International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

State-of-the-Art Computer Vision and ML Operations: An Integrated Analysis of Software and Hardware Architectures

Jyothiprakash Reddy Thukivakam, Sayanna Chandula
NIT, Warangal, India

Date of Submission: 20-03-2025 Date of Acceptance: 30-03-2025

ABSTRACT: This comprehensive article examines the current state of computer vision and machine learning operations across software frameworks and hardware platforms. It explores the evolving landscape of CV/ML deployments, analyzing the integration challenges and optimization strategies in modern systems. This article evaluates prominent software libraries and frameworks with primary usecases. It encompasses hardware acceleration techniques, dedicated neural processors, and vision processing units, offering insights into their performance characteristics and power efficiency trade-offs. It highlights emerging trends in hardware-software co-design, standardization efforts, and future directions in the field. This article contributes to the understanding of optimal CV/ML deployment strategies while addressing the growing demands of edge computing and real-time processing requirements.

Keywords: Computer Vision Acceleration, Inference, Edge AI Computing, Neural Processing Units, Hardware-Software Optimization.

I. INTRODUCTION & CURRENT LANDSCAPE

Computer vision is a field of artificial intelligence (AI) that enables computers to "see" and interpret the visual world. In essence, it aims to replicate the capabilities of human vision, allowing machines to extract meaningful information from images and videos. It relies on various techniques, including but not limited to:

- **Image processing:** Manipulating and enhancing images
- **Machine learning:** Training algorithms to recognize patterns in visual inputs.
- **Deep learning:** Using neural networks to analyze complex visual information.

Computer vision has a wide range of applications, including but not limited to:

- **Object detection:** Identifying specific objects with bounding boxes in images or videos.
- **Semantic Segmentation:** Similar to object detection but identifies objects on a pixel level granularity instead of bounding boxes and classify objects with precise boundaries
- **Image classification:** Categorizing images based on their content.
- **Facial recognition:** Identifying individuals from their facial features.
- **Autonomous vehicles:** Enabling cars to "see" and navigate their surroundings.
- **Medical imaging:** Assisting in the diagnosis of diseases.

The computer vision (CV) and machine learning (ML) operations landscape has experienced unprecedented growth, reshaping industries from manufacturing to healthcare. According to Cognitive Market Research, the global computer vision market is expected to grow at a CAGR of 16.2% during the forecast period of 2023-2030, with the market size projected to reach $67.3 billion by 2030. This growth is primarily

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

driven by increasing adoption in autonomous vehicles, robotics, and smart manufacturing sectors. The market dynamics have shifted significantly, with enterprise-grade CV solutions becoming more accessible and deployable across diverse computing environments, from cloud infrastructure to edge devices [1].

II. SOFTWARE FRAMEWORKS & LIBRARIES

A computer vision library provides a high-level Application Programming Interface(API) abstraction for key image/video processing tasks with pre-build functions, algorithms and utilities designed. A good computer vision library abstracts away the low-level complexities of hardware-specific optimization, memory management and internal resource management. CV APIs allow developers to integrate CV functionalities into their applications without extensive coding. This section attempts to discuss a few widely used open source CV libraries/frameworks available.

2.1 OpenCV

Open Computer Vision Library (OpenCV) is the most widely used computer vision library, with a strong community and rich ecosystem. Initially developed by Intel in 2000, OpenCV now supports a wide range of applications from simple image processing to more advanced and complex computer vision applications like face recognition, object detection tasks. OpenCV is written in C++ and has bindings for Python, JAVA and MATLAB. OpenCV is open source project under Apache-2.0 license, and hosted on github: https://github.com/opencv/opencv

2.2 YOLO

You Only Look Once, or YOLO is among the fastest computer vision tools you can opt for in 2024. Developed by Joseph Redmon and Ali Farhadi in 2016, it was specifically made for real-time object detection. YOLO redefined object detection by predicting bounding boxes and class capabilities directly from the full image in single evaluation. YOLO object detection can process images in real-time. It's known for its speed and accuracy making it ideal for applications like surveillance and autonomous driving. YOLOv12 is released in Feb, 2025 with attention centric real time object detectors.

YOLOv12 is open source under AGPL-3.0 license and its hosted on the github: https://github.com/sunsmarterjie/yolov12

2.3 TensorFlow(TF)

TensorFlow is Google's open source library, which is a general purpose machine learning library, it offers robust computer vision tasks through various APIs and tools such as TensorFlowHub & TensorFlow Lite(light weight TF optimized for running on embedded/mobile). TensorFlow is one of the easy to use computer vision tools and allows development of computer vision related machine learning models for tasks like facial recognition, image classification, object detection etc,. It supports various languages like Python, C++ & JAVA.
TensorFlow is open source project under Apache-2.0 license hosted on github: https://github.com/tensorflow/tensorflow and TFlite https://github.com/tensorflow/tflite-micro

