Artificial Intelligence in Computer Vision: Enabling Technologies and New Capabilities

László Árvai[1]

[1]University of Miskolc, Hungary, Institute of Informatics
laszlo.arvai@uni-miskolc.hu

Abstract. Machine vision has been reshaped over the past decade by the adoption of artificial intelligence, and by deep neural networks that learn visual features from data rather than relying on rules designed by hand. This paper reviews the principal AI technologies now employed in computer vision and examines how each has extended the capability of vision systems. It traces the shift from hand-crafted feature engineering to learned representations and surveys the main architecture families in current use: convolutional networks, detection and segmentation architectures, vision transformers, structured landmark representations, and generative and self-supervised approaches. It then examines the cross-cutting enablers that decide whether a method is practically usable: transfer learning, benchmark infrastructure, and deployment technologies including open exchange formats and model compression. Attention is paid to the consequences for higher education, where pretrained models running on commodity hardware have made practical machine vision teachable and reproducible in ordinary classrooms. The review concludes that the decisive contribution of AI has been less any single architecture than the emergence of a reusable ecosystem of pretrained models, portable runtimes, and shared benchmarks, and that the principal open challenges are now data cost, domain shift, interpretability, and the widening gap between frontier-scale models and deployable systems.

Keywords: Machine Vision, Deep Learning, Convolutional Neural Networks, Vision Transformers, Transfer Learning, Engineering Education.

1 Introduction

Machine vision extracts actionable information from images. It underpins industrial inspection, robotic guidance, medical image analysis, autonomous navigation and human-computer interaction, and in each domain the practical question is the same: given a stream of pixels, what decision should the system make?

For most of the field's history the answer was constructed by hand: an engineer identified the visual cues distinguishing the cases of interest and wrote an explicit sequence of operations converting them into a decision. That approach still works where lighting, background and object presentation are controlled, but degrades sharply outside them.

The adoption of deep neural networks changed this qualitatively. Instead of specifying the features, the engineer specifies the task and supplies examples; the network

© The Author(s) 2026
S. Árvai-Homolya et al. (eds.), *Proceedings of the 2nd Innovation in Engineering Education Conference (IEE 2026)*, Atlantis Highlights in Social Sciences, Education and Humanities 58,
https://doi.org/10.2991/978-94-6239-823-8_3

discovers the features. This inversion has propagated into every application domain, including the engineering curricula in which discipline is taught. This review surveys the AI technologies presently used in machine vision, asking which are in use and what problem each solves, what capabilities they have opened, and what obstacles remain. The scope of the survey and the criteria used to select the literature are set out in the next section.

2 Scope and Review Method

This paper is a narrative scoping review rather than a systematic one. Its aim is not to enumerate every published vision architecture, but to identify the AI technologies that have changed what a practitioner can deploy, and what a university course can teach, on ordinary hardware.

Literature was identified through IEEE Xplore, the ACM Digital Library, Scopus and arXiv, covering 2012 to 2025, with earlier landmark work such as SIFT [1] and convolutional networks trained by backpropagation [3] retained for lineage. Search terms combined "computer vision" or "machine vision" with the architecture families treated below — convolutional networks, object detection, semantic segmentation, vision transformers, foundation models and transfer learning — and with the practical filters that organise the review: deployment, edge inference, model interchange and engineering education.

A method was included if it became a default choice in applied systems, introduced a capability that earlier pipelines could not provide, or bears directly on teaching machine vision in higher education. Application-domain catalogues, unpublished code-only artefacts and papers whose contribution is a dataset rather than a method were excluded, except where the dataset defined a shared benchmark used throughout the field (ImageNet [2], COCO [21]). Selection was iterative: the seed set comprised the canonical results that shifted practice, after which citing and cited work was followed until additional papers no longer changed the deployment trade-offs reported in Table 1. That table is organised by labelled-data need and by where a model can run, not by chronological completeness or peak benchmark score.

Relative to existing surveys of deep learning for vision, which typically organise material by task [30] or by architecture family [31], this review is organised by enabling technology and by the conditions under which a method is usable outside a well-funded laboratory. The educational consequences are treated as part of that usability argument, not as a separate application area.

3 From Hand-Crafted Features to Learned Representations

The classical pipeline was a sequence of deterministic image-processing stages: grey-scale conversion, noise suppression, thresholding, morphological clean-up, and geo-

metric measurement of the surviving regions. Recognition systems added explicit feature extraction, with descriptors such as SIFT [1] converting local image structure into vectors a conventional classifier could separate.

