. It reorders the computation and inserts normalization into the residual connection before self-attention.

Characteristic of Transformers

Besides CNNs, ViTs differ in their architectural design due to their self-attention mechanism. Because self-attention is designed for sequential data, we assume the input data is a sequence of length L.

1. **Global receptive field**: Each visual token in the input sequence is connected to every other token, regardless of distance. Globality can explicitly capture long-range dependencies and accumulate pairwise relationships to build global contextual features. However, the adopted global receptive fields, rather than sampling, result in a computational complexity of $\mathcal{O}(L^2)$ that quadratically increases with the length of the input sequence.

2. **Flatten structure**: Since the self-attention mechanism directly adopts a global receptive field, there is no need to use a hierarchy of layers to expand receptive scopes gradually. Instead, the ViT stacks multiple self-attention modules with the same structure to introduce nonlinearity and improve the model's capacity to capture complex relationships in the data. Notably, the global receptive fields reduce the inductive bias of locality in ViTs that require feature representations with small spatial variances.

3. **Dynamic Parameter**: The aggregation weights in the self-attention mechanism are dynamic because they are computed as pairwise correlations and vary depending on the input. The dynamic parameters can be interpreted as gating information that adaptively highlights significant content and captures complex relationships from the input. Unlike CNNs, which enforce translation equivariance with static kernels, dynamic weights in self-attention allow the model to prioritize input-dependent contexts rather than rigid spatial invariance.

In summary, the self-attention mechanism is closely related to dynamic gating convolution 2.2.1. Within a global receptive field, gating information is computed as pairwise correlations based on the input content and can highlight all important locations for each query element. Compared to CNNs, ViTs impose fewer inductive biases and, in theory, can process a broader range of data.

Improvements of the Attention Mechanisms

Given that images involve more dimensions (e.g., spatial dimension), noise, and redundant modality, they are believed to be more difficult to capture robust long-range dependencies than sequential data. Following NLP conventions, ViTs use tokenization to group pixels into a small number of visual tokens, each representing a semantic concept in an image. By capturing long-range dependencies among tokens, self-attention mechanisms offer significant flexibility in feature representation learning, as their dynamic weights adapt to each input image. However, the adaptability comes with four limitations: (1) the quadratic computational complexity increases with the

sizes of images; (2) the data-hungry requires a large scale dataset (e.g., JFT-300M [599] dataset with 300 million images) for pre-train otherwise impedes fine-tuning of downstream tasks with smaller datasets; (3) fixed patch size cannot provide a plausible granularity to describe the true semantic parts in different images at various scales. (4) long-range dependencies at different scales cannot freely flow across feature maps at different scales. To address one or more limitations, a series of variants of the Vision Transformer [156] have been proposed to improve attention mechanisms and modify the architectures.

Reduce computational complexity

The quadratic complexity of self-attention hinders its applicability to long sequences, i.e., high-resolution images. The core bottleneck stems from the heavy matrix multiplication used to compute pairwise dependencies. Two main categories simplify the self-attention computation process.

Similar to the core idea of separable convolution, the *matrix decomposition* factorizes the large matrix of self-attention into a product of several sub-matrices with lower ranks. CaiT [627] proposed class attention (CA) that computes attention between class tokens and patch tokens. CA reduces the complexity and is used with self-attention to build Transformers with very deep architectures. Geng et al. [193] proposed the "Hamburger" module, which reformulates matrix decomposition as a low-rank reconstruction process. Based on the low-rank matrix, it can efficiently compute dependencies among different tokens. Linformer [670] proposed a different decomposition scheme. Based on the distributional Johnson–Lindenstrauss lemma [418], it first projects the key and value matrices to low-rank matrices and then computes attention between the query and the low-rank matrices. Performer [113] proposed a novel Fast Attention Via positive Orthogonal Random features (FAVOR+) approach that can efficiently model kernelizable attention mechanisms beyond softmax. Katharopoulos et al. [318] proposed linearized attention, which rearranges the original self-attention as a linear dot product between a query vector and two intermediate matrices. The advantage is that both intermediate matrices are computed once and reused for all query vectors. Nyströmformer [722] adapted the Nyström method to approximate standard self-attention. XCiT [6] proposed cross-covariance attention that computes attention across feature channels instead of tokens. Cross-covariance attention has linear complexity with respect to the feature dimension rather than the number of tokens. Singularformer [700] used neural networks to learn the singular

value decomposition process and decompose the attention matrix into the product of three matrices with lower ranks. SeT [604] factorized self-attention into pixel-wise local and patch-wise global attention, reducing the computational cost while retaining the information at different granularities. CSWin [154] decomposed self-attention into horizontal and vertical strips in parallel that form a cross-shaped window, reducing the computational cost.

Another direction seeks to find sparsity patterns in the matrix and approximate self-attention with *sparse attention*. Given an attention matrix $A \in \mathbb{R}^{L\times L}$ of sequence length L, sparse attention only computes pairwise interactions on a subset $\mathcal{S} = \{(i,j)|A(i,j) \neq 0\}$, reducing the quadratic complexity $O(L^2)$ to near-linear. Sparsity enables processing of high-resolution images with large L, making sparse attention preferable for tasks such as segmentation or generation on large inputs. Therefore, there has been great enthusiasm to explore different sparsity patterns $\mathcal{S}$. Child et al. [110] introduced sparse factorizations of the attention matrix and proposed fast attention kernels for training. Yun [788] argued that a sparse attention model can universally approximate any sequence-to-sequence function. The results showed that sparse Transformers with $\mathcal{O}(L)$ connections per attention layer can approximate the same function class as the dense model with L^2 connections. The Sinkhorn Transformer [618] is based on differentiable sorting of internal representations and computes quasi-global attention using only local windows, thereby improving the memory efficiency of the self-attention module. Reformer [332] observed that the softmax function in self-attention will be dominated by a small subset of keys closest to the query in computation and defined these sparse sets by finding nearest neighbors in high-dimensional spaces. Based on locality-sensitive hashing (LSH), Reformer proposed a simplified sparse attention mechanism, called LSH attention, to reduce computational complexity. Likewise, KVT [663] adopted the K-NN clustering method to find top-k sparse similar tokens. Inspired by graph theory, many works [116, 590, 791] have formulated attention mechanisms as graphs and have adopted various graph sparsification methods to reduce computational complexity. Chung et al. [116] adopted random graph theory and approximated the dense graph with sparse graphs. Spielman et al. [590] based on spectral similarity of graph Laplacians and proved that every graph has a spectral sparsifier of nearly linear size. A construction algorithm was proposed to build a dense graph from sparse graphs. BIGBIRD [791] used a directed graph and proposed sparse attention, where each query attends over a small set of randomly chosen keys. Inspired by deformable convolution, DAT [707] emphasized the importance of key-value pairs in a data-dependent manner and focused on relevant patches, thereby capturing informative features from sparse

sets.

Alleviate data-hungry

ViTs yield modest results when trained on mid-sized datasets such as ImageNet [138], with 1.2 million images, achieving lower accuracy than CNNs with comparable parameter counts. Because transformers lack some inductive biases inherent to CNNs–such as translation equivariance and locality–they do not generalize well when trained on insufficient data. However, Han et al. [225] found that training ViTs on large datasets (14 million to 300 million images) overcame the limitations of inductive bias, resulting in excellent fine-tuning performance on downstream tasks even with fewer data samples. Many variants are being explored to improve data efficiency in ViTs trained on smaller datasets. Many ViT architectures use knowledge distillation to transfer knowledge from a larger network (i.e., the teacher model) to a smaller network (i.e., the student model) [219]. The DeiT [626] is the first work to demonstrate that transformers can be trained on mid-sized datasets (e.g., 1.2 million ImageNet examples) in a relatively short training time. Using a pre-trained Vision Transformer as the teacher model, DeiT learned to capture the most important pattern in tokens and produced similar results with fewer parameters. DINO [55] introduced self-supervised learning and proposed a form of self-distillation with no labels. The key difference is that the teacher model was built from past iterations of the student network and was updated using an exponential moving average of the student weights, i.e., serving as a momentum encoder. To avoid model collapse, the student model was trained to predict the teacher model's output after a centering and sharpening layer. TaT [413] argued that the one-to-one spatial matching strategy in a patch-based manner cannot always be maintained in distillation learning because different architectures may cause semantic inconsistency between patches from the teacher and the student model, respectively. Instead, TaT proposed a one-to-all matching knowledge distillation pipeline that allows each teacher patch to teach the entire set of student features dynamically. To avoid intractable computational complexity arising from correlations in large feature maps, TaT proposed a two-step hierarchical distillation pipeline, i.e., patch-group and anchor-point distillation, to reduce computational burden and achieve decent results. TinyViT [697] proposed a fast distillation method in the pre-training stage and built a family of tiny yet efficient ViTs suitable for devices with limited resources. In addition, TransKD [432] is the first framework for distilling knowledge from large-scale semantic segmentation transformers. By focusing on comprehensive distillation across different knowledge sources within

the transformer-specific building, TransKD distills information from both feature maps and patch embeddings.

Hierarchy of dynamic patches

In ViTs, an image is first divided into a grid of patches, projected to a linear embedding, and treated as a sequence of tokens. Each token corresponds to a semantic region in an image. Since images are highly complex, with abundant detail and color information, the granularity of patch division is insufficient to extract semantic regions at different scales and locations. In addition, a better image partition can reduce computational redundancy from irrelevant patches in self-attention. Consequently, many researchers have explored different architectures for flexible patch generation. Two main strategies to dynamically adjust patch sizes are (1) Patch-fused Self-Attention and (2) Pooling-based Self-Attention.

Patch-fused Self-Attention fuses fixed-sized patches into windows with dynamic sizes to better cover semantic parts. Moreover, the merging strategy progressively reduces the patch sequence length and constructs multi-scale representations. T2T-ViT [780] proposed a Tokens-to-Token (T2T) transformation that applied a sliding window algorithm to combine neighboring tokens into a single token with overlapping recursively. T2T-ViT adopted an efficient backbone and reduced the parameter count by half due to shorter token sequences, enabling training from scratch on the mid-sized ImageNet dataset. Similar to Network In Network (NIN) [507], TNT-ViT [226] proposed a Transformer in Transformer (TNT) architecture that adopts a hierarchical patching scheme. In each TNT module, smaller patches (visual words) were fused into larger patches (visual sentences) without overlapping. Within the TNT architecture, local correlations among visual words are captured within each visual sentence, while global correlations are extracted simultaneously across visual sentences. DPT [103] proposed a Deformable Patch (DePatch) module, which learns to adaptively split the images into patches in a deformable way with learnable positions and scales. The method can generate dynamic patches that capture image semantics based on their content. Swin-ViT [441] proposed a shifted-window architecture. This framework generated windows from patches at the same layer without overlapping and computed local self-attention within each window. To gradually capture long-range dependencies across layers, Swin-ViT shifted window partitioning within successive blocks and built connections between windows across layers. Several works [286, 174, 544, 817, 587, 440] improved the shifted window strategy. Shuffle Transformer [286, 174] utilizes the spatial shuffle operation instead of

shifted window partitioning to allow cross-window connections. DW-ViT [544] explored the upper limit of the effect of window settings on model performance with dynamic multi-scale windows; VSA [817] employed window regression modules to predict the size and location of the target windows from data; CSWinTT [587] utilized a cyclic shifting window strategy at multi-scales.

Pooling-based Self-Attention, inspired by hierarchical structures used in CNNs, gradually reduces the sizes of feature maps and expands the effective coverage of patches. PVT [673] introduced a progressive shrinking strategy and generated a pyramid of patches without overlapping. The pyramid structure can represent semantic parts in images at various scales. PVT proposed spatial-reduction self-attention (SRA) to capture long-term dependencies at low-resolution key-value pairs. RegionViT [65] combined a pyramid structure with an efficient regional-to-local (R2L) self-attention mechanism. Like TNT-ViT, RegionViT adopted regional self-attention (RSA) to learn global information, while local self-attention (LSA) was used to learn local features within regions. Like PVT, PiT [252] also used a pooling layer to reduce token length. Based on pyramid structures, Twins [114] and CAT [408] alternately performed local and global attention across the pyramid layers. MS-VIL[816] proposed a mechanism that combines local attention with global memory. The local attention only captured short-term dependencies among neighboring tokens, whereas global memory propagated information over long distances, as in Longformer [34] in NLP tasks. Focal Transformer [747] sampled all surrounding tokens for each query on every pyramid layer and computed local self-attention among concatenated neighboring tokens across multi-scales. CrossFormer [676] progressively enlarged group size along the pyramid for patch embedding and decomposed self-attention into long/short distance attentions, capturing local and global visual cues.

In summary, hierarchical tokenization exponentially reduces sequence length, thereby lowering the cost of subsequent attention layers. The reduction is because complexity quadratically scales with token count. Consequently, Hierarchical tokenization is especially beneficial for high-resolution inputs and tasks requiring multi-scale reasoning, such as object detection and semantic segmentation, providing a better accuracy-efficiency trade-off than fixed-resolution transformers.

Attention across scales

Although ViTs can propagate global context information sequentially through transformer decoder layers, this information cannot freely flow across different scales. Many

approaches applied multiple ViTs in their architectures to extract long-term dependencies from multi-scale features and to directly interact with global information across any pair of scales. CrossVit [66] proposed a ViT architecture having dual branches and defined a novel type of attention mechanism (called cross-attention) that enables information transfer at different scales. The cross-attention assigned the class tokens to query and used the concatenation of class and patch tokens as the key and value. Therefore, the simplified attention mechanism computed only long-term dependencies between class tokens and image patch tokens, and removed the most expensive matrix computation involving the image patch tokens, thereby achieving linear computational complexity. Similarly, MMViT [439] adapted dual branches and introduced Cross-Pooling-Attention to fuse information at different views. Moreover, Dual-ViT [763] extended cross-attention with two pathways: a semantic pathway that compressed token embeddings into global semantic queries and a pixel pathway that mined finer local details. The global semantic queries served as useful prior information, guiding the refinement of the feature maps at the pixel level.

Efficient Alternatives Beyond Attention Mechanisms

Although many improvements aim to mitigate the quadratic scaling of attention mechanisms through approximations, Visual Transformers still struggle to achieve a trade-off between efficiency and performance as sequence length increases. This fundamental bottleneck has spurred intense research into deeper-level architectural alternatives. Among these, structured state space models (SSMs), rooted in classical control theory, have recently emerged as a theoretically grounded option that entirely avoids attention, favoring recurrent state transitions with linear-time complexity. Mamba [207], an outstanding representative, proposes a selective, data-dependent SSM that challenges the necessity of global attention for high-performance sequence modeling and offers a compelling, linearly scalable alternative. Although challenges remain, such as optimizing training stability and further validating scaling at the very largest scales, Mamba has firmly established itself as a cornerstone in the landscape of post-attention efficient foundation models. Its emergence underscores that the future of sequence modeling may not be a choice between attention or alternatives, but an intelligent synthesis of complementary computational principles.

