applied sciences

MDPI

Article

A Comparative Analysis of the Usability of Consumer Graphics Cards for Deep Learning in the Aspects of Inland Navigational Signs Detection for Vision Systems

Pawel Adamski and Jacek Lubczonek *

Maritime University of Szczecin, Faculty of Navigation, Waly Chrobrego 1-2, 70-500 Szczecin, Poland; p.adamski@pm.szczecin.pl

* Correspondence: j.lubczonek@pm.szczecin.pl

Abstract: Consumer-grade graphics processing units (GPUs) offer a potentially affordable and energy-efficient alternative to enterprise-class hardware for real-time image processing tasks, but systematic multi-criteria analyses of their suitability remain rare. This article fills that gap by evaluating the performance, power consumption, and cost-effectiveness of GPUs from three leading vendors, AMD, Intel, and Nvidia, in an inland water transport (ITW) context. The main objective is to assess the feasibility of using consumer GPUs for deep learning tasks involving navigational sign detection, a critical component for ensuring safe and efficient inland transportation. The evaluation includes the use of image datasets of inland water transport signs processed by widely used detector and classifier models such as YOLO (you only look once), ResNet (residual neural network l), and MobileNet. To achieve this, we propose a multi-criteria framework based on a weighted scoring method (WSM), covering 21 different characteristics such as compatibility, resting power, energy efficiency in learning and inference, and the financial threshold for technology adoption. The results confirm that consumer-grade GPUs can deliver competitive performance with lower initial costs and lower power consumption. The findings underscore the enduring value of our analysis, as its framework can be adapted for ongoing comparisons of evolving GPU technologies using the proposed methodology.

Keywords: inland shipping; navigation; vision system; graphics cards; notice marks; energy

check for updates

Academic Editor: José A. Orosa

Received: 13 February 2025
Revised: 27 April 2025
Accepted: 29 April 2025
Published: 6 May 2025

Citation: Adamski, P.; Lubczonek, J. A Comparative Analysis of the Usability of Consumer Graphics Cards for Deep Learning in the Aspects of Inland Navigational Signs Detection for Vision Systems. *Appl. Sci.* **2025**, *15*, 5142. https://doi.org/10.3390/app15095142

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The rapid development of machine learning technology is opening up new fields of application in many fields, including water transport. Currently, artificial intelligence methods are very often used to support navigation processes. This is mainly related to the state of movement of the vehicle and, in many cases, to the prediction of its movement, detection of the vehicle, or its tracking [1–6]. Real-time processing of image data, thanks to deep learning and advanced algorithms, allows for the automation of key processes such as object detection and classification. Undoubtedly, an important role in the application of machine learning is played by autonomous units [7,8], which should "think" like humans. These units set a fairly significant trend related to the use of machine learning methods, which, in the future, could be aimed at the intelligent automation of processes in the field of water transport.

In the context of inland navigation, this paper focuses on navigational signs [9] in the machine learning process, which play a key role in ensuring safety and enabling the efficient movement of vessels on waterways. Navigation marks are a basic element of the

navigational infrastructure. They provide information on safe routes, warn of potential hazards such as shoals or obstacles, and regulate water traffic. Unlike shipping, inland navigation often takes place in restricted waters such as canals, rivers, and lakes, where precise navigation is particularly important. Reading or overlooking navigational signs can lead to collisions between ships or serious damage to infrastructure. Critical infrastructure includes bridges, which are often the main restriction of navigation. A collision with a bridge can have very serious consequences caused by the interruption of land transport and communication ways or the exclusion of the shipping route from operation for a longer period. The current lists of accidents on waterways [10,11] indicate a continuous need to improve systems that will reduce them. The responsibility for reading and interpreting the signs lies with the ship's crew. However, in the case of making mistakes, the human factor is a common one, often associated with fatigue, distraction, lack of proper navigational knowledge, or simply neglect of professional duties. Difficulties in interpreting navigation signs may also result from changing weather conditions, such as fog, rain, or generally limited visibility.

The solution to the above problems may be the use of vision systems for ship detection, recognition, identification, and tracking [12–18], supported often by machine learning algorithms. The increasing role of vision sensors in autonomous vehicles can now also be noted. Taking into account the increased efficiency in information acquisition and its automated processing, data fusion, including cameras, is often used. Examples of such fusions are the combination of radar [19] and lidar [20]. Nowadays, various navigation processes are often supported by artificial intelligence methods, e.g., those related to the detection and tracking of objects in the marine environment. The spectrum of the use of cameras as part of a vision system is quite broad. This includes the detection and tracking of people, including the analysis of camera and thermal imaging data [21], analysis of algorithms for detecting vessels [22], deep neural network optimization methods for detecting marine objects on low-energy devices (edge devices) [23], the use of YOLO (v5 and v8) for real-time vessel detection and classification [24], the integration of stereovision imagery with MOOS-IvP for improved obstacle detection, and real-time USV navigation [25]. In the development aspect, the role of intelligent systems should be emphasized, including for inland vessels [26]. Ref. [27] proposes a functional division of intelligent inland vessels (e.g., intelligent navigation, hull, power plant, and energy management), indicating the development of intelligent technologies in the 2030, 2035, and 2050 perspectives. In parallel, researchers are also working on systems such as real-time traffic object detection for autonomous vehicles [28].