These methods have virtues that are easily forgotten: they are fast, need no training data, are deterministic, and their failure modes are interpretable, since an engineer can inspect the intermediate images and see where the pipeline went wrong. Their limitation is brittleness. A threshold tuned for one lighting condition fails under another; a pipeline that works against a plain background collapse against a cluttered one. Each new operating condition demanded another hand-written rule, and the rules interacted badly.

The break came from three developments arriving together: data, in the form of the ImageNet database [2]; compute, through general-purpose GPU programming; and the architecture itself. Although convolutional networks trained by backpropagation had been demonstrated a decade earlier [3], it was the ImageNet result of Krizhevsky et al. [4] in 2012 that made the case decisively, cutting error far below the best hand-engineered systems. Residual connections [5] later enabled networks of over a hundred layers to train stably.

The conceptual shift matters more than any individual result. Early layers converge on generic structure - oriented edges, color transitions, simple textures - closely resembling what decades of hand-designed descriptors encoded explicitly but now discovered automatically. The engineer no longer designs the representation, but curates the data and specifies the objective. Fig. 1 summarises this progression.

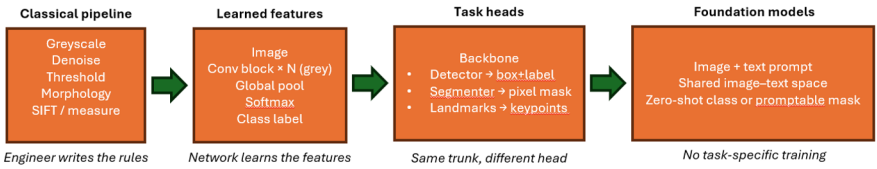


Fig. 1. Evolution of the machine-vision pipeline, from a hand-specified sequence of image-processing stages to learned convolutional features, task-specific heads, and foundation models that transfer across tasks.

4 Core AI Technologies in Machine Vision

4.1 Convolutional Networks and Task Architectures

The convolutional neural network (CNN) remains the workhorse of applied machine vision. A CNN learns small spatial filters applied across the whole image, so a feature detected in one location is detected in any location. This weight sharing reduces the parameter count relative to a fully connected network and builds in tolerance to translation; pooling extends that tolerance and enlarges the receptive field.

Their primary advantage is end-to-end trainability: no hand-crafted feature extraction is required, and the network learns to distinguish classes from raw pixels given enough labelled examples. Their limitations are equally practical. CNNs need large,

labelled datasets that are expensive to annotate; they are sensitive to changes in background, lighting and camera geometry not represented in training; and larger variants need GPU acceleration for real-time inference. Depthwise-separable convolutions [6] and compound scaling [7] address the cost, and are what make CNN inference feasible on ordinary laptops.

Classification assigns one label to a whole image. Most tasks need more: where is the object, how many are there, and what is its exact extent? Each additional degree of localization costs more computing and more annotation.

Object detection returns a bounding box, label and confidence per instance. Two-stage detectors propose regions then classify them, as in Faster R-CNN [8]; one-stage detectors predict all boxes in a single pass, and YOLO [9] and SSD [10] showed that a modest accuracy penalty buys a large speed gain, so they dominate real-time and edge deployment. Semantic segmentation assigns a class to every pixel: fully convolutional networks [11] made this practical, and the encoder-decoder design with skip connections introduced by U-Net [12] proved dense prediction could be learned from remarkably few annotated images, making it the default wherever labels are scarce. Mask R-CNN [13] combines both, returning a mask per object. Fig. 2 contrasts these task heads.

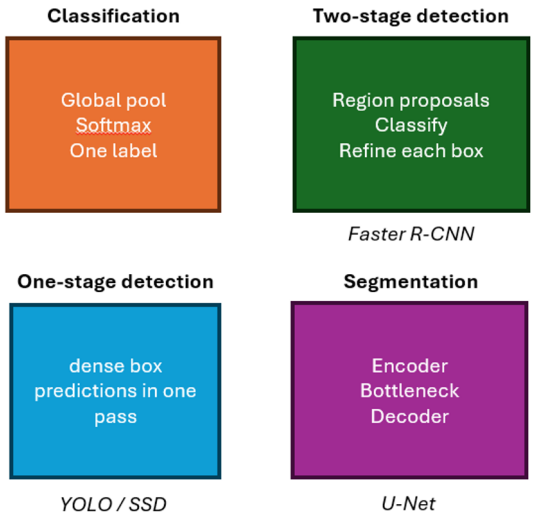


Fig. 2. Task architectures built on a shared visual backbone: image-level classification, two-stage and one-stage object detection, and encoder-decoder semantic segmentation.