Selective SSMs as Linear-Time Alternatives

At its heart, Mamba departs from the Transformer's fixed, input-agnostic context

mixing. It is built upon a structured, parameterized SSM that maps a 1-D input sequence $x(t)$ to a latent state $h(t)$ and output $y(t)$ through continuous-time differential equations, which are then discretized for digital computation. In contrast to attention's dynamic weighting, the conceptual counterpoint of Mamba is selectivity, which converts the matrices governing dynamics and input projection into functions of the current input x_t. Specifically, attention computes pairwise relevance scores across all tokens. At the same time, the selective SSM modulates its internal state transition dynamics and input sensitivity at each step based on the incoming token. Selectivity allows the model to propagate or discard information contextually throughout the sequence, emitting a "hidden state" that carries a compressed, relevant history. Crucially, thanks to a hardware-aware parallel algorithm inspired by parallel prefix scans, this inherently sequential process can be trained efficiently. During inference, Mamba operates with linear-time complexity $O(L)$ and a constant memory footprint relative to sequence length L, as it only needs to maintain a fixed-size hidden state, eliminating the massive KV cache of autoregressive Transformers.

Theoretical Unification and Hybrid Models

Based on a profound theoretical connection, there is an established link between SSMs and Transformers through the structured state space duality (SSD) framework [133]. It shows that both attention and SSM-based layers can be derived as different instantiations of computations on semi-separable matrices. This mathematical duality reveals that Mamba and attention are not opposed but lie on a unified continuum of sequence-mixing primitives, offering a rigorous lens for understanding and potentially unifying these seemingly disparate approaches. Therefore, a prominent research direction is the exploration of hybrid models that integrate Mamba blocks with attention blocks. Architectures such as Jamba [404] demonstrate that such hybrids can outperform pure Transformers or pure Mamba models with equivalent total compute, leveraging Mamba for efficient long-context processing and attention for precise, localized reasoning. Its success has catalyzed a broader reassessment of the fundamentals of sequence modeling, leading to practical hybrid systems and deeper theoretical unification.

Scaling Towards Universal Large Foundation Models

Apart from the previous research trajectory of reducing ViT parameter scales for computational efficiency, a significant yet opposite paradigm has emerged: embracing scale and complexity to pursue universal functions and extreme generalization. The

extreme generalization has led to the rise of large-scale foundation models, which are pre-trained on web-scale data and exhibit remarkable zero-shot or few-shot adaptability across a wide range of downstream tasks, effectively functioning as general-purpose cognitive systems. Their development represents a pivotal shift from task-specific models to universal intelligence across multiple modalities.

Large-Scale Visual Foundation Models

As a self-supervised learning framework built on ViTs, the DINO family has been a significant contributor to the quest for universal visual representations. Analogous to DINO [55], which uses large-scale, curated data from diverse sources, DINOv2 [502] produces generalized visual features that work across image distributions and tasks without fine-tuning, making them exceptionally effective drop-in replacements for traditional backbones in diverse visual applications. Furthermore, DINOv3 [576] introduced the pivotal Gram anchoring and effectively addressed the known yet unsolved issue of dense feature maps degrading during long training schedules. Therefore, DINOv3 could produce high-quality, dense features for various vision tasks, surpassing previous self- and weakly supervised foundation models. In summary, the DINO family demonstrates that with carefully designed self-supervised objectives and large-scale, curated data, ViTs can learn a comprehensive model of visual appearance and structure without labor-intensive manual annotations.

In parallel with advances in visual representation, the Segment Anything Model (SAM) [330], and its successor, SAM2 [537], stand as landmark achievements in universal segmentation frameworks. SAM introduced a promptable segmentation model trained on an unprecedented dataset of over 1 billion masks. Its core innovation lies in decoupling the heavy image encoder (a ViT-H) from a fast prompt encoder/mask decoder, demonstrating zero-shot transfer to new image distributions and tasks. SAM2 further advances promptable visual segmentation for images and videos. The model is equipped with a streaming memory that stores information about the object and its previous interactions, allowing it to generate masked predictions throughout the video and effectively correct them based on the object's memory context from previously observed frames. The SAM family exemplifies how scaling data, model capacity, and task diversity (through prompting) can yield a model with unparalleled generalization in spatial understanding. For more information about large-scale visual foundation models, please refer to the survey [17].

Multiple-Modal Models

While visual foundation models have dramatically advanced computer vision tasks, their capabilities are inherently bounded by the visual modality. They often require extensive task-specific labeling and struggle with open-world semantic understanding and generalization beyond predefined categories. This limitation has catalyzed a pivotal shift towards multi-modal foundation models that integrate and align information from diverse data sources. By learning joint representations across modalities, these models harness the complementary strengths and rich supervisory signals present in uncurated, web-scale multi-modal data.

Due to the limitations of visual foundation models that require a closed set of predefined categories, Visual-Language Foundation Models (VLMs) have transcended this boundary by anchoring visual representation learning in the open and semantically rich modality of natural language. The foundation of this paradigm lies in learning a joint embedding space in which VLMs can pull together semantically matched images and texts while pushing apart mismatched pairs. Therefore, the mainstream research on VLM focuses on learning more robust and versatile image-text correlations. The pioneering work CLIP [530] established the efficacy of large-scale contrastive pre-training for zero-shot transfer. To mitigate data noise and scale limitations, ALIGN [303] demonstrated training on billions of noisy pairs, while DeCLIP [386] and UniCL [748] improved data efficiency through self-supervision and label integration. For a finer-grained understanding, FILIP [761] and GLIP [376] introduced region-word alignment, while FLAVA [578] integrated masks for flexible guidance. Architectural innovation has evolved from dual-encoder towers (CLIP) to unified single-tower designs [630, 297] to enable efficient cross-modal fusion. Furthermore, models like DetCLIP [760] and FIBER [157] extended VLMs from image-level to region-level tasks, paving the way for open-vocabulary object detection and segmentation. This trajectory illustrates a clear path from learning global image-text associations to mastering fine-grained, task-agnostic visual-semantic correspondence. For more information about large-scale visual-language foundation models, please refer to the survey [810].

Building on the foundational architecture and training paradigms established by VLMs, the multi-modal learning paradigm has been successfully extended to a wide range of domains, demonstrating remarkable versatility. In audio-visual learning, models align spectrograms with textual descriptions to perform tasks such as sound event localization and captioning. The video-language domain leverages temporal alignment between video frames and transcripts or instructions to enable video question

answering and action recognition. In affective computing, multi-modal sentiment analysis models fuse visual (facial expressions), acoustic (prosody), and linguistic (text) cues to infer emotional states. The medical imaging field integrates radiology images with clinical reports to build foundation models for automated diagnosis and report generation. Furthermore, robotics utilizes models that combine visual perception, language instructions, and sensorimotor data for task planning and human-robot interaction. These expansions consistently rely on core VLM principles—cross-modal alignment, contrastive or generative pre-training on paired data, and zero-shot generalization—thereby demonstrating the transferability of the paradigm and establishing a unified framework for multimodal intelligence.

ViT Summary

While ViTs excel at capturing global context through dynamic attention, their lack of inherent spatial inductive biases, such as locality and translation equivariance, and their high computational complexity present significant barriers for efficient and robust visual modeling. The bottleneck provides compelling motivation for hybrid architectures that integrate the complementary strengths of convolutional locality and attentional globality into a unified, synergistic framework.

2.2.3 Hybrid Networks

Although ViTs can efficiently capture long-term dependencies among visual tokens, they cannot flexibly represent local semantic parts at multiple scales. Except for the aforementioned improvements 2.2.2, using a simple linear layer remains less straightforward than convolution for extracting local features with spatial geometric information within each visual token. Besides, because ViTs lack some inductive biases inherent to CNNs, such as translation equivariance and locality, they require large datasets (e.g., JFT-300M) to overcome these biases and generalize. Empirically, a self-attention mechanism in early layers works similarly to a standard convolution, showing locality and sparsity in pattern extraction [122, 312]. These limitations have motivated applying convolution operations to ViTs, especially in the early layers, to reduce computational redundancy. On the contrary, CNNs can be trained on mid-sized datasets (e.g., ImageNet). Still, they cannot efficiently capture long-range interactions because of their local receptive fields. To combine the strengths of CNNs and ViTs, considerable effort has been devoted to hybridizing them. Hybrid architectures are a deliberate synthesis that combines the computational efficiency and spatial priors of CNNs with

the expressive power and global context modeling of ViTs. The advantage is achieved not by mere concatenation, but through deep architectural integrations that embed convolutional operations within attention mechanisms (Con-Trans) or augment convolutional feature maps with attention-based feature recalibration (Attn-CNNs). The subsection explores how hybrid models navigate trade-offs between locality/globality and between static/dynamic aggregation to achieve state-of-the-art performance across diverse vision tasks.

Attention enhanced CNNs

Inspired by non-local means operations [45], global or local self-attention mechanisms are reformulated as special instances of dynamic convolution. Given a feature map, non-local operations compute the response at a position as a weighted sum of the features at non-local positions in the feature map, regardless of the distance between them. Therefore, equipped with the capability to model long-range interactions in a feed-forward fashion, CNN architectures have demonstrated superiority for feature extraction by expanding the receptive field and focusing on different spatial locations based on their importance, as depicted in Figure 2.5.

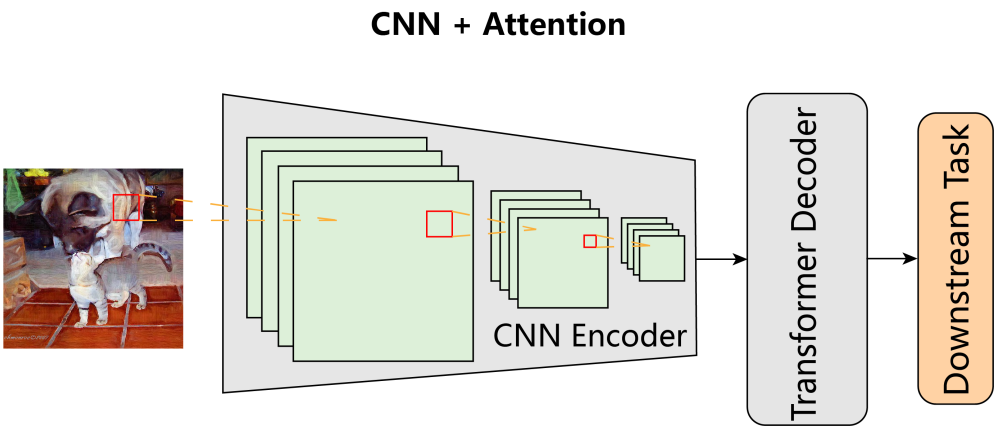


Figure 2.5: Schematic of a generalized framework for the hybrid framework of CNN with an attention mechanism.

Adapting Attention for CNNs

Attention mechanisms, as instances of dynamic convolution, can adjust their aggregation weights in response to changes in the visual input based on pixel similarities. Many Attn-CNNs applied attention modules to higher-dimensional visual data than to sequential

data for adaptation. RANet [652] stacked attention modules with residual units to generate attention-aware features adaptively varying as layers go deeper. SENet [268] focused on the channel relationship and proposed a "Squeeze-and-Excitation" (SE) block that adaptively captured interactions between channels. Non-local Network [678] introduced a non-local operation as a generalization of spatial attention, computing a weighted sum of features across all positions. CBAM [691] combined spatial and channel attention modules sequentially, while BAM [508] and DANet [183] combined them in parallel. A^2-Nets [98] proposed a double attention block that aggregated and propagated informative global context from the spatial-temporal representative space of input images and videos. AA-Net [33] introduced two-dimensional relative position encoding [570] into CNNs to maintain translation equivariance for handling images and applied MHSA in parallel to a standard convolution operation as an augmented operation. GCNet [52] proposed a three-step general framework that unified the Non-Local network and SENet to model global context. Ramachandran et al. [533] proposed a local self-attention layer that aggregates only within a specific window centered on each pixel and can be used in place of convolutional operations in CNNs. Zhao et al. [836] proposed two types of vector attention, pairwise and patchwise self-attention, to replace conventional attention with scalar weights. Both attention mechanisms learned vector weights for the spatial and channel dimensions simultaneously. Pairwise attention was a set operator that computed interactions within a local scope of aggregation. In contrast, the patchwise self-attention was an instance of the dynamic convolution operator that adaptively generated the local aggregation weights. FEDER [234] proposed a learnable wavelet transformation and applied spatial/channel attention and convolution to mine different frequency bands. Likewise, FADNet [461] captured long-term dependencies within the frequency domain and converted them into query vectors that contained global input-dependent contexts.

Reducing Computational Complexity

Although self-attention enables modeling of global contextual information, it is both memory- and computation-intensive. The dense attention map computation has a high complexity of $O(N^2)$, where N denotes the number of pixels in feature maps. To reduce this computational burden, many Attn-CNNs have explored matrix factorization or simplification to improve efficiency. One stream of research employed sparsity or locality properties to simplify self-attention mechanisms. CcNet [287] proposed the criss-cross attention module for each pixel position that approximated the dense matrix

multiplication with two simple vector multiplications along one criss-cross path. By recurrent criss-cross attention, each pixel position can capture long-range interactions from other pixels with a complexity of $O(2\sqrt{N})$. LRNet [267] proposed local relation networks to compute attention in a local window with lower complexity adaptively, and introduced geometric priors into CNNs. Other work has explored approximating the self-attention mechanism using simpler matrix multiplication. Similar to the core idea behind cross-attention (see 2.2.2), most efforts have focused on adapting cross-attention to achieve linear complexity for CNNs, extracting high-level semantic information between class tokens and pixels in feature maps. EncNet [806] introduced channel attention to highlight significant feature channels via a context-encoding embedding. The context vector was the output of cross-attention, which used the Euclidean distance rather than the cosine distance to compute similarity. EMANet [379] interpreted the cross-attention as a Mixture of Gaussian models solved by the Expectation-maximization algorithm. It updated static class tokens and re-estimated feature maps to remove unnecessary noise. On the contrary, OCRNet [784] introduced object-region representations as dynamic class tokens and obtained pixel-region relations via cross-attention to reconstruct object-level contextual representations. ECA-Net [665] is dedicated to reducing the complexity of the channel attention mechanism and proposes an efficient channel attention module with fewer parameters. Moreover, ResNeSt [807] introduced the Split-Attention block to implement channel attention across different groups of feature channels within multi-path feature representations, resulting in a simple, unified computational block with fewer parameters.

Convolution enhanced ViTs

Convolutional operations help mitigate several inherent drawbacks of visual transformers, including the lack of inductive biases (e.g., reliance on large-scale pretraining data), high computational complexity, and difficulty in capturing local low-level context and positional embeddings. Incorporating convolutions into visual transformers enables these hybrid models to leverage the strengths of both architectures, leading to improved robustness, efficiency, and performance across various computer vision tasks, as depicted in Figure 2.6.