2.4 CV-CUDA

Compute Unified Device Architecture (CUDA) is a parallel computing platform and API model developed by NVIDIA. It enables developers to use GPUs to make computationally intensive applications faster. The toolkit includes NVIDIA's Performance Primitive library that provides GPU accelerated Image, Video processing and signal processing functions including computer vision. CUDA architectures are useful for a wide range of applications like face recognition, 3D graphics rendering and image processing functions. It supports languages like C, C++, Python and is compatible with most operating systems. CV-CUDA is an open source project for building cloud safe CV applications:
https://github.com/CVCUDA/CV-CUDA

2.5 OpenVINO

Open Visual Inference and Neural Network Optimization is a set of comprehensive computer vision tools that are useful for developing applications emulating human vision. Developed by Intel, it is an open source and cross platform toolkit. OpenVINO toolkit comes with models for several tasks like object detection, face recognition, movement recognition etc.
OpenVINO™ is open source toolkit (Apache-2.0 license)
https://github.com/openvinotoolkit/openvino

2.6 BoofCV

BoofCV is a Java based computer vision software library which is lightweight and targets

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

embedded vision real time applications. It's written entirely in Java, making it platform-independent and easy to integrate with other Java applications. It is open source and cross platform capable. BoofCV is optimized for speed, allowing for real-time processing of images and video.
BoofCV is an open source library released under Apache-2.0 license for academic and commercial use. https://github.com/lessthanoptimal/BoofCV

2.7 DeepFace

Deepface is an open source computer vision library for face recognition with deep learning. This library offers an easy way to perform face recognition based computer vision with Python. It simplifies the process of working with facial recognition and analysis, making it accessible to a wider range of developers and applications. DeepFace is an open source under MIT license
https://github.com/serengil/deepface

2.8 Torch Vision

TorchVision is a computer vision toolkit companion library that is part of the PyTorch ecosystem, designed specifically for handling computer vision tasks. It provides utility functions for image transformation, model architectures and required support to load the datasets for machine learning tasks. Primary use cases supported are: Object detection, semantic segmentation and image generation models. TorchVision is open source project under BSD-3.0-Clause license https://github.com/pytorch/vision

2.9 Mediapipe

MediaPipe provides a pre-build, realtime AI, ML solutions for various computer vision tasks with cross functional support focusing on performance targeting mobile and embedded systems. Its open source project under Apache-2.0 license terms, https://github.com/google-ai-edge/mediapipe

The relationship between CV and ML is deeply intertwined, with ML serving as crucial engine driving advancements in CV. Popular ML frameworks built with compatible computer vision frameworks like TensorFlow, PytTorch, Sickit-Learn etc., with training and deployment focus.

With so many available tools, the necessity of interoperability has become an issue which is addressed by an open source AI model format called ONNX (Open Neural Network Exchange). It allows you to train a model in one framework (e.g., PyTorch, TensorFlow) and deploy in another like OpenVINO etc.,

Software Library	Pros	Cons
OpenCV	* Open Source and standard tool for image processing * Large community support	- Not easy to use - Steep learning curve
YOLO	* Fast, accurate object detection with bounding box * Real time capable	- Limited community support - Not effective in small object detection
Tensor Flow	* Scalable and Compatible with many languages * Powerful features and performance packed.	- Resource hungry as it is generic machine learning model inferencing toolkit
CV-CUDA	* Fast, effective and high performance video analysis * NPP library with many image and signal processing primitives	- High Power consumption
Open VINO	* Supports multiple deep learning kits * Good hardware support (Intel)	- Limited sample applications - Not optimized for non-intel platforms
BoofCV	* User friendly interface * Multiple language support * Light weight	- Needs JVM to run - Lower performance compared to C++ implementations of openCV

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

DeepFace	* Lightweight and easy to install * Optimized to perform real time on device inference.	- No cloud API support
Torch Vision	* Efficient data processing for image transform operations * Strong community support	- Large model sizes
MediaPipe	*Realtime CV applications for AR/Pose and Hands tracking * Cross Platform support	- Prebuilts only, modifications are not possible

Table 1 Summary popular CV frameworks/tools

III. HARDWARE ACCELERATION PLATFORMS

Need for Acceleration

Pixel's values are nothing but a 2D array of numbers which can be represented as a matrix. CV libraries perform convolution on images for feature map generation. Total number of multiply-accumulates required to compute is determined by the input image & kernel size.

$$\text{FeatureMap OutW} = 1 + (\text{InW} - \text{KW} + 2 * \text{P})/\text{S}$$
$$\text{FeatureMap OutH} = 1 + (\text{InH} - \text{KH} + 2 * \text{P})/\text{S}$$
$$\text{Total MACs} = \text{OutW} * \text{OutH} * \text{MACs Per Knl}$$

OutW: OutputWidth, InW: InputWidth, KW: KernelWidth, KH: KernelHeight, P: Padding and S: Stride

For example, a 352x288 Grayscale Input, with 8x8 kernel grid, with stride 1 (i.e. non overlapping), no padding requires 345x281x64 = 6,204,480 Total number of MACs for one input frame and repetitive for each frame in the input stream. It shows the arithmetic pressure on the hardware. This will increase exponentially for complex networks with higher input sizes with more channels. Memory bandwidth is the speed at which data can be transferred between memory and processing blocks. High compute power capable units require higher memory bandwidth for performing faster processing w/o memory stalls. It totally makes sense to have dedicated processing blocks with primary MAC focused for achieving best throughput.