4.2 Attention and Vision Transformers

The transformer [14] replaces convolution's local receptive field with self-attention, in which every element attends directly to every other. Applied to vision by treating image

patches as tokens, the Vision Transformer [15] showed that convolution is not a necessary ingredient: given enough pretraining data, a nearly architecture-agnostic model matches or exceeds convolutional networks.

The trade-off is instructive: self-attention gives global context immediately but discards the assumption that nearby pixels are related, a useful prior when data is limited. This is why transformers are data-hungry and why hybrid designs reintroducing locality and hierarchy [16] perform well at moderate scale.

4.3 Structured Intermediate Representations

Not every problem is best solved by mapping pixels directly to a label. An influential alternative inserts an intermediate representation: a first model extracts a compact structured description of the scene, typically a set of named keypoints, and a small second model reasons over it. Frameworks such as MediaPipe [17] package this pattern as ready-to-run pipelines running at real-time rates on ordinary CPU hardware. The engineering argument is strong. Because the intermediate features encode geometric relationships between parts rather than raw pixel values, they are substantially robust to variation in lighting, background, colour and moderate changes in scale and orientation. A classifier trained on such coordinates generalises to new environments more readily than one trained on raw images, and needs far less data, because the expensive perception work is already done. The cost is a loss of generality: information the representation discards is unavailable downstream. Where a task is inherently temporal, recurrent models such as LSTM [18] extend these representations across time.

Generative models learn the distribution of images rather than a mapping from image to label. Diffusion models [19] can synthesise photorealistic imagery conditioned on text; for machine vision their most direct use is augmenting training sets for cases that are rare or expensive to capture. Self-supervised learning attacks the annotation bottleneck from the other direction, defining objectives that need no labels at all: methods such as DINOv2 [20] yield features that transfer with very little labelled data, which is the regime most practical projects occupy.

5 Cross-Cutting Enablers

5.1 Transfer Learning and Benchmarks

Training from random initialisation requires resources few projects possess, and in practice almost nobody does it. The dominant workflow takes a model already trained on a large general dataset, retains its backbone, and retrains only a small task-specific head. This works because early-layer features are largely universal: edges and textures are as relevant to circuit boards as to natural photographs, and only the final decision needs domain-specific data.

Transfer learning is arguably the single most important practical development in applied machine vision, because it converts a problem requiring millions of images into one requiring hundreds. Fig. 3 shows the resulting train-adapt-export-deploy loop.

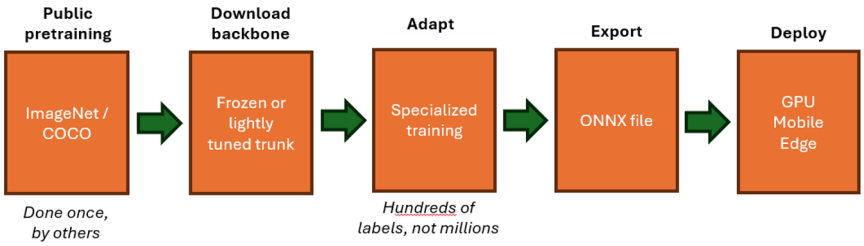


Fig. 3. The dominant applied workflow: a backbone pretrained on a large public benchmark is adapted with a small task-specific head, exported to a portable format such as ONNX, and executed on commodity or edge hardware.

Progress has been driven as much by datasets as by architectures. ImageNet [2] and COCO [21] did not merely supply training data; they defined shared tasks and metrics against which competing methods could be compared, and that comparability is what allowed the field to accumulate results rather than merely produce them.

Where benchmarks are absent the pattern is less healthy. In many applied sub-domains every published study relies on a custom dataset assembled by its own authors, with few participants and controlled acquisition conditions. This limits reproducibility, prevents comparison across studies, and yields models that generalise poorly: reported accuracy is evidence that a method can work, not a measurement of how well it will work elsewhere.