Enhancing Tokenization

VTs [692] integrated convolutional operations into ViTs to enhance tokenization and proposed an efficient VT-FPN module to extract multi-scale feature maps with fewer

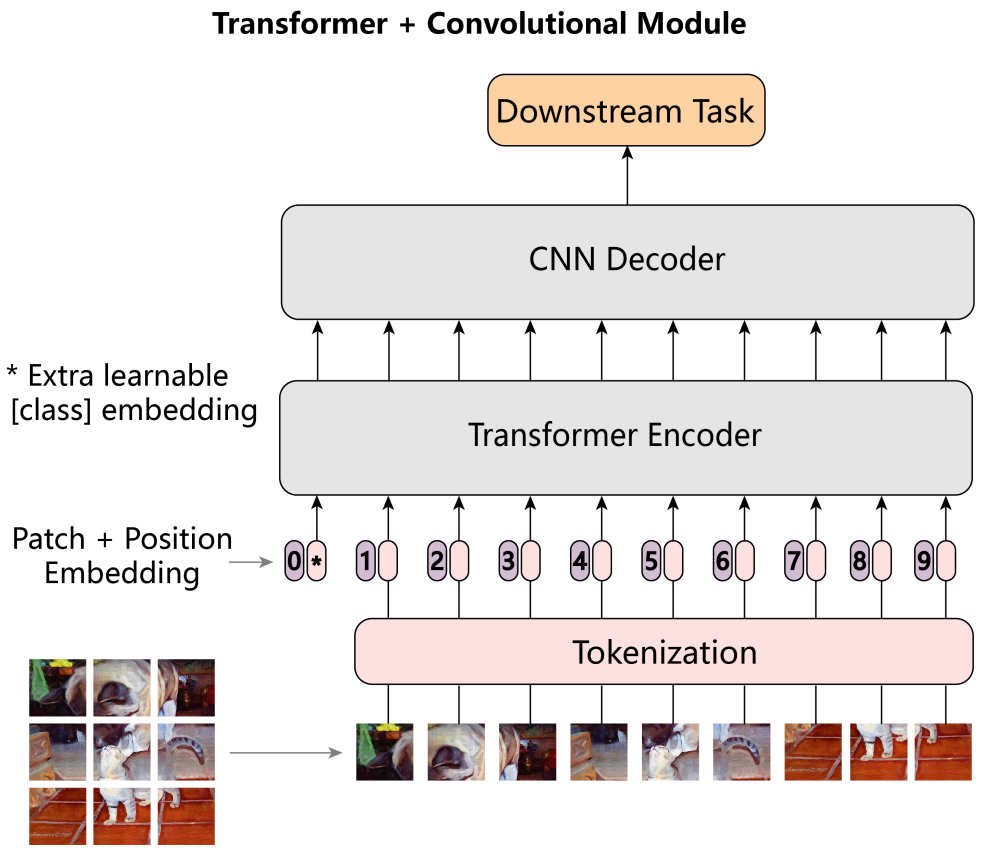


Figure 2.6: A generalized framework for the hybrid framework of CNN with an attention mechanism.

FLOPs than conventional FPN. Xiao et al. [711] substituted the original tokenization process with several stacked 3×3 kernels with a stride 2, facilitating the stability and generalization of ViT for downstream tasks. CvT [693] replaced Linear projections with convolutions to capture the spatial structure and low-level details for tokenization and adopted a hierarchical structure to progressively reduce the token length, imitating the impact of spatial down-sampling as in CNNs. CeiT [779] integrated the advantages of CNNs in extracting low-level features and enforcing locality into an image-to-token module that extracts patches from the generated local context. CCT [232] introduced a convolutional tokenization process and proposed a sequence pooling scheme to utilize multi-scale information. In addition, CCT can be trained from scratch on smaller datasets, e.g., CIFAR10, a remarkable property not possible with traditional ViTs. CPVT [115] utilized depth-wise convolutions to define Positional Encoding Generators,

incorporating positional information into patch embeddings. ResT [818] replaced the patch embedding as a stack of overlapping convolution operations with a stride on the token map and incorporated depthwise convolutions and adaptive position encoding to tackle varying image sizes. MPVit [359] integrated convolution operations with overlap to generate patch embeddings at different scales and proposed a multi-path architecture to interact global and local features across scales. CrowdFormer [750] proposed an overlap patching transformer block to capture the global contextual information in a top-down manner to extract overlapping patches at different scales.

Integrating locality into Attention

LocalViT [393] introduced depthwise convolution into Feed-Forward networks and brought locality into self-attention mechanisms. LeViT [203] adopted a pyramid architecture with pooling, similar to the LeNet [350], and explored the convolutional behavior in the transformer patch projection layer. BoTNet [592] adapted ResNet Bottleneck Modules that replaced the spatial convolutions with global self-attention. CMT [213] integrated convolution operations into the FFN of each transformer block to promote the correlation among neighboring tokens. ConViT [161] designed a dual path that attached a parallel convolution branch to a transformer branch for soft integration of convolutional inductive biases via a Gated Positional Self-Attention (GPSA). Refiner [851] observed the over-smoothing issue of MHSA in deep architectures and directly integrated convolution kernels into the self-attention mechanism, thereby augmenting local patterns in the attention matrices and establishing a coaction between global context aggregation (via self-attention) and local context modeling (via convolution). NesT [835] adopted a hierarchy of blocks to represent images at different scales and aggregated blocks across the hierarchy using convolution and pooling operations, thereby communicating information among blocks. Visformer [102] performed a comparative study between convolution operations and self-attention modules. It revealed that replacing self-attention modules with residual convolutional blocks [716] in the early stages can effectively capture both spatial and local features while requiring fewer FLOPs. CoAtNet [131] introduced relative attention that combined depthwise convolution with self-attention and vertically stacked convolution and attention in the shallow layers. Likewise, ACMix [506] addressed an underlying relation between convolution operations and self-attention mechanisms and combined them in parallel as a lightweight aggregation operation. MOBILEVIT [469] combined the strengths of CNNs and ViTs to build a lightweight and low-latency network for mobile vision

tasks. Conformer [515] adopted a concurrent structure consisting of one CNN and a transformer branch. It proposed a Feature Coupling Unit (FCU) to integrate local features into global representations.

Reducing Computation Complexity

ConvBERT [311] simplified self-attention calculation by proposing a new module - called span-based dynamic convolution - that combines the fully connected and convolutional layers. Long-Short Range Attention (LSRA) [703] reduced the computational cost of the self-attention mechanism by using convolutional operations to capture local information. Efficient-Attention [571] decomposed the pairwise matrix multiplication with two channel-attention functions and converted the matrix production into vector multiplication. Coat [737] devised a conv-attentional mechanism that approximated the softmax attention matrix as a product of two linear functions, inspired by LambdaNets [32]. Perceiver [292] combined cross-attention with latent transformers that mapped arbitrary high-dimensional input onto a fixed-sized latent space, reducing computational complexity. Maxvit [632] introduced a Multi-Axis attention mechanism that integrates local blocked attention with dilated global attention. This design enables global-local spatial interactions across arbitrary input resolutions while maintaining only linear computational complexity. PVT-v2 [674] proposed a linear spatial reduction attention module that reduced the spatial dimension of feature maps to a fixed size via average pooling, enjoying linear computational and memory costs like a convolutional layer. EfficientViT [47] proposed lightweight multi-scale attention to handle high-resolution input via ReLU-based global attention [318]. SwiftFormer [567] introduced a novel, efficient additive attention mechanism to replace the quadratic matrix multiplication operations with linear element-wise multiplication.

2.2.4 Development Trend Summary

The progression from CNNs to ViTs to Hybrids illustrates a clear conceptual trajectory: starting with strong, locally biased priors, moving towards flexible, globally aware computation, and culminating in architectures that strategically balance both. This evolution is governed by the continuous interactions between the core mathematical operations of convolution (local, static) and attention (global, dynamic). The resulting hybrid models are not merely a combination of two components, but a new class of architectures where locality and globality, static and dynamic, coexist and cooperate, defining the current frontier in visual representation learning.

2.3 Other Techniques

Some stand-alone techniques are used in deep learning to improve automation and expand the range of applications. Neural Architecture Search (NAS) is a technique that reduces the difficulty of designing efficient and general architectures for given complex datasets. Specifically, NAS formalized model architectures as a set of learnable hyperparameters and automated hyperparameter tuning via dedicated training processes. In addition, Neural Architecture Compression (NAC) explored patterns of feature vectors and mapped them to similar vectors using fewer bits. The compression aims to deploy powerful deep learning models in resource-limited scenarios, such as mobile and embedded devices, without significantly degrading performance. Moreover, Knowledge Distillation (KD) is used to transfer knowledge from source models to target models. The knowledge transfer enables the easy generation of new models as an ensemble of an enormous number of off-the-shelf models. The last aspect is to alleviate the hunger for annotated data. Despite the almost unlimited growth in data, the corresponding high-quality manual annotations lag far behind due to the high cost of labeling, especially in domains such as biology, medicine, and chemistry. Insufficient annotations have hindered the rapid development of supervised deep learning models. Therefore, self-supervised learning (SSL) or unsupervised learning (USL) was used to improve the robustness of feature representations without requiring additional supervision (i.e., expensive manual annotation). Moreover, Diffusion models provide a powerful generative framework for capturing data distributions and producing high-quality samples.

2.3.1 Neural Architecture Searching

Neural Architecture Search (NAS) is a deep learning technique that automates the optimal design of networks in architecture and engineering. Traditionally, designing neural network architectures has required specialized knowledge and extensive trial and error. Instead, NAS formulates the architecture search space as a set of possible combinations of layers, skip connections, operations, and hyperparameters. The mathematical core is to relax the search space so that continuous weights represent the choice between operations. Then the output is a mixed operation over the candidates. Therefore, a joint optimization of architecture choices and network parameters is performed via gradient descent. A final discrete architecture is derived by retaining the operation with the highest weight. Moreover, NAS proposes different searching strategies to explore the search space. It employs performance-estimation strategies to identify optimized archi-

tectures for visual tasks, such as image classification, segmentation, and object detection [163]. Early NAS works explored various search spaces defined by units, operations, and their connections for CNNs [873, 874, 422, 875, 531, 611, 826, 742, 148, 470, 739]. Later, NAS further explored the design of more efficient and scalable ViT architectures suitable for vision tasks [82, 64, 391, 596] by combining tokenization and attention blocks. Recent advancements in NAS have identified optimal combinations of CNNs and ViTs, leading to more powerful and efficient models [216, 579, 363, 777, 468, 771]. Nevertheless, NAS requires a delicate trade-off among search space expressiveness, search strategy cost, and estimation accuracy. While gradient-based methods are fast, they may favor narrow, continuous spaces. Evolutionary strategies are more flexible but computationally expensive, requiring thousands of model evaluations.

2.3.2 Neural Architecture Compression

In this subsection, we exclude some techniques (e.g., matrix decomposition and knowledge distillation) that, though applicable to model compression, are not dedicated solutions due to their broader applicability in other applications [498]. Instead, we mainly introduce two dedicated directions for model compression: pruning and quantization.

Pruning

Pruning techniques aim to sparsify neural networks by removing redundant components with minimal impact on performance [736, 249]. Currently, pruning is the most common method of reducing network complexity and overfitting [120]. The pruning methods began with unstructured pruning in the early stages for CNNs. Inspired by Dropout and DropConnect operations, early unstructured pruning methods removed neurons or connections whose magnitude or importance ranking fell below a predefined threshold. The variants [266, 775, 1, 464, 801, 605, 15] alternated between weight updates and pruning by dynamically adjusting thresholds. More recent structured pruning techniques have introduced different levels based on pruning granularity. From fine to coarse, they are kernel-level, block-level, and filter-level pruning [341]. The kernel-level pruning [11, 411, 495, 463, 147] refers to removing entire convolutional filters, thereby achieving channel-wise sparsity. Additionally, block-level pruning [348, 392, 430, 805] aims to trim multiple filters grouped by their similar inherent sparsity patterns. The filter-level pruning [452, 250, 473, 616, 821] focused on removing entire layers of blocks based on more sophisticated rules. In ViTs, pruning techniques

trim unimportant attention heads to improve efficiency during deployment. The redundancy of attention heads can define the importance scores to estimate the influence of each head [475, 168, 522, 263, 863, 614].

Quantization

Quantization aims to lower the number of bits required for storing weights or intermediate features by mapping representations to a small set of discrete values [639, 721, 759, 372]. Early quantization methods have been explored for convolutional neural networks. They can be divided into different categories: vector quantization that adopted code-book and quantization code for restoring the original data [298, 198, 695, 354], multi-bit quantization that substituted floating-point weights with low bit(s) directly [146, 664, 539, 396, 538, 741, 688], and hashing quantization that adopted a random weight sharing strategy to reduce the model redundancy substantially [91, 271, 781]. Recently, there has been growing interest in quantization schemes for ViTs [615], aiming to compress the inputs and weights by low-bit representations [523, 575, 442, 660].

Comparative Analysis

Pruning reduces parameter count and FLOPs, directly improving model size and theoretical computation. Quantization reduces memory bandwidth and enables integer acceleration on dedicated hardware. In practice, pruning is favored for aggressive FLOP reduction (e.g., edge devices with strict latency limits), while quantization is essential for maximizing throughput on commodity integer processors (e.g., mobile CPUs). They might be combined sequentially as a pipeline, i.e., pruned first and then quantized, to maximize efficiency.