3.1 Advanced CPUs

Modern CPUs enabled with multi-core and multi level cache support are increasingly optimized for computer vision workloads, leveraging specialized instructions for vectorization and dedicated accelerators to improve performance, efficiency, and parallel processing.

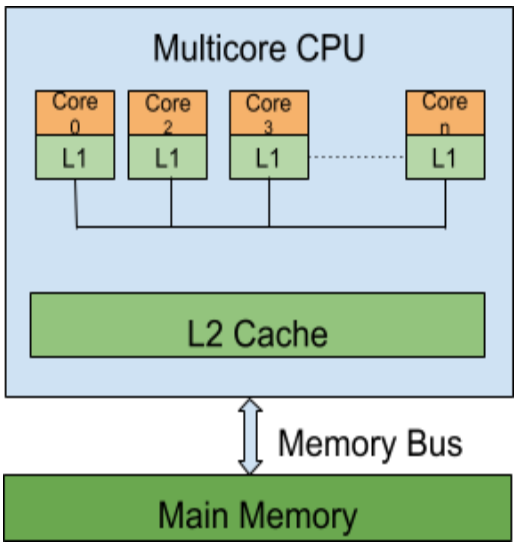


Fig. 1 Multicore CPU architecture

However CPUs are designed for general purpose computing, they are optimized for sequential processing but not for repetitive large scale tasks and have low memory bandwidth compared to GPUs, hence not optimal for CV/ML workloads.

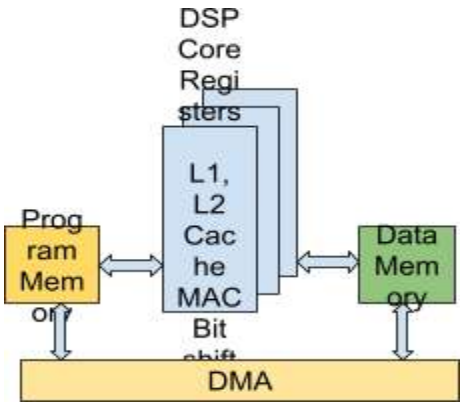
3.2 DSP (Digital Signal Processors)

Digital signal processors accelerate computer vision algorithms through specialized instruction sets. DSP has special functional blocks for performing signal processing operations:

- MAC - For filtering, Fast Fourier Transform (FFT) and convolution
- SIMD - Parallel processing of multiple data points.
- Circular buffers & barrel shifter - For faster FFT computation, FIR filtering

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

DSPs provided real-time processing for audio, video, and signal processing applications, leveraging their optimization for fixed-point data formats. DSP architectures advanced to fasten the processing and increase throughput. TI's C66x[5] uses VLIW with Von Neumann architecture i.e. with dedicated Program and Data memory, and has two hardware MAC units and allows instruction execution in parallel with raw peak performance of 32 counts of 16 bit x 16 bit fixed point MAC in 1 clock cycle. So with a 1GHz operating clock, C66x can perform up to 32 GMAC/s.



Fi.g 2 Typical DSP architecture

There are optimized algorithms available for computing convolution in a faster way. One such is Winograd's algorithm which is derived from Winograd's minimum filtering algorithm. It reduces the number of multiplications by a large factor. This made it possible to deploy CNNs on multi-core high end DSPs with floating point support. There are optimized CNN implementations on high end DSPs available[6]. Flex Logix's InferX High Performance IP provides scalable AI inference with integrated DSP and compiler support, achieving high accuracy across PyTorch, ONNX, and TFLite formats, while significantly reducing cost (1/10th) and power (40%) [13].

DSPs are heavily optimized for sequential signal processing operations which is not a scaling friendly architecture which makes it not an ideal choice for inferencing and ML training. Although researchers who want to explore inferencing with multicore DSPs, lack of sophisticated software support makes developers effort increase by large amounts to achieve meaningful/comparable throughput as contemporary processing units achieve.

3.3 GPU (Graphics Processing Unit)

GPU is designed to accelerate the rendering of images and videos on a device. Initially developed for video games and computer-aided design (CAD), GPUs have since evolved to manage a variety of parallel processing tasks beyond graphics rendering.