5.2 Deployment and Portability

A model that runs only in the notebook that trained it has limited engineering value. Open exchange formats, principally ONNX [22], decouple a trained model from the framework that produced it, so one model file can be executed by different runtimes on different hardware without rewriting.

Table 1. Architecture families and their practical trade-offs. Data requirement refers to labelled data needed by the practitioner, not by the original pretraining.

Family	Representative	Typical task	Data need	Deployment
Classical pipeline	Threshold, SIFT	Inspection	None	Controlled scenes only
CNN backbone	ResNet, EfficientNet	Classification	High	CPU feasible, GPU preferred
Lightweight CNN	MobileNet	Classification	High	CPU, mobile, embedded
One-stage detector	YOLO, SSD	Detection	High	Real-time video, edge
Encoder-decoder	U-Net	Segmentation	Moderate	Modest hardware
Vision transformer	ViT, Swin	Classification	Very high	GPU, or pretrained transfer
Landmark pipeline	MediaPipe	Keypoints, pose	Low	Real-time on CPU
Foundation model	CLIP, SAM	Zero-shot tasks	Pretrained	Cloud API or desktop GPU

Quantisation to integer arithmetic [23] typically reduces model size roughly fourfold with modest accuracy loss. Table 1 summarises the principal families against the criteria that govern this choice.

6 New Capabilities and Educational Impact

6.1 Zero-Shot and Multimodal Recognition

The most striking recent capability is recognition without task-specific training. By learning a shared embedding space for images and text, CLIP [24] classifies images into categories specified in natural language at inference time, with no examples of those categories ever labelled. Segment Anything [25] applies the same logic to segmentation, producing masks in response to a prompt rather than a fixed class list. This dissolves an assumption so basic it was rarely stated: that the category set must be fixed before training begins. Vision-language models extend this further, accepting an image with a question in ordinary language, so that a plausible baseline can be obtained by writing a prompt rather than collecting a dataset.

These developments changed not only what vision systems do but how quickly the field advances. When a backbone can be downloaded rather than trained, a portable runtime executes it anywhere, and shared benchmarks make claims comparable, the cost of testing an idea falls by orders of magnitude. The ecosystem, not the algorithm, is the accelerant.

6.2 Machine Vision in Higher Education

The consequences for teaching run in two directions. As subject matter, machine vision has become dramatically more teachable. A decade ago a practical course required GPU hardware, long training runs and awkward datasets, and a student project could exhaust its time budget before producing a working system. Today a pretrained detector runs on a laptop CPU in milliseconds, a landmark pipeline tracks in real time from a built-in webcam, and a complete train-export-deploy cycle fits within a single session; structured curricular frameworks have begun to appear [26]. The reproducibility benefit is significant: a notebook-based laboratory exercise executing offline on commodity hardware can be distributed to a whole cohort and produces the same result for each of them.

As a tool, computer vision is being applied within higher education itself - to adaptive learning, automated assessment, accessibility and campus administration - and recent reviews survey this uptake and the ethical questions it raises [27]. Purpose-built visualisation tools have also been shown to improve comprehension of otherwise abstract technical material [28]. Both directions rest on the same foundation: pretrained models plus commodity hardware.

7 Discussion: Open Challenges

Data cost remains the binding constraint. Transfer learning and self-supervision have reduced the labelled data requirement by orders of magnitude but have not eliminated it, and for any task whose classes do not resemble existing pretraining corpora, annotation remains the dominant project cost.

Domain shift is poorly handled. Models degrade, sometimes sharply, when deployed on data differing systematically from their training distribution, and the failure is rarely announced by the model's own confidence, which tends to remain high.

Interpretability remains limited. A classical pipeline could be debugged by inspecting intermediate images; a deep network cannot, and attribution methods such as Grad-CAM [29] indicate where a network attended without explaining why. In regulated domains this is a deployment barrier; in education it is a pedagogical hazard, since students can obtain a working result without understanding the mechanism - a risk that a course built only around downloading pretrained models actively amplifies.

Finally, the gap between frontier and deployable models is widening rather than closing. Foundation and vision-language models advance rapidly but are large, while systems that must run in real time on modest hardware still depend on compact convolutional models established years ago. Techniques for bridging the two, principally distillation and quantisation, currently lag the growth in frontier model size.