2.3.3 Self-supervised/unsupervised learning

Self-supervised learning (SSL) [259, 645, 637] and unsupervised learning (USL) [568, 621, 54, 324, 149, 731] aim to leverage abundant unlabeled visual data [136] for learning robust representations. These approaches have been widely adopted in CNNs and ViTs to enhance generalization. USL and SSL share similar feature-learning pipelines. Therefore, we focus more on the more popular SSL approaches. In the early stages, SSL was applied to CNNs to learn rich visual representations for downstream tasks [511, 377, 857, 584, 792]. Later, contrastive learning, one of the most important frameworks in SSL, has been prominent and developed by VAEs [326, 546, 150, 582], SimCLR [87],

SwAV [640, 54], BYOL [206, 514], and MoCos [244, 93]. Recent works [81, 534, 366, 399, 789, 16, 84, 720, 479] also explored self-supervised schemes for ViTs. DeiT [626] introduced a distillation token-based approach to SSL that enables efficient training on smaller datasets. Likewise, DINO [55] proposed a self-distillation scheme that integrates knowledge distillation with SwAV, focusing on learning representations via clustering without requiring labels. BEiT [26] applied random masking of image patches (tokens) from BERT in NLP [320] to ViTs. Moreover, MAE [242] applied a masking technique, forcing ViTs to learn contextual relationships among patches by reconstructing masked image patches. In addition, some works that were originally developed for CNNs have been adapted for ViTs. MoCo v3 [95] adopted momentum-based contrastive learning to obtain robust image representations between different augmented views of the same image for ViTs using a memory bank. MoBY [719] combined MoCo v2 and BYOL for Swin Transformers [441]. PeCo [155] improved masked image modeling by optimizing image tokenization with a perceptual codebook. We briefly summarize the effectiveness of SSL Methods across domains. The efficacy of self-supervised learning (SSL) paradigms varies significantly across application domains, dictated by data characteristics and task objectives. In natural image understanding, contrastive learning methods (e.g., SimCLR, MoCo) excel by learning invariant representations from geometric and photometric augmentations, providing strong features for downstream classification and detection. In contrast, masked image modeling (MIM) methods (e.g., MAE, BEiT), which learn to reconstruct masked content, have shown superior performance on dense prediction tasks such as segmentation because they inherently model patch-level context and semantics. In medical imaging, where annotations are scarce and textures are subtle, MIM often outperforms contrastive learning because its reconstruction objective forces the model to learn fine-grained anatomical structures, thereby improving transfer to segmentation and detection tasks. In remote sensing, multimodal SSL that leverages geographic and temporal metadata alongside images has proven more effective than unimodal approaches. Meanwhile, for video data, contrastive methods that exploit temporal consistency (e.g., by contrasting frames from the same video clip) are dominant for action recognition. This domain-specific performance highlights that no single SSL method is universally optimal. Hence, selecting or designing a pretext task must align with the inherent structure and annotation constraints of the target data.

2.3.4 Knowledge Distillation

Knowledge Distillation (KD) is a machine learning technique that transfers knowledge from a source model (the teacher model) to a destination model (the student model). The general process of KD involves training the student network to mimic the teacher's intermediate or final outputs based on a loss function that measures mutual similarity. The transfer facilitates effective knowledge propagation from the teacher to the student. KD has been widely applied across various domains, including model compression and ensemble learning. As for model compression, KD can derive smaller but more efficient student models out of larger and more complicated teacher models for CNNs [46, 18, 549, 254, 86, 454, 620, 145, 1, 854, 859, 212, 302] and ViTs [485, 675, 477, 626, 304]. As for ensemble learning, KD can fuse information from multiple teacher models into a single but more powerful student model [185, 493, 13, 215, 7, 501, 61, 385, 317, 500]. Generally, KD is a versatile bridge between other techniques. It can recover from accuracy loss caused by aggressive pruning or quantization, distill knowledge from ensemble foundation models via NAS into a single model, or serve as the final fine-tuning step for a model trained via SSL.

2.3.5 Diffusion Models

The diffusion Model (DM) methodology uses CNNs and ViTs as building blocks to approximate data distributions. It defines a framework that learns to generate new data by adding and removing noise from the inputs. Specifically, diffusion models are iterative generative models based on *Markov chains*. Every chain begins with true data and ends with pure noise. Besides, each state corresponds to one noisy intermediate data point with increasing noise levels. A diffusion model consists of two stages: a forward diffusion stage and a reverse diffusion stage. In the forward diffusion stage, the input data is converted into pure noise by gradually adding Gaussian noise. In the reverse stage, a diffusion model learns to progressively recover the original input data from the pure noise using CNNs or ViTs as backbone architectures. The fundamental paradigm shift from Generative Adversarial Networks (GANs) to DMs represents a mathematical evolution in generative modeling. GANs [202], introduced by Goodfellow et al., formulate generation as a min-max game between a generator and a discriminator. The generator learns to produce realistic samples, while the discriminator attempts to distinguish real from generated data. The training objective seeks a Nash equilibrium that minimizes the Jensen–Shannon divergence between the

real and generated distributions [201]. However, GANs rely on an indirect adversarial mechanism: the generator and discriminator jointly learn implicit representations, and convergence is achieved when the two distributions become indistinguishable. The core idea is to use two mutually close approximations to the unknown real distribution. Despite their success, GANs suffer from well-documented limitations, including training instability (e.g., vanishing gradients when the discriminator becomes too strong), mode collapse (where the generator produces only a limited variety of samples), and difficulty in capturing complex multi-modal distributions. Diffusion models [124] emerged from principles of non-equilibrium thermodynamics and adopt a fundamentally different paradigm. Instead of adversarial training, they employ a denoising objective grounded in rigorous probabilistic theory. Diffusion models can be formulated within the framework of stochastic differential equations (SDEs) [586], unifying popular variants such as DDPM [260], DDIM [583], and NCSNs [585] as special cases. By optimizing a well-behaved surrogate likelihood bound (e.g., the variational lower bound or a simplified mean-squared error), DMs achieve stable training and comprehensive coverage of the data distribution without mode collapse, directly addressing the core weaknesses of GANs. Originally developed for data generation, DMs [483] have demonstrated performance comparable to that of GANs and VAEs [326]. Recently, they have been adapted for diverse computer vision tasks (e.g., image segmentation, object detection), offering enhanced robustness by modeling additive noise. We will elaborate on the applications of DMs in section 2.4. For more information on applications, please refer to the survey [237].

2.3.6 Deep Unfolding Techniques

Iterative optimization methods play a central role in deep learning, serving as the mathematical foundation for training, inference, and estimation. Despite interpretability and theoretical guarantees, conventional iterative optimization algorithms suffer substantial computational latency due to the complexity of hyperparameter tuning. Conversely, deep learning models offer powerful data-driven modeling capabilities but lack the structure, transparency, and efficiency needed for optimization-driven inference. Therefore, deep unfolding has recently emerged as a remedy to bridge these two paradigms by systematically transforming iterative optimization algorithms into structured, trainable ML architectures [574]. The hybrid framework, also known as the Deep Unfolding Network (DUN), can integrate prior domain knowledge into network design. Therefore, DUNs need relatively few trainable parameters in theory. The evolution of DUNs

demonstrates their role as a versatile meta-technique. Initially, DUNs were primarily incorporated into CNNs to tackle low-level tasks, such as compressed sensing [335] and image denoising [733], leveraging their efficiency in local feature extraction. Later, CNN-based DUNs have attracted attention in high-level vision, such as concealed object segmentation [240] and image super-resolution [811]. A significant advancement occurred as researchers began to integrate ViTs (or Vision-Language models) into the unfolding framework to capture long-range dependencies, which is crucial for tasks such as image restoration [796] and video processing [134]. The most recent trend focuses on designing hybrid modules within the DUN skeleton, combining convolutional layers for local processing and attention blocks for global interaction into a single, optimized pipeline [846]. This progression underscores DUNs as a flexible framework that can harness and enhance the advantages of underlying architectural components (CNNs, ViTs, or their hybrids) and other techniques (e.g., Diffusion [238]) to improve performance and interpretability across diverse inverse problems. Generally, DUNs are a prime example of hybridizing domain knowledge with data-driven learning. Their interpretable structure makes them particularly attractive for trustworthy deployment in scientific and medical domains. For a comprehensive exploration of deep unfolding methodologies, please refer to the detailed survey [135, 574].

2.3.7 Section Summary

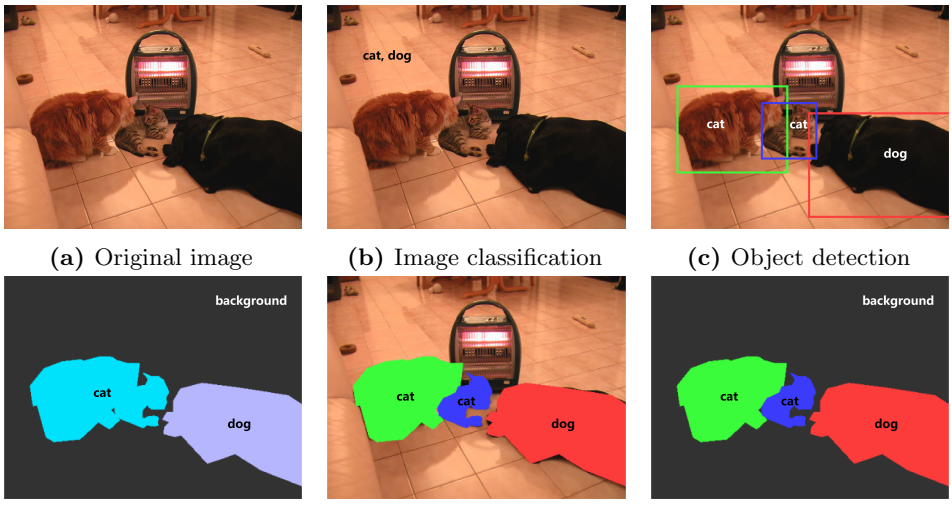
Some techniques have different intrinsic characteristics. For instance, SSL and DM represent different points on the spectrum of data utilization: SSL condenses information from real data into representations, while DMs expand the data distribution through synthesis. In practice, these techniques are rarely used in isolation. A common pipeline might involve: 1) SSL is the workhorse for pre-training general-purpose representations, especially in label-scarce domains. 2) DM can use SSL-enhanced representations to synthesize high-fidelity samples for data augmentation or to provide robust generative priors as fake labels. 3) NAS can discover an efficient backbone for a target task. 4) Compressing the model via pruning and quantization for deployment, and 5) Employing KD to fine-tune the compressed model and recover performance from the original large model as a teacher. 6) DUNs inject strong domain priors into optimization for structure transparency. Understanding their complementary strengths and formal interconnections is crucial for building efficient, robust, and deployable vision systems.

2.4 Applications

Due to their ability to automatically fit complex data, deep learning models have been widely applied to computer vision tasks and have achieved good performance. In this section, we introduce the numerous applications of different model types and reveal trends in architecture selection based on dataset properties.

2.4.1 Classification

Classification is the most fundamental task in computer vision. It aims to predict class labels at multiple levels: image-level (class presence), pixel-level (semantic segmentation), and instance-level (instance segmentation). As depicted in Figure 2.7, Image-level classification identifies the existence of classes but lacks spatial localization. Semantic segmentation refines this by partitioning the image into disjoint regions corresponding to semantic classes. Instance segmentation further distinguishes individual instances within each semantic class.



(a) Original image **(b)** Image classification **(c)** Object detection

(d) Semantic segmentation **(e)** Instance segmentation **(f)** Panoptic segmentation

Figure 2.7: Classification hierarchy from coarse-grained to fine-grained inference: classification at image level, detection or localization at part/region level, semantic segmentation, instance segmentation, and panoptic segmentation all at pixel level.

Image-level classification

Large-scale datasets like ImageNet [138] have driven the widespread adoption of CNNs for image classification. Their ability to automatically extract complex information from data has surpassed traditional image analysis algorithms. Inspired by LeNet-5 [351], AlexNet [340] was a pioneering CNN-based classification architecture that achieved superior performance on ImageNet. The automation sparked the enthusiasm of subsequent researchers for designing more complex architectures, driven by the importance of stacking additional layers. VGG [577] designed deeper networks by stacking more convolutional and fully connected layers sequentially, gaining improvement in classification tasks. At the same time, GoogLeNet [608] was designed with wider networks using an inception module inspired by NiN [410]. Since replacing all fully connected layers, GoogLeNet has outperformed VGG with around one-eighth the number of parameters. Several improvements to the inception module [610, 288] advanced the accuracy of image-level classification. As depth increases, CNNs can suffer from vanishing or exploding gradients. ResNet [248] introduced a skip connection to interact with arbitrary pairs of layers at different depths and enabled training a deep CNN up to 1000 layers. The core innovation lies in parameter-free residual functions, which bypass intermediate layers to stabilize gradient flow and mitigate vanishing gradients. The residual function is almost necessary for deep learning models and has been improved by the following research [607, 827, 99, 716, 277, 758]. In addition, capsule networks [559, 256, 220, 2] adopted steerable convolutions and introduced equivariance to geometric transformations in image classification.

Later, visual transformers became popular alternatives to CNNs for image-level classification due to their self-attention mechanisms, which capture long-range dependencies. Vision Transformer [156] proposed the first ViT framework for image classification. The successive ViTs focus on tokenization improvement, multi-scale feature extraction, self-attention simplification, and training efficiency enhancement (see more details in section 2.2.2). Most recently, hybrid architectures have gained popularity by combining the strengths of both CNNs and ViTs [815, 384]. There are two main branches, i.e., enhancing CNNs with the attention (Attn-CNNs) mechanism and Transformers with Convolution (Con-Trans). Attn-CNNs introduce self-attention mechanisms into hierarchical feature representations to capture global context information at different scales and model generic object relations to understand complex scenes (refer to subsection 2.2.3). Con-Trans incorporates convolutional operations into ViTs to build multi-scale features and alleviate the data-hungry problem during

training (refer to subsection 2.2.3). Some significant work on image classification tasks is summarized in the Table 2.1, including CNN-based, ViT-based, and hybrid models.

Pixel-wise classification

Pixel-wise classification is more conceptually difficult than image-level classification because disjoint spatial partitions are needed to locate the present entities. Depending on the partition granularity, semantic and instance segmentation are defined. More specifically, semantic segmentation assigns each semantic class to a single disjoint partition, which may contain multiple instances. Instance segmentation pushes the concept of segmentation one step further, identifying clues to individual semantic objects and separating disjoint semantic partitions into non-overlapping instance regions at finer granularity. However, many semantic classes (uncountable regions such as sky, sea, grass, etc.) cannot be further divided into individual entities, unlike other semantic classes (countable entities such as cat, dog, person, etc.). Therefore, panoptic segmentation [329] unifies uncountable "stuff" classes and countable "thing" classes into one segmentation framework to deal with all possible semantic entities. CNNs have developed specialized frameworks for segmentation tasks, whereas ViTs have enabled general-purpose frameworks for all pixel-level classification tasks. Therefore, we will introduce task-specific and generic architectures separately. We organize task-wise models in the Table 2.2 for the semantic, instance, and panoptic segmentation, respectively. Moreover, some representative universal models are listed in Table 2.3.