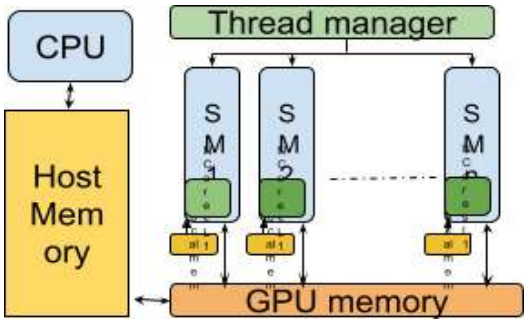


Fig. 3 GPU Architecture

Host CPU is responsible for loading the GPU memory with appropriate data which should definitely be optimized for maximum throughput and effective GPU resource utilization. Thread management should be taken care to configure individual SM and load balances for complex compute tasks in a way that yields maximum throughput.

As shown in Fig. 3, GPU has many Streaming multiprocessors(SM) with internal memory connected in parallel and shared across a GPU memory. Each SM has many cores with shared memory.

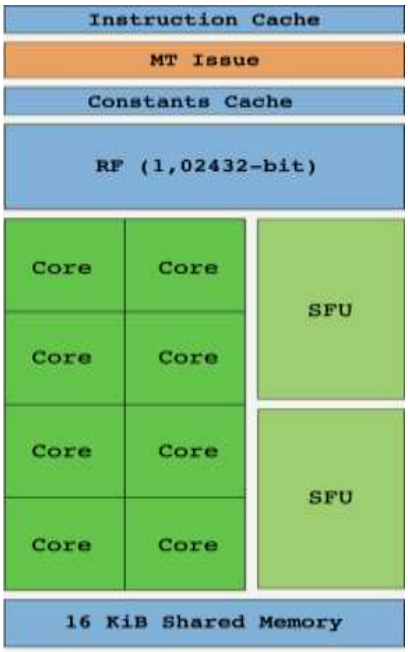


Fig. 4: NVIDIA's "Tesla" SM architecture[12]

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

Each SM as shown in Fig. 4, has multiple cores capable of handling one integer/float32 instruction per clock. Special function units(SFU) are for complex arithmetic calculations like inverse square root, trigonometric or exp etc., these can run in one clock cycle. Register Files (RF) is where thread states are saved. While 'Frankenstein' methods offer increased precision, they come at the cost of reduced throughput due to their computational expense. Though GPUs have many cores, developers need not worry too much about programming. All heavy lifting is taken care of by GPU frameworks like CUDA or OpenCL that leverage the full potential of GPUs, resulting in a performance boost for complex computations. GPU-accelerated platforms have emerged as the preferred solution for training deep learning models and high-performance inference in data center environments and AI research laboratories. These specialized architectures leverage massive parallel processing capabilities to handle the computational demands of large-scale vision processing workloads.

Here are the main things that standout in GPUs that made it suitable for CV and ML tasks:

- GPUs have thousands of smaller, efficient cores allowing them to handle tasks simultaneously. For example NVIDIA's high end GPUs have 10000+ CUDA cores.
- Higher memory capacity in terms of GBs
- Higher memory bandwidth. NVIDIA's high end GPUs have bandwidth ~2 TB/s
- Availability of matured software framework tool kits like CUDA made effective uses of thousands of cores utilized for maximum throughput.
- Scalability: Easy to add more GPU for increased parallel processing.

In data center deployments, modern GPU architectures demonstrate exceptional performance for training complex vision models. Research indicates that current-generation GPUs can process training batches for large convolutional neural networks significantly faster than alternative computing platforms, making them indispensable for AI research laboratories developing cutting-edge vision algorithms[2]. Their architecture is specifically optimized for the matrix multiplication operations that dominate deep learning workloads, enabling efficient scaling for model development and experimentation.

For high-performance inference scenarios requiring maximum throughput, GPU platforms offer substantial advantages. Analysis shows that in data center environments, GPU-accelerated inference serves multiple simultaneous processing streams while maintaining consistent latency profiles, a critical requirement for production AI services [3]. This capability makes them particularly valuable for research institutions processing large datasets or running computationally intensive simulations.

The flexibility of GPU architectures provides significant benefits for AI laboratories, where algorithmic experimentation requires adaptable computing resources. Studies demonstrate that the programmable nature of GPUs allows researchers to implement and test novel vision processing approaches without hardware constraints, accelerating the pace of innovation[2]. The extensive ecosystem of development tools and libraries further enhances their utility in research environments.

While power consumption remains higher than specialized accelerators, the performance advantages make GPUs the dominant choice for environments where training speed and inference throughput take precedence over energy efficiency. Research indicates that modern GPU clusters in data centers implement sophisticated workload distribution techniques to optimize resource utilization across multiple processing nodes, enabling efficient scaling for large-scale vision applications[3].