8 Conclusion

Artificial intelligence changed machine vision less by supplying a better algorithm than by replacing one methodology with another. The defining shift was from features designed by engineers to features learned from data, and every subsequent development is a consequence of that inversion. Surveying these technologies together suggests their individual merits are less decisive than their composition. What makes machine vision practical today is not that convolution or attention is powerful in isolation, but that a pretrained backbone can be downloaded, adapted with a modest dataset, exported to a portable format and executed on commodity hardware. That ecosystem is the field's real achievement of the last decade, and it is what has made the discipline teachable in ordinary higher-education settings rather than only in well-funded laboratories.

The open problems are correspondingly systemic. Data cost, domain shift, interpretability and the distance between frontier-scale and deployable models are not defects of any single architecture but properties of the methodology as a whole. Progress on them is likely to come from better shared infrastructure, better benchmarks and better techniques for compressing what large models know into small ones, rather than from architectural novelty alone.

References

1. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60(2), 91-110 (2004)

2. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: CVPR 2009, pp. 248-255. IEEE (2009)
3. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE 86(11), 2278-2324 (1998)
4. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25, pp. 1097-1105. Curran Associates (2012)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR 2016, pp. 770-778. IEEE (2016)
6. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861 (2017)
7. Tan, M., Le, Q.V.: EfficientNet: rethinking model scaling for convolutional neural networks. In: ICML 2019, PMLR, vol. 97, pp. 6105-6114 (2019)
8. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems 28, pp. 91-99. Curran Associates (2015)
9. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: CVPR 2016, pp. 779-788. IEEE (2016)
10. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: SSD: single shot multibox detector. In: Leibe, B., et al. (eds.) ECCV 2016, LNCS, vol. 9905, pp. 21-37. Springer, Cham (2016)
11. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR 2015, pp. 3431-3440. IEEE (2015)
12. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., et al. (eds.) MICCAI 2015, LNCS, vol. 9351, pp. 234-241. Springer, Cham (2015)
13. He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask R-CNN. In: ICCV 2017, pp. 2961-2969. IEEE (2017)
14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems 30, pp. 5998-6008. Curran Associates (2017)
15. Dosovitskiy, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. In: ICLR 2021 (2021)
16. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin Transformer: hierarchical vision transformer using shifted windows. In: ICCV 2021, pp. 10012-10022. IEEE (2021)
17. Lugaresi, C., et al.: MediaPipe: a framework for building perception pipelines. arXiv:1906.08172 (2019)
18. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. 9(8), 1735-1780 (1997)
19. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Advances in Neural Information Processing Systems 33, pp. 6840-6851. Curran Associates (2020)
20. Oquab, M., et al.: DINOv2: learning robust visual features without supervision. Trans. Mach. Learn. Res. (2024)
21. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollar, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: Fleet, D., et al. (eds.) ECCV 2014, LNCS, vol. 8693, pp. 740-755. Springer, Cham (2014)
22. ONNX: Open Neural Network Exchange, https://onnx.ai, last accessed 2026/07/17

23. Jacob, B., et al.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: CVPR 2018, pp. 2704-2713. IEEE (2018)
24. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: ICML 2021, PMLR, vol. 139, pp. 8748-8763 (2021)
25. Kirillov, A., et al.: Segment anything. In: ICCV 2023, pp. 3992-4003. IEEE (2023)
26. Jeon, I.-S., Kang, S.J., Kang, S.-J.: A staged framework for computer vision education: integrating AI, data science, and computational thinking. Appl. Sci. 14(21), 9792 (2024)
27. Oqaidi, K., Aouhassi, S., Mansouri, K.: How does computer vision reshape higher education? A comprehensive review. J. Eng. Educ. Transform. 38(4) (2025)
28. Cuevas, C., Cortes, C., Garcia, N.: Empowering computer vision in higher education: a novel tool for enhancing video coding comprehension. Comput. Appl. Eng. Educ. 33(1) (2025)
29. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: visual explanations from deep networks via gradient-based localization. In: ICCV 2017, pp. 618-626. IEEE (2017)
30. Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E.: Deep learning for computer vision: a brief review. Comput. Intell. Neurosci. 2018, 7068349 (2018)
31. Khan, S., Naseer, M., Hayat, M., Zamir, S.W., Khan, F.S., Shah, M.: Transformers in vision: a survey. ACM Comput. Surv. 54(10s), 1-41 (2022)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