A. Semantic Segmentation

The success of CNNs in image-level classification motivated researchers to explore the capabilities of pixel-level labeling on semantic regions. The key advantage of CNNs, which enables them to outperform traditional methods, is their ability to learn appropriate feature representations from a particular dataset in an end-to-end manner, rather than relying on handcrafted features that require extensive domain expertise and manual parameter tuning to adapt to the specific scenario. However, CNNs for image-level classification struggle to achieve good partitions since their sequential/linear architectures are not designed to extract spatial information at different scales. Therefore, CNNs for semantic segmentation [672] have to find rich details about locations and appearances to infer the pixel-wise affiliation at different depths. With residual skip connections, the following research focuses on designing architectures that integrate local and global information across multiple scales. FCN

		Model	Dataset	Year	Publication	Top-1 Acc(%)	GFLOPs	#Param (M)
Image Classification	CNN	AlexNet[340]	ImageNet-1K	2012	NeurIPS	63.3	1.45	62
		VGG[577]		2015	ICLR	75.6	15.5	144
		GoogLeNet[608]		2015	CVPR	68.3	1.5	6.8
		Inception-v3[610]		2016	CVPR	77.2	11.4	23.8
		ResNet-50[248]		2016	CVPR	76.1	4.1	25.6
		ResNet-101[248]		2016	CVPR	77.4	7.9	44.7
		Inception-v4[607]		2017	AAAI	80.1	13.4	55.8
		PolyNet[827]		2017	CVPR	81.3	20.8	94.7
		DPN[99]		2017	NeurIPS	80.7	11.7	61.7
		ResNeXt[716]		2017	CVPR	79.6	8.0	83.7
		DenseNets[277]		2017	CVPR	77.8	7.7	28.9
		CliqueNet[758]		2018	CVPR	76.0	8.5	14.4
		MobileNetV3[264]		2019	ICCV	75.2	0.22	5.4
		Capsule-Net[559]	CIFAR-10	2017	NeurIPS	92.1	0.23	4.2
		Capsule-EM[256]		2018	ICLR	88.5	0.17	0.8
		CapsNet[220]		2019	NeurIPS	92.2	0.14	3.2
		Star-caps[2]		2019	NeurIPS	91.2	0.51	0.3
	ViT	Vision Transformer[156]	ImageNet-1K	2021	ICLR	79.8	17.6	86.6
		CaiT-S[627]		2021	ICCV	83.3	13.9	48.0
		XCiT-L[6]		2021	NeurIPS	86	417.9	189
		DeiT-B[626]		2021	ICML	81.8	17.6	86.6
		DINO[55]		2021	ICCV	78.2	17.6	85
		Swin-B[441]		2021	ICCV	83.3	15.4	88.0
		Performer[113]		2021	ICLR	76.5	10.2	87
		PiT-B[252]		2021	ICCV	84	12.5	73.8
		CrossVit[66]		2021	ICCV	82.8	9.5	44.3
		SeT-33[604]		2022	PAMI	83.3	7.7	40.3
		KVT-PB[663]		2022	ECCV	82.6	12.5	74
		DAT-B[707]		2022	CVPR	84.8	22.6	100
		TaT[413]		2022	CVPR	72.4	2.7	13.8
		DW-B[544]		2022	CVPR	83.8	17.0	91
		VSA[817]		2022	ECCV	83.9	16.0	88
		CrossFormer++[676]		2023	PAMI	85.0	21.8	96
		Dual-ViT[763]		2023	PAMI	85.7	18.0	73
	Hybrid	SENet-R50[268]	ImageNet-1K	2018	CVPR	78.4	4.4	30.5
		FANet[533]		2019	NeurIPS	77.6	7.2	18.0
		SAN[836]		2020	CVPR	78.2	3.3	20.5
		LeViT[203]		2021	ICCV	82.5	2.3	39.4
		ConViT[161]		2021	ICML	82.5	30	152
		ResNeSt-50[807]		2022	CVPR	81.1	5.39	27.5
		MPVit-B[359]		2022	CVPR	84.3	16.4	74.8
		CMT-L[213]		2022	CVPR	84.8	19.5	74.7
		Swin-ACMix-S[506]		2022	CVPR	83.5	9.0	51
		Dilate-B[312]		2023	TMM	85.6	31.1	48
		Conformer-B[515]		2023	PAMI	84.1	46.6	83.3
		SwiftFormer-L3[567]		2023	ICCV	83.0	4.0	28.5
		EfficientViT-B2[47]		2023	ICCV	82.7	2.1	24
		SwiftFormer-L3[567]		2023	CVPR	83.0	4.0	28.5

Table 2.1: A comparative analysis between different models on Image-level classification in terms of parameter complexity and Top-1 (%) Accuracy on ImageNet-1K. For a direct comparison, we consider models with input of size 224×225.

[444] is a milestone in semantic segmentation, enabling CNNs to be trained end-to-end. It replaced the fully connected layers with 1×1 convolutional layers, inspired by NIN [410], to generate pixel-wise predictions. Additionally, it introduced deconvolution [795, 793] as an up-sampling operation to learn how to restore spatial information from hierarchical feature maps, rather than relying on fixed interpolation methods. However, FCN adopted complicated models (AlexNet, VGG, GoogLeNet, ResNet, etc.) as encoders for feature extraction but used a very simple decoder for inference, limiting its performance. The following research explored integrating feature maps from multiple paths into decoders to capture object location and appearance cues across different scales. U-Net [551] and its variants [20, 510, 63, 529, 299, 10, 151] employ a ladder-shaped architecture that facilitates bilateral information flow from encoder layers to their corresponding decoder counterparts. This design effectively combines low-level local cues with high-level global context. To merge information from multiple paths, feature pyramid structures [532, 555, 36, 68, 406] adopted different down-sampling and up-sampling strategies to obtain rich information flows more flexibly, like dilated convolution [770, 661, 510, 79] expands receptive fields with dilation rates to collect statistics from wilder scope at the cost of memory consumption, and spatial pyramid pooling [435, 79, 837, 21, 80, 694, 657, 706] to capture contextual information from different receptive field sizes for semantic regions of varying scales. The multi-path structures allow the insertion of pixel-wise loss functions at arbitrary depths for deep supervision. In addition, traditional probabilistic models, like Conditional Random Fields [338, 77, 845, 58, 78], random walk [35, 309], were adapted into deep learning to capture more representative features to support pixel-level inference. Most recently, CNNs [656] integrated with knowledge distillation techniques to enhance segmentation performance.

Some ViTs have been specifically applied to semantic segmentation and use self-attention to model long-range pixel interactions. The ViTs [847, 712, 595, 6, 671, 51, 800] follow the standard ViTs that divide an input image into patches by a tokenization process and generate better feature representations with self-attention. Then, an additional semantic segmentation module is required to produce pixel-wise semantic masks. In addition, many hybrid models have achieved impressive performance on semantic segmentation. Attn-CNNs [785, 678, 769, 806, 287, 184, 494] adopt an attention mechanism to capture long-range context and predict semantic masks. Con-Trans [73, 592, 632, 674, 47, 314, 294] replace the patch tokenization process with CNNs to better locate meaningful semantic regions, and use self-attention to obtain enhanced semantic feature maps.

B. Instance Segmentation

Unlike semantic segmentation, instance segmentation requires finer-grained location information to divide semantic regions into individual entities further. The purpose is to reason about spatial relationships among various instances to better understand the scene. Currently, it remains challenging for CNNs to represent each entity with location and semantic information. Like object detection, instance segmentation can be divided into one-stage and two-stage methods. As for two-stage methods, CNNs developed instance segmentation upon object detection approaches, which detect and locally segment sequentially: SDS [229], MCG [12], SharpMask [517], MultiPath [790], Instance-FCN [125], Mask R-CNN [245], PANet [433], MaskLab [76], Mask Scoring R-CNN [284], Cascade Mask R-CNN [50], HTC [74], BlendMask [70], CondInst [622] and Refinemask [804]. As for one-stage instance segmentation, CNNs build grid structures and use each grid point to represent nearby potential instances: YOLACT [39], TensorMask [94], SOLOs [679, 680], PointRend [331]. Moreover, Deep Snake [513] offers a different perspective on anchor-free instance segmentation by adapting active shape models to deep learning. Few transformer-based approaches have focused on instance segmentation rather than generic segmentation. QueryInst [178] adopts shared queries to represent all possible instances within the dataset and uses cross-attention to find instance-wise responses. In addition, some CNNs with attention mechanisms offer novel perspectives on modeling long-range object relations. They enhanced instance segmentation tasks [360, 516, 707, 319, 506, 861] with an attention mechanism to highlight informative channels or locations, thereby improving the representation of potential candidate entities.

C. Panoptic Segmentation

Panoptic segmentation [329] further extends the partition criteria and introduces two auxiliary classes, "things" and "stuff" tags, corresponding to "countable" and "uncountable" concepts in natural language. The added intermediate classes divide the original semantic classes into two groups, and panoptic segmentation employs two selection branches: one for semantic segmentation (handling 'stuff' classes) and another for instance segmentation (handling 'thing' classes). This approach unifies semantic and instance segmentation tasks within a joint framework. Therefore, panoptic segmentation must define flexible architectures that can select between semantic and instance segmentation to assign each pixel a semantic label or an instance ID based on a given auxiliary "things" or "staff" tag. Initially, panoptic segmentation models were

heavily based on CNNs, leveraging their ability to extract hierarchical spatial features. It is challenging for CNNs to jointly represent the concepts of "staff" and "things". The early CNNs [519, 520, 708], such as Panoptic FPN [328], UPSNet [724], DeeperLab [752], AdaptIS [580], Panoptic-DeepLab [104], EfficientPS [481], and LPSNet [262], are ensembles of existing instance segmentation (like Mask R-CNN) and semantic segmentation architectures (like DeepLab) and fuse their results directly. Later, CNNs adopted unified architectures that concentrated on determining the most effective ways to partition "things" and "stuff" together. DR1Mask [69] proposed a novel dynamic convolutional module called dynamic rank-1 convolution to merge high-level context information with low-level detailed features. Panoptic FCNs [390, 389] generated kernel weights for each entity that fused dual information from position and kernel heads. Then, each generated kernel can locate and predict both "things" and "stuff" entities directly on high-resolution feature maps from the encoder. In addition, ViTs have adopted convolution operations for panoptic segmentation [88, 22, 877]. Panoptic Segformer [400] decoupled queries into "stuff" and "things" groups and inserted extra location information into "things" queries only. Then, it proposed a unified mask decoder to predict panoptic segmentation for two query groups. kMaX-DeepLab [774] replaced the computationally intensive computation of key-query correlation with a simple k-means clustering, thereby improving the efficiency of panoptic segmentation. Panoptic-partformer[382, 383] formulated panoptic segmentation and part segmentation as a unified mask prediction and classification problem. Moreover, hybrid architectures have attracted research interest in panoptic segmentation [382, 281]. CNNs can integrate attention mechanisms to model long-range object relations. Axial-DeepLab [655] introduced position-sensitive self-attention into CNNs and designed axial attention to reduce computational complexity. EPSNet [60] introduced cross-layer attention to generate dynamic prototypes to represent different entities and predict semantic and instance masks separately. Max-DeepLab [654] was the first end-to-end model. It proposed path-transformer architectures and directly predicted class-labeled masks for "stuff" and "things" entities. CMT-DeepLab [772] built upon the Clustering Mask Transformer and viewed both "stuff" and "things" entities as cluster centers, generating pixel affiliation with cross-attention mechanisms. Mask-DINO[369] extended DINO [808] with a mask branch and unified panoptic segmentation with object detection.

D. Universal Image Segmentation

Developing a generic framework to partition images into multiple perceptual levels

Model				Dataset	Year	Publication	metric (%)	GFLOPs	#Params (M)
Semantic Segmentation (mIoU)	CNN		FCN[444]	ADE20K	2015	CVPR	29.4	178	134
			U-Net[551]		2015	MICCAI	33.9	65	31
			RWN[35]		2017	CVPR	37.2	210	141
			G-CRF[58]		2017	ICCV	37	92	37
			DeepLab-v3+[78]		2018	PAMI	44.1	255	63
			HRNet-v2[657]		2020	PAMI	49.2	695	65.9
	ViT		Light-Ham[193]		2021	ICLR	51.5	71.8	61.1
			Segmenter[595]		2021	ICCV	53.6	380	307
			XciT[6]		2021	NeurIPS	48.4	200	32
			SegFormer-B5[712]		2021	NeurIPS	51.8	84.0	183
			SegViT-v2*[800]		2022	IJCV	55.7	308.8	234
			TransKD*[432]		2024	ITSS	50.7	12	80
	Hybrid		NL*[678]		2018	CVPR	44.7	300	54
			DANet[184]		2019	CVPR	45.3	250	69.0
			OCRNet[784]		2021	IJCV	48.0	260	71.0
			HMANet*[494]		2021	TGRS	44.2	180	55
			MaxViT-L*[632]		2022	ECCV	51.1	245.4	212
			Semask[294]		2023	ICCV	53.5	356	212
Instance Segmentation (mAP)	CNN	one-stage	YOLACT[39]	MSCOCO	2019	ICCV	29.8	45.1	30.8
			TensorMask[94]		2018	CVPR	37.3	71.6	44.7
			SOLO[679]		2019	ICCV	40.5	172.2	50.7
			SOLO-v2[680]		2020	NeurIPS	41.7	190.3	63.7
			PointRend[331]		2020	CVPR	36.3	284.9	46.7
			DeepSnake[513]		2020	CVPR	30.3	24.8	21.5
		two-stage	Mask R-CNN[245]		2017	ICCV	38.5	266	44.4
			PANet[433]		2018	CVPR	42.0	286.8	49.1
			HTC[74]		2019	CVPR	41.2	324.6	58.3
			BlendMask[70]		2020	CVPR	41.3	256	42.4
			CondInst[622]		2020	ECCV	39.1	185	38.5
			Refinemask[804]		2021	CVPR	41.5	360	61.2
	Hybrid		CenterMask-Lite[360]		2020	CVPR	38.3	24.6	96.9
			Conformer[516]		2021	ICCV	40.74	457.7	56.9
			DAT-B[707]		2022	CVPR	45.8	1003	145
			MaxViT-B[632]		2022	ECCV	45.7	346.8	92.5
			Mask-Transfiner[319]		2022	CVPR	41.6	326.7	62.5
			Biformer-B[861]		2023	CVPR	43.7	245.3	58.7
Panoptic Segmentation (PQ)	CNN		Panoptic FPN[328]	Cityscapes	2019	CVPR	58.1	275	63.0
			AdaptIS[580]		2019	ICCV	59.0	76.4	32.6
			Panoptic-DeepLab[104]		2020	CVPR	63.1	144.9	44.3
			LPSNet[262]		2021	CVPR	61.9	80	18.9
			Panoptic FCN[389]		2022	PAMI	61.4	213.0	62.5
			Mask-Pyramid[708]		2024	Sensors	65.2	215	49.2
	ViT		kMaX-DeepLab[774]		2022	ECCV	68.4	285.0	54.1
			Panoptic-Segformer[400]		2022	CVPR	65.1	205.0	62.6
			Center-Trans[22]		2023	Electronics	65.3	428	72.0
			Panoptic-partformer[382]		2022	ECCV	64.7	249	56
			Pix2Seq-D[88]		2023	CVPR	64.9	1120	78.0
			PanSR[877]		2024	arXiv	67.1	305	54.8
	Hybrid		Axial-DeepLab[655]		2020	ECCV	62.1	55.6	178.5
			EPSNet[60]		2020	ACCV	60.8	98.7	24.3
			Max-DeepLab[654]		2021	CVPR	63.5	214.0	360.0
			CMT-DeepLab[772]		2022	CVPR	67.8	238	48.1
			Mask DINO[369]		2023	CVPR	64.6	987.3	72.5
			BEE-Net[281]		2024	ICRA	65.0	305.7	45.3

Table 2.2: A comparative analysis between different models on segmentation

Model			Year	Publication	ADE20K mIoU(%)	COCO mAP(%)	COCO PQ(%)	#Params (M)
Universal Segmentation	ViT+Hybrid	K-Net[822]	2021	NeurIPS	50.6	40.1	48.3	56.3
		Mask2Former[105]	2022	CVPR	56.1	50.1	57.8	216
		SEEM[876]	2023	NeurIPS	-	46.3	56.1	40.0
		OneFormer[293]	2023	CVPR	58.1	49.2	58.0	223
		DaTaSeg[208]	2023	NeurIPS	54.0	-	53.5	-
		DFormer[653]	2023	arXiv	48.3	44.4	52.5	44
		ViT-P[566]	2025	arXiv	58.6	49.5	58.0	309

Table 2.3: A comparison of universal segmentation models.