3.4 TPU (Tensor Processing Units)

Tensor Processing Units (TPUs) represent a specialized class of AI accelerators that have emerged as ideal solutions for both cloud-based AI workloads and energy-efficient edge inference which is the process of running a trained neural network on an edge device with limited memory and processing power. These purpose-built processors are specifically designed to accelerate tensor operations fundamental to computer vision and deep learning tasks.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

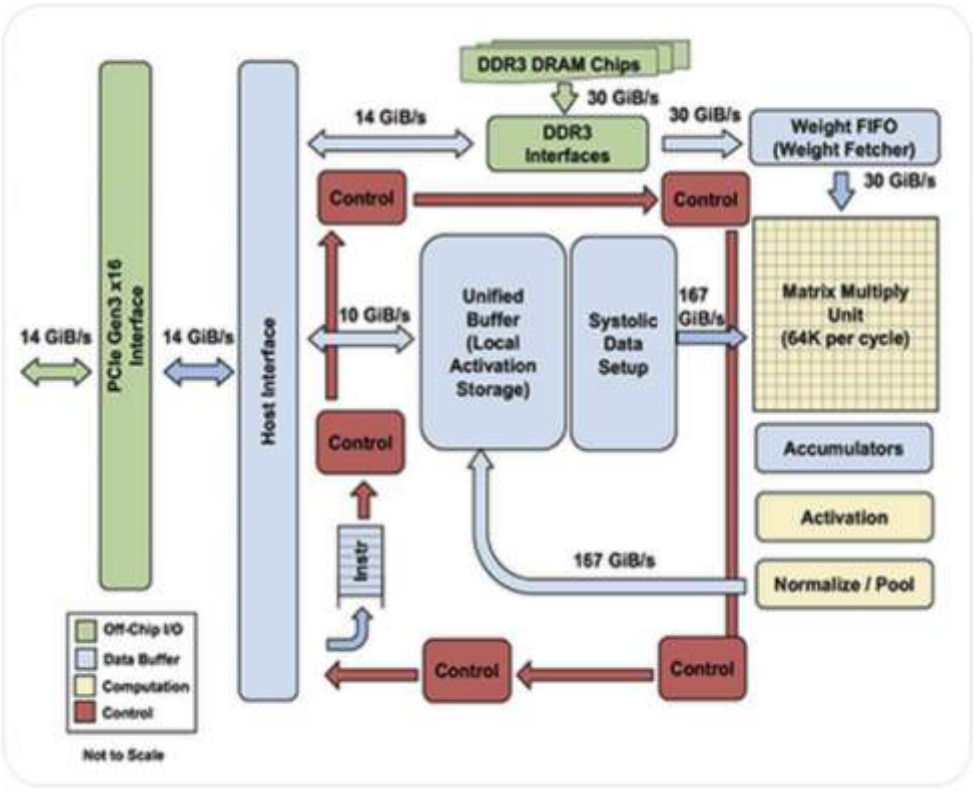


Fig. 5: TPU Block diagram[7]

A neural network inference is done via a sequence of matrix multiplications and additions which loosely represents an artificial neuron as in Fig. 6

$$\text{output} = Act\left(\sum_{1}^{3} xn \,.\, wn \;+\; b\right)$$

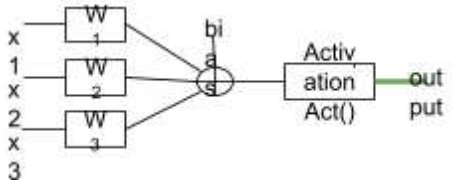


Fig. 6 Fundamental neuron in inference model

Number of MACs depends on the network configuration and input size & channel, kernel size and layers. For example, CNN1 has 89 network layers and has 100M weights[7] shows intensity of MACs needed for every single inference.

TPUs are designed with a primary fixed function goal to optimize convolutional inference tasks in general. TPUs are designed to run thousands of matrix operations in a single clock cycle which needs both software and hardware techniques. TPUs encapsulates neural network essentials into a **systolic array** which is parallel computing architecture for homogeneous networks of tightly coupled nodes[8]. This enables TPUs to use less memory operations to compute the results and during execution intermediate results are directly passed between functional units without additional memory access, significantly reducing the power consumption and increasing throughput. As a result, CISC based TPUs provide 83X better ratio compared to contemporary CPUs and almost 29X better ratio than GPUs[7]. TPUs are less widely available and produced only by Google.

Another key advantage of TPU platforms lies in their seamless integration within the software ecosystem. Studies highlight that robust development tools and highly optimized software frameworks substantially lower implementation complexity, facilitating quicker deployment of vision applications across both cloud and edge environments[3]. This integration capability, coupled with the performance and efficiency benefits, establishes TPUs as highly versatile accelerators capable of addressing a wide range of needs—from cloud data centers to resource-constrained edge devices.

3.5 FPGA-Based Acceleration

Field-Programmable Gate Arrays (FPGAs) offer a distinctive approach to computer vision acceleration, providing customizable, low-latency processing capabilities ideally suited for real-time embedded applications. These reconfigurable platforms enable hardware-level optimization for specific vision algorithms, delivering deterministic performance characteristics essential for time-sensitive operations.