(e.g., scene, object, part) has recently become an active research area [710]. Compared with the task-specific framework, the generic framework must define a common entity representation and a universal pixel-level classification process shared across semantic, instance, and panoptic segmentation tasks within a single model. As for CNNs with attention mechanisms, K-Net [822] adopted a group of learnable kernels as the common entity representation, where each kernel can dynamically update meaningful groups conditioned on the input image. Therefore, K-Net allows contextual information to flow among kernels via multi-head attention and establishes a one-to-one mapping between dynamic kernels and entities, generating one mask per potential "things" or "stuff" partition. As for ViT-based approaches, many researchers are developing unified frameworks that use learnable embeddings (also called queries) and cross-attention to integrate entity representation, long-range entity relation inference, and pixel-wise classification. MaskFormer [106] introduced a set of mask embeddings associated with a global binary prediction for each entity. Mask2Former [105] proposed masked cross-attention for a new architecture capable of addressing universal image segmentation. OneFormer [293] proposed a task-conditioned joint training strategy that enables training on ground truths of each domain (semantic, instance, and panoptic segmentation) within a single multitask training process. Integrated a task token into the task-conditioned prediction head, OneFormer can support multi-task training and inference based on the given task. DFormer [653] viewed the universal image segmentation task as a denoising process using a diffusion model. By adding Gaussian noise at various levels to ground-truth masks, DFormer used the noisy masks as inputs and gradually learned to predict denoising masks using a diffusion-based decoder. SEEM[876] proposed a promptable and interactive model for segmenting everything everywhere all at once in an image. It used a text encoder to encode text queries and mask labels into the same semantic space. Dataseg[208] trained a universal multi-dataset multi-task segmentation model with a shared representation between mask proposals and class predictions for all tasks. ViT-P[566] introduced a novel two-stage

segmentation framework that decouples mask generation from classification for accurate mask classification, especially in the presence of ambiguous boundaries and imbalanced class distribution.

Evolution trends and practical deployment gaps

The evolution of image classification models follows a clear trajectory from pure CNNs to Vision Transformers and, finally, to hybrid architectures, driven by the need to balance local feature extraction with global context modeling. Early CNNs excelled in efficiency and translation equivariance but struggled with long-range dependencies. ViTs introduced global attention for superior context capture, but at a high computational cost and substantial data consumption. Hybrid models have primarily combined the convolutional inductive bias with the attention mechanism's flexibility, achieving state-of-the-art accuracy with manageable FLOPs. However, key deployment gaps remain: (1) efficiency–accuracy trade-off for lightweight models (MobileNetV3, EfficientViT) that sacrifice accuracy for speed, limiting their use in high-stakes applications; (2) hardware optimization for many hybrid designs that are not yet optimized for mobile or edge deployment due to irregular memory access patterns; (3) generalization under domain shift of models trained on curated datasets (ImageNet) that often degrades in real-world noisy environments. Future work must focus on hardware-aware co-design and dynamic inference to adapt the computational cost to the input.

2.4.2 Object Detection

As a fundamental task in computer vision, object detection is the key block for more complex tasks such as autonomous driving [428, 872, 762, 588], robotics [740, 426, 231], surveillance [59], pose / behavior estimation [375, 269, 842], etc. The core difficulty in object detection is processing two distinct sources of information: spatial information (the location of each target object) and appearance information (the pattern of each object for category classification). Compared to image classification, object detection operates at a regional level, where each object is localized with a bounding box. This approach bridges the gap between semantic segmentation (pixel-level) and instance segmentation (instance-level), simplifying the task by avoiding pixel-wise annotations. Although regional representations require fewer parameters and simpler annotations for training, this comes at the cost of reduced spatial exclusivity across different instance regions. The compromise means that bounding boxes may overlap in crowded scenes, leading to ambiguities in instance separation. In this subsection, we introduce the

development of generic object detection and then elaborate on one specialized sub-task, small object detection. We summarize the key models for generic and small-object detection in Table 2.4 for easier comparison.

Generic object detection

Generic object detection has been an active research area in computer vision for several decades, aiming to detect a broad range of general categories [429]. Although many traditional algorithms have been proposed to address the challenging problem, deep learning techniques have obtained significant attention for learning feature representations automatically from data, further advancing the development of object detection.

At an early stage, CNNs, as powerful tools for automating feature learning, have been a major focus of research aimed at addressing the core difficulty of object detection: how to represent spatial and appearance information. Then, set loss functions are used to solve the many-to-one mapping problem, assigning each ground-truth object to the most likely candidate among many others. Around the representation schemes, CNN-based detectors were organized into two main categories: two-stage frameworks extract different information sequentially, while one-stage frameworks represent features in parallel. The two-stage methods consist of two sub-networks. The former sub-network represents spatial information as regional proposals, e.g., using RPN (region proposal network), and the latter extracts appearance information within each region. The two-stage methods can generate sparse proposals flexibly because RPN over-samples only the potential regions and ignores the rest, thereby achieving high precision on average. Sorted by release time, two-stage CNNs mainly include R-CNN [196], SPP-Net [247], Fast R-CNN [197], Faster R-CNN [545], ION [31], HyperNet [334], R-FCN [126], FPN [414], DCN [127], Mask R-CNN [245], ZIP [371], Light R-FCN [398] and Cascade RCNN [503, 49]. Alternatively, the one-stage method employs a unified regression framework. Each position in the generated feature maps encodes spatial coordinates (e.g., bounding boxes) and appearance information (e.g., class probabilities). Therefore, it utilized a grid structure where each grid point serves as an anchor corresponding to a potential object, resulting in dense predictions. Due to the shared feature maps, the one-stage methods have lower computational complexity in resource-limited scenarios. The one-stage CNNs mainly include MultiBox [164], DetectorNet [609], OverFeat [565], AttentionNet [767], YOLO [540], G-CNN [488], SSD [436], DSSD [181], RON [333], YOLOv2 [541], STDN [853], FSSD [402], RefineDet [820], RFBNet [434], YOLOv3 [542], CornerNet [346], CenterNet [160], FCOS [623] and other improved YOLOs

[446, 409, 224, 279, 38, 295, 649, 647]. However, CNN-based detectors cannot solve the ambiguities in bounding box assignment. Some post-processing steps, such as non-maximum suppression (NMS), must be applied during inference to eliminate false-negative candidates near each true positive.

Later, the attention mechanism was explored in object detection, as query embeddings can be object proposals that efficiently combine spatial and appearance information. Therefore, attention mechanisms can capture long-range interactions among potential object proposals and infer complex contextual patterns across different images. However, the pure ViT-based detectors have obtained limited research interest because their characteristics are less effective for object detection. Specifically, the patch tokenization process cannot provide sufficient and precise object proposals. At the same time, the plain and non-hierarchical architecture cannot generate multi-scale features to handle diverse shapes and sizes [659]. YOLOS [176] adapted its patch tokenization process to provide sufficient *Det-Tok* that are learnable query tokens for object binding. Moreover, CSWin Transformer [154] and ViTDet [387] explored hierarchical structures to generate multi-scale feature maps.

Hybrid architectures that combine CNNs and ViTs have recently emerged as the primary research direction for improving object detection performance [825, 633]. The hybrid frameworks can be grouped in two ways: (a) ViT backbones are incorporated with various CNN detection heads (e.g., Mask R-CNN [245], Cascade R-CNN [49], FCOS [623]) to perform object detection. ViT-FRCNN [30] added a ViT-based backbone to a Faster R-CNN object detector for better generalization ability. Similar to FPN [414], FPT [803] combined a non-local network with multi-scale features, while PVT [673, 674] introduced a high-to-low resolution process into RetinaNet [554] to improve detection performance. Moreover, Swin Transformer [441], SwinV2 [440], ViL [816], and Focal Transformer [747] constructed a local-to-global combination to efficiently extract both local and global dependencies with less computational overhead. (b) CNN backbone with ViT-based decoder [844, 462]. DETR [53], as a pioneer for transformer-based detectors with CNN backbones, defined a small fixed number of static learnable embeddings (called object queries) and adopted cross-attention to learn long-range dependencies between queries and feature maps. By using set loss functions for object prediction, DETR eliminated traditional handcrafted components, such as RPN and NMS post-processing. The following research aims to improve DETR's convergence speed, computational complexity, and other aspects. ACT [844] adopted a locality sensitivity hashing (LSH) method to cluster the query features for a lower computational burden adaptively. Inspired by deformable convolutions [127],

			Model	Dataset	Year	Publication	mAP (%)	GFLOPs	#Params (M)
Generic Object Detection	CNN	one-stage	CenterNet[160]	MSCOCO	2019	ICCV	47.0	-	-
			FCOS[623]		2020	PAMI	44.7	-	-
			RepPoints-v2[96]		2020	NeurIPS	49.4	-	-
			EfficientDet[613]		2020	CVPR	52.6	-	-
			PP-YOLOE[735]		2022	arXiv	50.9	110.1	52.2
			YOLOv10[647]		2024	NeurIPS	29.5	160.4	29.5
		two-stage	Faster RCNN[545]		2015	NeurIPS	42.7	-	-
			Mask RCNN[245]		2017	ICCV	37.1	63.8	248.0
			DCN[127]		2017	ICCV	37.5	-	49.5
			Cascade RCNN[49]		2018	CVPR	42.8	257.0	87.1
			Dynamic RCNN[809]		2020	ECCV	49.2	-	-
			Sparse RCNN[603]		2021	CVPR	51.5	-	-
	ViT		YOLOS[176]		2021	NeurIPS	42.0	538	127
			CSWin[154]		2022	CVPR	48.7	526	97
			ViTDet[387]		2022	ECCV	58.7	3600	692
			Swin-v2[440]		2022	CVPR	56.4	1043	160
			DINO[808]		2023	ICLR	63.2	860	218
			ViT-Adapter[101]		2023	ICLR	52.1	-	158
	Hybrid		DETR[53]		2020	ECCV	44.9	253	60
			FPT[803]		2020	ECCV	42.6	88.2	346.2
			Focal Transformer[747]		2021	NeurIPS	49.0	533	110
			Deformable DETR[870]		2021	ICLR	46.2	173	40
			Swin-ACMix[506]		2022	CVPR	51.1	754	58
			Next-ViT-L[373]		2022	arXiv	48.0	391	77.9
Small Object Detection	CNN	one-stage	Polardet[839]	DOTA	2021	IJRS	76.6	-	-
			R^3Det[753]		2021	AAAI	71.9	264.2	37.8
			S^2ANet[222]		2021	TGRS	74.2	287.0	39.7
			GWD[754]		2021	ICML	76.3	336	41.9
			KLD[755]		2022	PAMI	77.3	336	41.9
			KFIoU[756]		2023	ICLR	74.2	287.0	39.7
		two-stage	O-RCNN[718]		2021	ICCV	76.3	199	41.1
			ReDet[223]		2021	CVPR	76.3	-	31.6
			CFA[218]		2021	CVPR	76.7	-	-
			AOPG[107]		2022	TGRS	75.4	-	-
			ARC[524]		2023	ICCV	77.4	-	74.4
			PKINet-S[48]		2024	CVPR	78.39	70.2	30.8
	ViT+Hybrid		ViTDet[387]		2022	ECCV	74.4	502	103.2
			RVSA[650]		2023	TGRS	81.2	414	114.4
			ViTDet-OBB[659]		2023	IGARSS	75.7	-	-
			AO2-DETR[128]		2023	CSVT	77.7	-	-
			Strip R-CNN[783]		2025	arXiv	80.6	52.3	13.3
			LSKNet[397]		2025	IJCV	81.6	161	31.0

Table 2.4: A comparative analysis between different models on detection.

Deformable DETR [870], and DINO [808] introduced deformable self-attention that samples on a sparse set of significant positions across all scales, reducing computational complexity. TSP [606] and SMCA [189] explored accelerating the convergence by improving the cross-attention mechanism. Conditional DETR [472] proposed a spatial

prior, called conditional spatial embedding, to narrow down the valid search range for localizing distinct regions. Efficient-DETR [764] introduced an additional region proposal network into the detection pipeline as a dense prior to improve initialization. Anchor DETR [682] replaced object queries with anchor points to predict multiple objects at one position and proposed a variant of attention for lower computational complexity. Different from the DETR series, Pix2Seq [89] is a generalized Transformer-based framework that uses a quantization and serialization scheme to convert bounding boxes and class labels into a sequence of discrete tokens. Diffusion efficiently represents potential candidates of small objects without handcrafted anchors in a one-stage framework equipped with dynamic anchors [814, 173, 438]. DiffusionDet was the first work to introduce diffusion models into object detection. It used a forward diffusion process to generate normalized box coordinates from Gaussian noise and adopted cross-attention blocks to learn how to predict high-quality boxes by denoising iteratively.

Small Object Detection

Small object detection (SOD)[3] [108] is a specialized object detection task that gains increasing popularity due to fast-growing data acquired from drone-view images, remote sensing satellite images, High-altitude surveillance images, and high-throughput microscopy images [458]. In addition to common difficulties in generic object detection, SOD has demonstrated unique challenges: (1) Image acquisition from high altitudes generates high-resolution images due to the wide-angle lens, causing computational difficulty in a detector [49], especially on mobile devices; (2) Fitting large data in detectors with down-sampling operations inevitably deteriorates representation details of small-sized objects; (3) The aerial viewed images are characterized severely uneven information densities because overcrowded objects normally occupy a minor proportion, imposing another request of efficient detection on key regions [421]; (4) Complicated situational factors, such as occlusion, background clutter, and various lighting conditions, further aggravate detection precision. Therefore, generic detectors cannot locate and classify small objects, which are typically densely packed in images, without dedicated adaptation. To tackle the dilemma between information density and efficiency, many works aim to restore missing information about small objects, focus on crowded regions, and extract long-range contextual information among objects. The Focus-and-Detect approach [745, 159, 336] was a two-stage framework that found the crowded regions and restored the missing information with super-resolution techniques

[3] In this task, the term *small* can be defined by thresholds of area [416] or length [751, 776].

[505, 137, 29, 192]. Another line of one-stage detectors fused feature maps at different scales to enhance the representation of small objects [97, 527, 199, 261, 744]. In addition, attention mechanisms have been introduced into SOD detectors for complicated context modeling [388, 182, 449, 283, 535, 447, 427]. Most recently, diffusion models have been introduced to efficiently represent potential candidates for small objects without handcrafted anchors [677, 658].

Evolution trends and practical deployment gaps

Object detection has evolved from two-stage CNN frameworks (e.g., Faster R-CNN) to one-stage anchor-free models (e.g., YOLO, FCOS) and recently to Transformer-based detectors (e.g., DETR). The shift reflects a trade-off between precision and inference speed, with one-stage models favoring real-time performance and transformers offering end-to-end training without handcrafted components. For small object detection in aerial/remote-sensing imagery, multi-scale feature fusion and attention mechanisms are critical, yet they increase computational cost. Deployment gaps include: (1) scalability on high-resolution images for costly dense prediction; (2) robustness to occlusion and scale variation, especially in crowded scenes; (3) integration with downstream systems, e.g., autonomous driving, requires low latency and high reliability, which current detectors struggle to guarantee simultaneously. Future directions could emphasize efficient multi-scale attention, diffusion-based refinement, and real-world robustness testing beyond standard benchmarks.

2.4.3 Image Generation

Image generation [162, 798, 307] refers to using algorithms to create or synthesize new images from scratch or by modifying existing ones. These algorithms learn from a large dataset in a generative manner, since they have to approximate the distribution of the dataset and output new, realistic, and detailed images from generated samples. As a result, the learning algorithms can capture patterns within the dataset and combine them to create new feature representations for data generation. It is a prominent field in computer vision and artificial intelligence, in which models generate realistic or stylized images that may resemble photographs, art, or imaginative scenes given specific input conditions. The primary goal is to generate images that appear authentic or fulfill a specified criterion, often mimicking real-world data distributions. This section primarily focuses on technique development for generating super-resolution images with enhanced quality and for translating images across different styles.

Image Super-Resolution

Image super-resolution (SR) [71, 419, 361] is an upsampling technique in computer vision that increases the number of pixels for a high-resolution (HR) image based on a given low-resolution (LR) input, as depicted in Figure 2.8. The goal is to analyze the contextual information surrounding observed pixels in an LR image and identify interpolation patterns to fill the unknown details between upsampled pixels in an HR image. SR is valuable in medical imaging, satellite imaging, and surveillance, where high-quality images are crucial for analysis and interpretation. In recent years, we have witnessed remarkable progress in image super-resolution using deep learning techniques. Initially, CNNs brought a breakthrough in SR by automating feature extraction from data and serving as discriminative models that directly learn LR-to-HR mapping functions, using two different architectures: interpolation-based and sub-pixel-based methods [686]. Interpolation-based approaches seek to identify patterns within a local region and interpolate a new pixel as a weighted sum of the surrounding feature vectors. Some SR models [152, 321, 322, 405, 273] combined static upsample interpolation with a stack of convolutions to decompose a learnable interpolation into static upsampling and learnable adjustment. In addition, transposed convolution (a.k.a deconvolution [794, 793]) is another learnable upsample method for SR models [466, 153, 624, 172, 227, 230, 188]. On the contrary, the sub-pixel-based methods [572] learn to encode the local patterns into feature channels and decode the local information from feature channels for new pixel reconstruction. Due to its end-to-end upsampling nature, the sub-pixel method is widely used in SR models [4, 812, 833, 344, 648]. Later, generative CNNs were developed to solve the SR problem. Starting with Generative Adversarial Networks (GANs), CNNs integrated adversarial training to directly learn to generate realistic textures and finer details for HR images. SRGAN [352] integrated sub-pixel modules into GAN-based architectures for better visual quality. Other variants [681, 8, 57, 378] combined different learnable upsampling methods with GANs and designed new architectures to improve SRGAN. Most recently, Diffusion models have been used as alternatives in generative CNNs for SR image generation [561, 548, 370, 787, 684].

Instead of using each pixel as input, ViTs take patches (a set of pixels) as input and introduce an attention mechanism to capture long-range dependencies in images, allowing models to focus on both global and local features for super-resolution. As a discriminative model, TTSR [746] treated LR images and reference images as the query and key, respectively, and adopted a hard-attention module to match relevant

Figure 2.8: Image Super-Resolution learns to compensate for the missing details for better visualization. The image example is cited from SR3 [561]

textures between the reference and low-resolution images using a transformer. IPT [67] proposed a multi-task architecture based on ViTs pre-trained on large datasets, including super-resolution. DPT [843] proposed a dual-path transformer to capture spatial and frequency domain information for improved SR. ESRT [450] introduced a light ViT for SR with less computational complexity. Some research focuses on improving the computational efficiency of local window-based self-attention. SwinIR [403] adapted the Swin Transformer for image SR by focusing on local and global feature extraction. Uformer [685] proposed a LeWin Transformer block that performed

non-overlapping window-based local self-attention to reduce computational cost on high-resolution feature maps. HAT [92] integrated channel attention into window-based self-attention to activate more input pixels for better reconstruction. Additionally, generative ViTs adopted GANs [358, 490, 536, 190] to connect a ViT generator and a ViT discriminator for adversary learning to generate high-quality SR images.

Recently, hybrid methods that combine the strengths of CNNs and transformers have been applied to image SR. The CNNs with attention mechanisms learn to select the most relevant pixels in a non-local scope to reconstruct more coherent interpolations. SelNet [111] proposed a Selection-Unit to attend to relevant information and filter out the irrelevant for the latter SR reconstruction. RCAN [832] integrated channel attention to highlight significant feature channels for HR image generation. DNLN [72] integrated a deformable non-local attention with a multi-branch weighted feature fusion for better capture pixel-wise correlations, long-range self-similarities, and multi-scale semantic combination. In addition, the hybrid GANs that integrate convolution with ViTs [456, 625, 841, 27] have demonstrated outstanding performance on SR image generation. SWCGAN [631] adopted GAN architectures to connect a local attention-enhanced CNN generator and a pure Swin Transformer [441] based discriminator for adversarial training. Similarly, SRTransGAN [24] designed a generator that combined PVTv2 [674] with sub-pixel methods and adopted a pure ViT as the discriminator.

Evolution trends and practical deployment gaps

Image SR has progressed from CNN-based interpolation to GAN-driven perceptual enhancement (SRGAN, ESRGAN) and Transformer-based methods (SwinIR, HAT) that model long-range texture dependencies. CNNs are efficient and easy to deploy, but often produce overly smooth results; GANs improve visual realism but introduce training instability and artifacts. Transformers capture global patterns better but are computationally intensive, especially for high-resolution outputs. Hybrid designs (RCAN, DNLN) combine channel attention with non-local mechanisms to balance quality and speed. Deployment gaps are pronounced in real-time video SR and mobile applications, where memory and latency constraints are severe. Lightweight SR networks sacrifice some reconstruction fidelity for speed, highlighting the trade-off between perceptual quality, inference time, and hardware compatibility.

Image-to-Image Translation

Image-to-image translation (I2I) is a generative task in computer vision that involves learning a mapping between a source distribution and a target distribution for domain

transfer, as depicted in Figure 2.9. It can be applied to various applications, such as style transfer. To train generative models, deep models, i.e., CNNs, ViTs, and hybrid architectures, have evolved from GANs to Diffusion models. CNN-based GANs were initially the primary approaches for I2I tasks due to their ability to automatically capture local patterns from images with convolutional operations and build a mapping between different domains with adversary training: Pix2Pix [290], CycleGAN [860], UNIT [431], DiscoGAN [323] and MUNIT [282]. Alternatively, CNN-based Diffusion models [492, 165] adopted diverse samples through iterative refinement, simplifying the training process of I2I tasks other than the GANs: SDEdit [471], Palette [560], BBDM [362], and DiffI2I [704]. ViTs have enabled global dependency modeling for I2I tasks that involve complex, non-local transformations, such as large structural changes or style modifications. GAN-based ViTs designed both generators and discriminators with pure visual transformers and integrated attention mechanisms into the adversary training: TransGAN [310], Styleformer [699], StyTr2 [141], StyleSwin [799], ViTGAN [210]. Additionally, there are a few Diffusion-based ViTs that find a mapping between two different Markov chains [705]: IPT [67], StyleDiffusion [687], DiT [512, 840], MDT [191], unidiffuser [25], DVT [221], DIFFIT [233]. Moreover, some researchers try to integrate the strengths of CNNs and ViTs into I2I tasks. The hybrid generative models equipped with GANs and Diffusion models learn to map the source and target domains: UVCGAN [625], TransWeather [636], and ITTR [848].

Evolution trends and practical deployment gaps

Image generation has progressed from CNN-based GANs to Transformer-driven generative models and diffusion paradigms. GANs offer high-quality synthesis but still suffer from training instability; diffusion models provide stable training and fine-grained control but are computationally intensive. ViT-based generators (StyleSwin, TransGAN) enable long-range coherence but require substantial memory. Hybrid approaches balance local texture details with global structure. Key deployment challenges are: (1) inference latency for diffusion models that require iterative sampling, limiting real-time use; (2) resource demands on high-resolution generation that are prohibitive on edge devices; (3) generalization to unseen styles/domains that models often overfit to training data distribution. Practical deployment in fields like entertainment necessitates model compression, distillation to lightweight generators, and efficient sampling algorithms to bridge the gap between quality and speed.

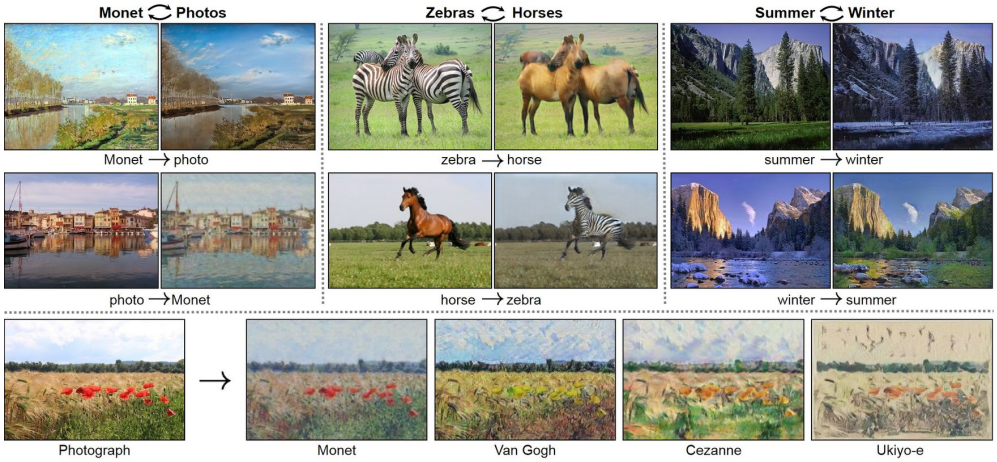


Figure 2.9: Image-to-Image translation learns a mapping between a source distribution and a target distribution for domain transfer. The image example is cited from [860].

2.4.4 Universal Models

The development of universal models paves the way for Artificial General Intelligence (AGI). It is no longer merely a tool for achieving higher accuracy on benchmarks for specific tasks but has become the foundational engine for constructing generalist visual AI. This pursuit of universality, while computationally intensive, is unlocking new frontiers in model capability for important but challenging domains.

Large-Scale Visual Foundation Models

These large-scale visual foundation models have proven effective in specific challenging domains. A compelling research thread involves adapting SAM variants to maneuver the challenges of scarce annotation in weakly-supervised settings. WS-SAM [235] leveraged SAM by using sparse annotations as prompts to generate pseudo segmentation masks of concealed objects. In order to improve mask quality, WS-SAM integrated a multi-scale feature grouping technique with other strategies to tackle the intrinsic similarity challenge. Furthermore, SEE [236] extended the WS-SAM to exploit the SAM for camouflaged object segmentation with incompletely annotated data. Most recently, SCALER [239] presented a unified collaborative framework that jointly optimizes a mean-teacher segmenter and a learnable SAM2 for label-deficient concealed object segmentation. This evolution, from WS-SAM (utilization) through SEE (distillation) to SCALER (collaboration), has demonstrated a clear development trajectory in which a general-purpose foundation model (SAM/SAM2) can be progressively and

deeply integrated to solve domain-specific problems with increasing sophistication and performance.

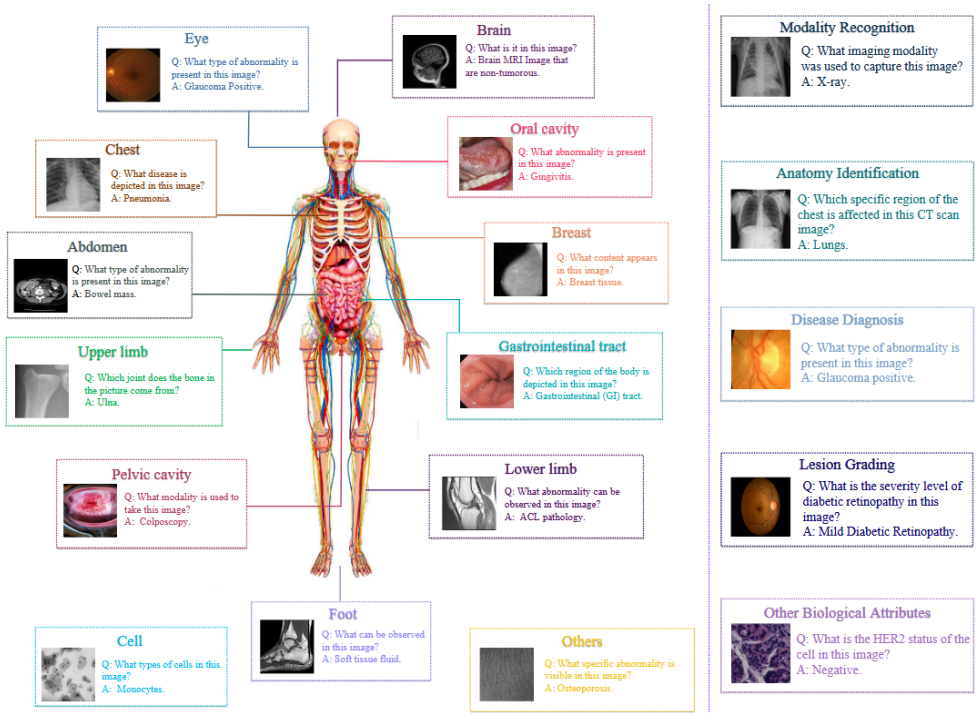


Figure 2.10: Left: Overview of OmniMedVQA dataset[274], which covers the majority of radiology modalities and anatomical regions of the human body. Right: Illustrations of samples from five different medical question types.