In embedded vision systems, FPGAs demonstrate exceptional latency characteristics that make them particularly valuable for applications requiring immediate response times. Research indicates that FPGA implementations achieve consistent processing latencies with minimal variation, a critical advantage for real-time applications such as industrial automation, robotics, and safety-critical systems [2]. This deterministic performance ensures predictable system behavior even under varying operational conditions.

The reconfigurable nature of FPGAs provides significant advantages for specialized vision tasks in embedded environments. Analysis shows that custom hardware implementations of vision algorithms can achieve substantial throughput improvements for specific operations while maintaining power consumption within strict embedded system constraints [3]. This customizability allows developers to precisely allocate hardware resources to critical processing paths, optimizing both performance and energy efficiency.

For applications with specialized processing needs, FPGAs provide a level of flexibility that fixed-function accelerators cannot. Research shows that vision pipelines can be efficiently structured in hardware, enabling high-throughput processing of streaming data with minimal buffering. This architecture is especially advantageous for embedded systems handling continuous video streams, where buffer memory is constrained.

FPGAs excel in embedded systems with diverse I/O requirements due to their ability to integrate multiple interfaces and processing elements within a single platform. Studies show that entire vision processing pipelines—from sensor input to algorithmic computation and control output—can be implemented on a single FPGA, minimizing system complexity and enhancing reliability in embedded applications. This high level of integration makes FPGAs an ideal solution for compact devices that require autonomous vision processing.

3.6 Neural Processing Unit Architecture

Neural Processing Unit(NPU) architecture is dedicated to energy efficient CNN acceleration. Just like DSPs, NPU architecture uses von Neumann architecture, which separates the memory and the processing units. Research proved that utilizing smaller bit-precision is enough for inference with low power consumption. This led hardware architects to investigate energy efficient NPU architectures with diverse software-hardware schemes for inference. In fact, hardware architects and researchers designed and proposed relatively simple neural network acceleration over 30 years ago. Mauduit et al. designed a single neuronal unit that integrates only memories and ALU for simple operations such as MUL, ADD, ACC, while non-linear activation is performed by off-chip components[8]. Viredaz et al. introduced 40 × 40 systolic array architecture for neural network acceleration[9].

The primary design objectives for NPUs are high-throughput parallel processing and energy efficiency. Since CNNs require large amounts of data for training and inference, memory bandwidth becomes a crucial concern. So NPUs rely on data reuse and skipping useless computations techniques to resolve large off-/on-chip memory bandwidth. Processing Elements (PEs), the fundamental computational units, are arranged in a grid-like structure with loose interconnections, mirroring the neural network of brain cells. During inference, partial sums and data are passed through the Processing Elements (PEs).

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

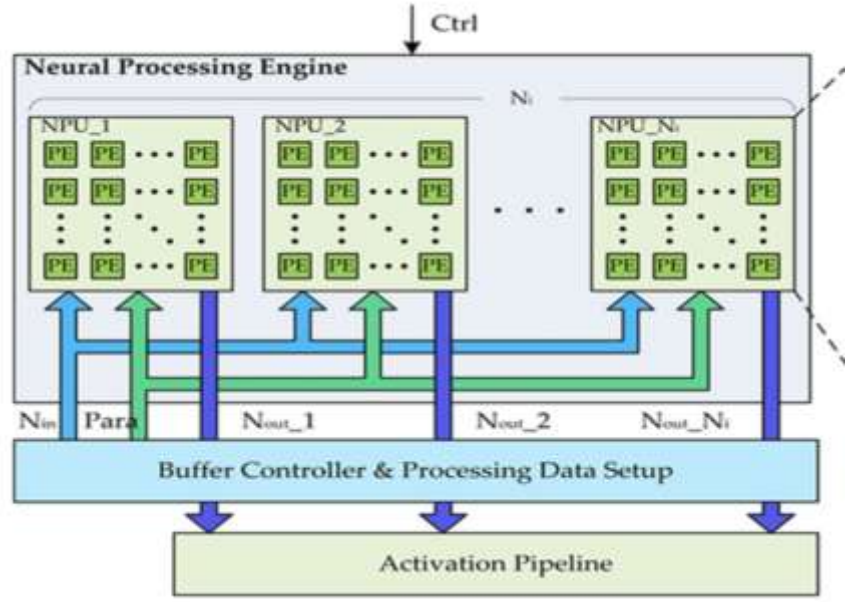


Fig. 7 Block diagram of NPU[11]