Multiple Modal Models

The transition from single-modal to multi-modal foundation models marks a critical phase in the evolution of General Artificial Intelligence. By transcending the confines of research benchmarks, these models demonstrate their paramount importance as versatile core technologies across diverse domains. Their ability to process and synthesize heterogeneous data streams is fundamentally reshaping methodologies and enabling novel solutions to complex real-world problems.

Vision-Language Models

The practical applications of Visual-Language Models (VLMs) are extensively demon-

strated by their zero-shot prediction capabilities across a wide spectrum of visual recognition tasks. The foundational model CLIP established the paradigm, achieving remarkable zero-shot accuracy on diverse image classification datasets, ranging from general benchmarks like ImageNet [138] to fine-grained tasks such as Stanford Cars [339] and Oxford-IIIT Pets [509]. Beyond image-level recognition, subsequent VLMs have been successfully adapted for more complex dense prediction tasks. For instance, models like GLIP [376] and DetCLIP [760] enabled zero-shot open-vocabulary object detection by aligning region proposals with text descriptions. In contrast, methods such as GroupViT [732] and SegCLIP [451] extend this capability to semantic segmentation by matching text embeddings with pixel or segment features. Furthermore, VLMs facilitate zero-shot image-text retrieval and have been evaluated on action recognition datasets such as UCF101 [589], thereby demonstrating their broad utility as general-purpose visual recognition engines.

Emotion Recognition Models

Building on the foundational cross-modal alignment principles of visual-language models, the multi-modal paradigm has been profoundly applied in the field of human–machine interaction. Significant advancements have been made in one important cross-modal application, i.e., emotional analysis, that integrates diverse data sources, such as speech, text, and images. Early explorations focused on novel fusion mechanisms, such as tensor decomposition [669], and brain-like hierarchical perception networks [866] to integrate visual, acoustic, and textual cues. The pursuit of robustness has become a central theme, leading to architectures such as the Robust Adversarial Fusion Transformer (RAFT) [668] for sentiment analysis and to frameworks employing random modality dropout [828] or contrastive learning [709] to mitigate negative information and handle incomplete data. Recent trends emphasize deeper contextual and generative modeling, as evidenced by works on contextual interaction based analysis [666] and methods that integrate audio-visual text generation with contrastive learning [667]. The state-of-the-art further incorporates diffusion models [868] and end-to-end systems [864], while practical deployment is explored through client-server recognition systems [867]. This concentrated research effort, as summarized in key reviews [865], demonstrates how multi-modal foundation models are specifically engineered to tackle the complexities of human emotion analysis across diverse and challenging real-world contexts.

Evolution trends and practical deployment gaps

The rise of large-scale foundation models (DINOv2, SAM) and multi-modal systems (CLIP, VLMs) marks a shift from task-specific models to general-purpose visual intelligence. These models excel in zero-shot transfer and open-vocabulary understanding but require substantial computational and data resources. While VLMs enable semantic grounding across vision and language, their size limits on-device deployment. Trade-offs involve scale vs. accessibility (since foundation models are powerful but require cloud-level resources), and generalization vs. specialization (since they may underperform in niche fields without fine-tuning). Deployment gaps include: (1) memory and latency, which make running billion-parameter models on edge devices infeasible; (2) privacy and data governance, which raise concerns in sensitive domains when centralized training is used; (3) energy consumption, which conflicts with sustainable AI goals. Future efforts could prioritize modular, scalable architectures, federated learning frameworks, and efficient cross-modal distillation to make universal models practical for widespread use.

2.4.5 Medical Data Analysis

Beyond generic vision tasks, deep learning has also revolutionized domain-specific applications, such as the medical field. Medical data analysis is one of the most important Interdisciplinary tasks, since it enables automated decision-making processes, such as disease diagnostics, treatment planning, and prognosis. Formally speaking, each decision-making process is a multi-modal task that fuses information from visual data (e.g., pathology staining and radiological images) and sequential data (e.g., clinical details, biomarkers, protein sequences, and medical history) to support quantitative analysis. Precise and thorough panoptic segmentation of a medical image is key to scene understanding, as it facilitates the representation of each entity separately. It is the first step for multi-modal tasks, but has attracted less attention than instance segmentation, and has recently gained increasing attention in medical image analysis. Moreover, a central challenge in designing medical AI systems lies in the architecture of multi-modal models, which must integrate diverse data sources to extract richer and more robust feature representations.

Image Segmentation

CNN-based segmentation for medical image analysis mainly focused on feature fusion from different branches due to the failure to represent concepts efficiently [140, 725]. Liu et al. [424] combined an auxiliary semantic segmentation branch with the instance

branch to integrate global and local features to enhance nuclei segmentation. PFFNet [423] incorporated the global-level semantic with local-level instance features and proposed a mask quality sub-branch to align the semantic and instance predictions. Panoptic Lintention Network [465] provided a comprehensive understanding of the environment of individuals with visual impairments in healthcare. Additionally, ViT-based instance segmentation models have been the primary options due to their attention mechanisms. Cell-DETR [521] adapted from the DETR for cell instance segmentation and achieved state-of-the-art performance from microscopy imagery. Medical Transformer [635] explored transformer-based solutions and proposed a Gated Axial-Attention model by introducing an additional control mechanism in the self-attention module. Verma et al. [644] explored self-training strategies for panoptic segmentation and implied their potential enhancement for medical image analysis.

Medical Vision-Language Models

Medical Vision-Language Models (MVLMs) [797, 275, 274] are advanced AI systems that combine visual and textual medical data to perform complex tasks such as medical report generation (MRG) and Medical Visual Question Answering (MVQA), as shown in Figure 2.10. These models have to understand medical data and generate corresponding textual descriptions. Beyond merely image captioning, MRG has evolved to generate descriptive reports from input visual and textual data automatically. Initially, MRG models combined different branches to process visual and textual data separately: CNN-RNN architectures [313, 395, 778, 766] and CNN-Transformer architectures [593, 491]. Later, knowledge graphs [425, 831] were adopted to represent heterogeneous data across modalities. Most recently, Transformer-based models [482, 617, 619, 819, 171] have been the most popular solutions. MVQA [272] can answer questions about medical images based on understanding both the visual content of the image and the natural language question. Likewise, MVQA models initially combined two branches: CNN-RNN [504] and CNN-Transformer [646, 280, 401], and later adopted transformer-based architectures [489, 786].

Evolution trends and practical deployment gaps

Medical vision has evolved from CNN-based segmentation to Transformer-enhanced models and multi-modal systems. These advances improve accuracy and enable report generation, but they face unique domain constraints: limited annotated data, high reliability requirements, and multi-modal integration. Hybrid CNN–ViT models balance local anatomical detail and global context, yet they struggle with interpretability and

robustness to rare pathologies. Deployment gaps are pronounced: (1) clinical validation that models rarely undergo rigorous real-world trials; (2) integration with hospital IT systems, where data privacy and interoperability are major hurdles; (3) computational resources that high-resolution 3D medical images demand GPU clusters, limiting point-of-care use. Moving forward, the field ought to invest in explainable hybrid architectures, federated learning for data privacy, and lightweight, domain-specific compression to transition from research prototypes to trusted clinical tools.

2.5 Discussions and Conclusion

Given the rapid evolution of hardware performance, deep learning models are and will be a hot topic in computer vision due to their competitive performance and tremendous potential compared with other machine learning approaches. To harness the power of deep learning models, as summarized in this survey, many methods have been proposed in recent years. These methods perform well across a range of visual tasks, including image classification, object detection, and image generation. Nevertheless, the potential of deep learning models for computer vision has not yet been fully explored, and both challenges and prospects remain.

2.5.1 Discussions

Thanks to automatic data fitting and generalized inference, deep learning models have transformed many computer vision tasks. Despite their successes, they still have considerable room for improvement, especially in more complex real-world scenarios. As deep learning evolves, it is crucial to incorporate novel ideas into deep learning models, especially in interpretability, data efficiency, multi-modal understanding, and scalability, making it a powerful tool for future AI applications. To address the aforementioned challenges, future research should focus on the following directions.

Improving Data Efficiency and Annotation Scarcity

The performance of all deep vision models, from CNNs to large-scale foundation models, remains intrinsically linked to the volume and quality of annotated data—a major bottleneck in data-scarce domains. While self-supervised learning (SSL) paradigms, exemplified by DINO, have made significant strides in learning transferable representations from uncurated data, they often demand vast computational resources. The integration of generative models, particularly Diffusion Models, offers a transforma-

tive complementary path. Future research could move beyond using these models in isolation and develop synergistic frameworks in which diffusion models generate high-fidelity, task-aware synthetic data or feature augmentations to robustify the training of discriminative CNNs, ViTs, and hybrids. Furthermore, unifying the principles of SSL with the generative capabilities of diffusion processes could lead to next-generation pre-training paradigms that learn compressed, semantically rich world models from minimal supervision, drastically reducing annotation dependency for downstream adaptation.

Enhancing Model Interpretability and Trustworthiness

As architectures grow in complexity—from attention maps in ViTs to dynamic pathways in hybrid or Mamba-based models—their opacity deepens, eroding trust in safety-critical applications like medical diagnostics. Merely visualizing attention weights or activation maps provides post-hoc insight but seldom explains causal reasoning. A promising frontier lies in embedding explainability directly into the model's architecture and training objective. Deep Unfolding Networks (DUNs) provide a foundational blueprint in which each network module corresponds to a step in an interpretable optimization algorithm. Extending this principle, future architectures could be designed to have inherently interpretable latent spaces, or by integrating differentiable probabilistic graphical models that enforce causal structure. For large foundation models, developing rigorous concept-based explanation methods that align internal representations with human-understandable concepts is crucial. Ultimately, trustworthiness will be achieved not by adding external explanation tools, but by co-designing models whose decision-making processes are transparent and aligned with domain-knowledge constraints by construction.

Achieving Real-World Robustness and Generalization

A significant gap persists between benchmark performance and reliability in dynamic, open-world environments, where models have to face unforeseen distribution shifts, adversarial perturbations, and novel object configurations. Architectural hybrids of CNNs and ViTs aim to balance local invariance and global context for better generalization, but they lack formal robustness guarantees. The strategic incorporation of multi-modal knowledge, particularly from Vision-Language Models (VLMs), offers a powerful regularizing force. Semantic constraints from language can ground visual predictions and help reject nonsensical adversarial outputs. Future research could focus on developing tightly coupled, efficient vision-language systems where a compact, task-specific vision model (CNN/ViT/hybrid) is continuously guided by the semantic and

commonsense knowledge of a (potentially frozen) large VLM. Furthermore, simulation-to-real and test-time adaptation techniques must be advanced to enable models to continuously self-correct and adapt to novel environments, leveraging principles derived from foundational pre-training, moving from static models to adaptive visual agents.

Exploring Design Paradigm for Efficiency

The pursuit of efficiency must evolve from applying isolated compression techniques post-hoc to embracing a holistic efficiency-by-design paradigm across the entire stack, encompassing everything from training dynamics to inference latency. While Neural Architecture Search (NAS) and compression techniques such as pruning and quantization are essential, they must be deeply integrated and hardware-aware. For hybrid CNN-Transformer or State Space Model (SSM) based models, this means developing NAS frameworks that can co-optimize architectural partitioning (e.g., deciding where to use local convolution or global attention) alongside compression strategies and deployment target constraints (e.g., memory bandwidth, parallel computing). The next generation of efficient models will likely be dynamic and conditional, activating different computational subgraphs based on input complexity. Research could therefore develop unified frameworks that jointly search for optimal architectures, their optimal compression profiles, and their dynamic inference policies, yielding a portfolio-optimal spectrum of models for diverse real-world constraints rather than a single point solution.

From Static Deployment to Adaptive Edge Ecosystems

The final translational challenge is deploying capable models in heterogeneous, resource-constrained, and dynamic edge environments. Current deployment typically involves distilling a large model into a smaller, static network, which becomes brittle under changing conditions. The future lies in adaptive edge ecosystems. The adaptation requires models, particularly efficient hybrids, to be endowed with context-aware adaptive inference capabilities, i.e., dynamically adjusting their depth, width, or even architectural pathways in response to real-time metrics such as battery level, computational load, or perceived task difficulty. Furthermore, the rise of foundation models presents a new opportunity: edge devices could run a highly compressed, task-specific model while occasionally querying a cloud-based foundational model for complex, out-of-distribution inferences, creating a collaborative hierarchy. Advancing federated learning frameworks to support the continuous, privacy-preserving fine-tuning of these adaptive models across devices will be critical. The ultimate goal is to transition from deploying static models to cultivating intelligent, collaborative, and self-improving

visual perception networks at the edge.

Ethical Considerations for Deep Learning Models

Beyond technological advances, the widespread deployment of deep learning systems in real-world applications necessitates consideration of ethical issues, since decisions influenced by AI can have profound consequences for individuals and communities in high-stakes environments. Key concerns include algorithmic bias, which can perpetuate or even amplify societal inequalities; privacy risks arising from data collection and model inference; and a lack of transparency that undermines trust and accountability. As large-scale foundation models become increasingly integrated into daily workflows, ethical considerations collectively implicate their deployment in sensitive domains such as medical diagnostics and surveillance, thereby making addressing ethical dimensions a fundamental prerequisite for responsible use. International organizations [4] assemble guidelines for responsible use that developers and users of deep-learning frameworks should take notice of and include in their line of work. Promising research directions for mitigating ethical risks include developing fairness-aware algorithms, adopting privacy-preserving learning paradigms, e.g., federated learning and differential privacy. In addition, and very importantly, is the endeavor to enhance model interpretability to ensure that decisions are both reliable and accountable. As researchers continue to push the boundaries of model performance, we suggest that parallel efforts would be directed toward understanding and addressing the ethical dimensions of deploying these systems in practice. We encourage the community to engage in interdisciplinary discussions that bridge technical robustness and ethical responsibility, ensuring that deep learning technologies evolve into positive and productive forces that serve society equitably and responsibly.

2.5.2 Conclusion

In conclusion, this survey has articulated the trajectory of visual recognition from dedicated CNNs to general-purpose Transformer-based foundation models, highlighting the convergent trend toward hybrid architectures that leverage complementary inductive biases. We have contextualized these core advancements within a broader package of efficiency, distillation, and generative enhancement techniques. The path forward, however, is not merely one of incremental architectural refinement. The next leap in computer vision will be driven by integrative research that crosses the boundaries

[4] 1. UNESCO. 2. AI4People.

among data efficiency and generation, performance and interpretability, and isolated model design and adaptive system deployment. By addressing the interconnected challenges of data scarcity, opacity, fragility, inefficiency, and rigidity through a holistic lens, the field can advance toward building robust, trustworthy, and universal visual intelligence that operates reliably in the complexity of the real world. The convergence of architectural innovation, learning paradigms, and systems-aware co-design will define the next era of discovery in computer vision.