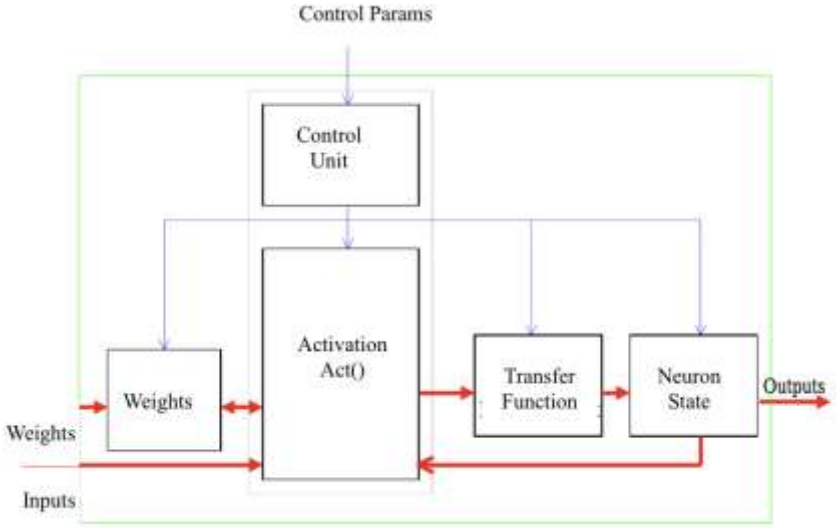


Fig. 8 Processing Element(PE) architecture

Each PE connected in the grid may have, activation block that performs MAC of weight & inputs which are always on the neuro-chip. Other blocks like neuron state, weights, transfer blocks may be on chip or prepared by the host computer. The data flow between the control blocks is controlled by the control unit that is always on chip. Control parameters are controlled by the host computer[10]. Despite their complex design and higher manufacturing costs, NPUs are becoming increasingly available, produced by major companies like Intel, Huawei, and Samsung.The landscape of neural processing units has witnessed significant advancements in both architecture and efficiency. NPUs have the advantage of having higher peak performance and hence handles more complex and diverse neural networks. They offer a substantial latency reduction of up to 64% compared to traditional GPUs, though they are still slower than TPUs[3]. The integration of specialized memory hierarchies has enabled remarkable improvements in data access patterns, with some designs achieving memory bandwidth utilization rates above 85%. Benchmark analysis demonstrates that current-generation NPUs can maintain processing efficiency above 92% for quantized neural networks while reducing power

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

consumption by approximately 56% compared to floating-point implementations [2].

HW Acceleration	Pros	Cons	Power Efficiency
CPUs	* Highly flexible * Easy to program	- Power Hungry - Poor performance for complex CV tasks	Low
GPUs	* Broad software support (CUDA, TensorFlow, Pytorch) * Excellent Parallel Processing	- Power Hungry - Expensive	Moderate
TPUs	* High Throughput for AI work loads * Optimized for Tensor operations	- Limited flexibility - Primarily tied to Google Cloud	High
FPGA	* Low latency *Reconfigurable for specific tasks	- Higher development complexity - Steep learning curve	High
NPUs	* Optimized for Neural Networks * Low power consumption	- Less flexible for general purpose tasks - Limited to AI workloads	High

Table 2: HW platform's comparison

3.7 Emerging Trends in HW support

The future of AI inference hardware accelerators is poised for significant advancements, driven by the need for efficient, and scalable architectures. Several emerging trends are shaping the next-gen hardware accelerator. Few such trends:

Heterogeneous computing: Frankenstein architecture with multiple processors such as CPUs, GPUs, NPUs etc., Challenge is to get an innovative software framework to manage this complex system.

3D chip stacking: Its emerging technology also known as 3D integration which involves stacking multiple layers of semiconductor chips(vertical scaling) to create a single, densely packed unit. This may reduce data transfers b/w functional blocks resulting higher throughput and low latency

Quantum computing: It is an emerging technology compared to above, but ongoing research suggest that quantum accelerators could eventually complement traditional hardwares and perform high complex computation beyond reach of current digital systems

IV. BENCHMARKING CONSIDERATIONS

Evaluating computer vision (CV) systems involves measuring accuracy, speed, efficiency, and scalability on various hardware and datasets. Consistent benchmarking requires standardized tools and use-case-specific datasets. Popular benchmarking datasets include:

International Journal of Advances in Engineering and Management (IJAEM)
Volume 7, Issue 03 Mar. 2025, pp: 928-939 www.ijaem.net ISSN: 2395-5252

Core Focus	DataSet	Use Case
Image Classification	ImageNet (Large scale dataset with 1000 object categories)	Image Classification Object Recognition
	MNIST (Handwritten digit images 0 - 9)	Basis classification task
Object Detection	COCO (80 object categories with bounding boxes)	Object Detection Segmentation
	KITTI	Autonomous Driving Object Detection
Semantic Segmentation	Cityscapes (Urban scenes)	Self driving Urban scene understanding
	Pascal VOC (pixel wise segmentation masks)	Semantic Segmentation Deep learning benchmarking
Face Recognition	VGGFace2 (face images with variations in pose, age and ethnicity)	Face recognition Verification
	CelebA (large scale face images with attribute annotations)	Face Recognition Attribute analysis
Video Segmentation	CamVid (Street driving videos with per frame labels)	Temporal Segmentation Video based CV

Table 3. Popular benchmarking dataset

Datasets are essential for evaluating and comparing the performance of algorithms and models across various tasks. These datasets provide standardized data for training, validation, and testing, enabling researchers and developers to measure progress and identify areas for improvement.

The open-source Computer Vision community actively maintains all models and benchmarking data, ensuring they reflect the current state of the art[4].

V. FUTURE TRENDS AND CHALLENGES

Architecture & Efficiency:

- Optimized architectures, hybrid cloud-edge deployments, and AI accelerators are driving cost reduction and improved processing speed.

Frameworks & Optimization:

- Modern frameworks enable model compression, faster video processing, and efficient training through techniques like transfer learning and hyperparameter optimization.

Standardization:

- Standardized deployments, testing, and data preprocessing improve system integration, maintainability, accuracy, and performance evaluation.

Security & Ethics:

- Privacy and security risks, especially with sensitive data like facial recognition, necessitate robust encryption and access controls.
- Bias mitigation through auditing, diverse datasets, and ethical frameworks is crucial for fair and responsible AI.

Multimodal Learning:

- Vision-LLMs integrate visual and linguistic understanding, enabling zero-shot learning and improved human-computer interaction.

3D Vision & Spatial AI:

- 3D vision and Spatial AI expand capabilities beyond 2D, enabling advanced applications in AR, robotics, and digital twins.

Emerging Technologies:

- Neuromorphic, quantum-inspired, and bio-inspired computing promise significant advancements in energy efficiency and performance.

VI. CONCLUSION

The comprehensive analysis of state-of-the-art computer vision and ML operations support demonstrates the rapid evolution and significant advancements in both software frameworks and hardware acceleration technologies. Through a detailed analysis of standardized benchmarking methodologies, cross-platform performance evaluations, and real-world implementations, this article highlights the critical role of optimized hardware-software integration in achieving superior processing efficiency and accuracy. The article on emerging hardware architectures, software optimization techniques, and standardization efforts provides valuable insights into future development directions while acknowledging the challenges in system integration and deployment. As the field continues to evolve, the convergence of specialized hardware accelerators, sophisticated software frameworks, and standardized implementation methodologies promises to further enhance the capabilities and accessibility of computer vision and machine learning systems across diverse applications. This progression, coupled with ongoing innovations in neuromorphic computing and quantum-inspired algorithms, positions computer vision technology for continued growth and advancement in meeting the demanding requirements of next-generation applications.

REFERENCES

[1]. Cognitive Market Research, "Global Computer Vision Market Report 2023," Cognitive Market Research, Dec. 2022. [Online]. Available: https://www.cognitivemarketresearch.com/assets/client_dashboard/pdf/Global_Computer_Vision_Market_Report_20231686062682.pdf

[2]. Sukhpal Singh Gill et al., "Modern computing: Vision and challenges," Telematics and Informatics Reports, vol. 13, March 2024. [Online]. Available: https://doi.org/10.1016/j.teler.2024.100116

[3]. Xubin Wang et al., "Optimizing Edge AI: A Comprehensive Survey on Data, Model, and System Strategies," arXiv:2501.03265v1, 4 Jan. 2025. [Online]. Available: https://arxiv.org/pdf/2501.03265

[4]. State of the art Computer Vision, https://paperswithcode.com/sotaTI C6xx specification, https://www.ti.com/lit/an/sprabg7/sprabg7.pdf?ts=1742219571737

[5]. Guozhao Zheng et al., "Optimizing Convolution Neural Network implementation on TI's C6678 multi core DSP," https://www.semanticscholar.org/reader/3aeea9c4b657ec1cdb0a14e3b3daa0d499707fbaKaz Sato and Cliff Young, "An in-depth look at Google's first Tensor Processing Unit(TPU)." https://cloud.google.com/blog/products/ai-machine-learning/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu

[6]. N. Mauduit et al., Lneuro 1.0: a piece of hardware LEGO for building neural network systems https://ieeexplore.ieee.org/document/129414

[7]. M.A. Viredaz et al., "MANTRA 1: a systolic neuro-computer." https://ieeexplore.ieee.org/document/714364

[8]. Isik Aybay et.al., "Classification of Neural Network Hardware." https://www.researchgate.net/publication/2470727_Classification_Of_Neural_Network_Hardware

[9]. Wenbin Li et.al., "Efficient Object Detection Framework and Hardware Architecture for Remote Sensing Images", https://www.researchgate.net/publication/336514386_Efficient_Object_Detection_Framework_and_Hardware_Architecture_for_Remote_Sensing_Images

[10]. Fabien Sanglad, "A history of NVidia Stream Multiprocessor", https://fabiensanglard.net/cuda/

[11]. Flex Logix Announces InferX High Performance IP for DSP and AI Inference: https://www.edge-ai-vision.com/2023/04/flex-logix-announces-inferx-high-performance-ip-for-dsp-and-ai-inference/