In general, processing imaging data with vision systems has various applications, such as inspections, analyses related to object detection and classification, environmental health monitoring, anomaly detection, and many more. Vision systems also have many advantages, such as accuracy and speed of detection, adaptation to changing conditions, or the elimination of human errors. This allows them to operate in real-time, detecting signs with high precision, even in harsh environmental conditions. By using the learning capacity of the neural networks used in them, they can be trained on different data sets. This translates into their effectiveness in various environmental conditions. An undoubted advantage may also be the ability to support the ship's crew in the process of identifying shipping marks, which automatically translates into a reduction in the risk of human error. The above features of the systems can bring many benefits, including increasing the safety of navigation, automating navigation processes, or preparing this technology for tasks related to autonomous inland waterway transport. The schematic role of vision systems in inland shipping sign detection is illustrated in Figure 1.

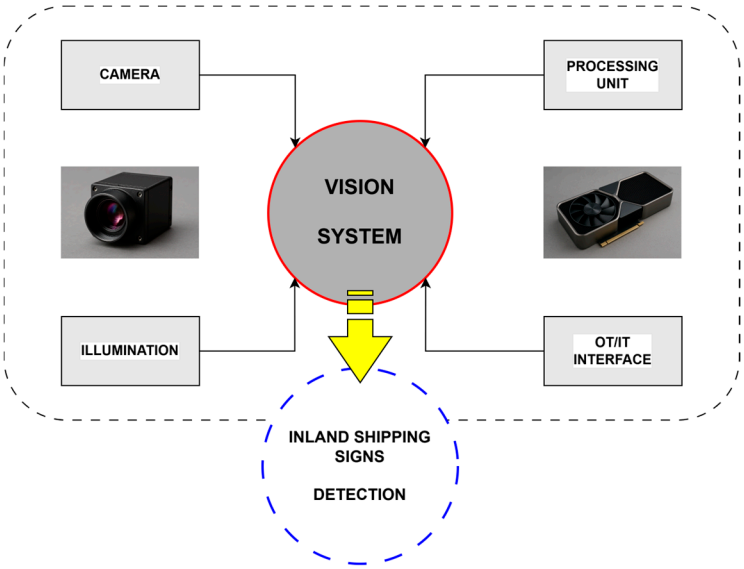


Figure 1. Vision systems as a tool for detecting navigational signs.

Deep learning is the foundation of modern vision systems, enabling automatic image analysis, including the detection and classification of various objects. Models based on neural networks, such as convolutional neural networks, are often used to perform these tasks [29]. In order to create an effective model, the network must be properly trained, which is often associated with the use of large data sets in the training process. Given this aspect, you can quickly conclude that training deep learning models is a computationally intensive process. In addition to the factor related to the size of the data (often tens of thousands of data or even millions), the computational complexity is influenced by the need to optimize a large number of parameters during the training process and the multiple processing of the same data (number of epochs). Selected problems related to the deep learning are presented in [30–32].

Graphics cards (GPUs) play a key role in accelerating deep learning calculations [33,34] thanks to the ability to perform many parallel mathematical operations. GPUs are optimized for matrix computing, which is the foundation of deep learning algorithms. Compared to traditional processors (CPUs), GPUs offer significantly higher performance for data-intensive applications. Many machine learning libraries, such as TensorFlow [35,36] and PyTorch [37], are designed to take advantage of GPU computing power. Choosing the right graphics card, therefore, has a direct impact on the efficiency and duration of the neural model training process. In addition, if we take into account the tasks related to the detection and classification of objects in real-time [38], vision systems require rapid data processing. It is clear that computer components, such as powerful GPUs, not only speed up training but also enable faster inference, which is crucial for navigation systems operating in dynamic environments. It is also safe to assume that as deep learning models become more complex and datasets become larger, hardware requirements will continue to increase. High performance of hardware, especially graphics cards, is therefore a key element enabling the practical application of deep learning in vision systems. Choosing the right hardware configuration is important not only for the effectiveness of training the models but also for their real-time applications, such as navigation systems for inland navigation.

The rapid expansion of deep learning in areas such as computer vision, robotics, and navigation has led to a surge in demand for high-performance graphics cards. Such a state of affairs has also translated into increased research interest. However, many existing comparisons of DL (Deep Learning) performance against hardware architectures focus on super throughput or benchmark results but do not take into account the practical

limitations of real-world applications. In particular, there is a lack of comprehensive studies that simultaneously assess learning performance, energy efficiency, and overall hardware compatibility in realistic scenarios. Also noticeable nowadays is the differentiated approach to graphics card testing, encompassing different environments and objectives. The following are examples of various tests related to the use of graphics cards.

Bianco et al. [39] compared the performance of different deep neural networks (DNNs) for image recognition, taking into account accuracy, complexity, computation time, and memory consumption. The experiments were conducted on two different hardware platforms (NVIDIA Titan X Pascal and NVIDIA Jetson TX1). Shi et al. [40] presented a performance comparison of four popular distributed deep learning frameworks (Caffe-MPI, CNTK, MXNet, and TensorFlow) when training neural networks, such as AlexNet, GoogleNet, and ResNet-50 [41], in single GPU, multi-GPU, and multi-node environments. Nvidia Tesla P40 graphics cards were used in the study. Ren et al. [42] presented a performance analysis of computer systems designed for deep learning workloads, such as NVIDIA DGX-2, AWS P3, IBM Power System AC922, and the Exxact TensorEX TS4 server, using AlexNet, ResNet, and BERT models. The impact of hardware configurations on GPU communication, computational performance, and scalability of the different models was evaluated. NVIDIA Tesla V100 and NVIDIA GeForce RTX 2080 Ti graphics cards were used in the tests.

Markidis et al. [43] analyzed the programmability, computational performance, and precision loss of NVIDIA Tensor Core units on the Volta architecture during mixed-precision matrix multiplication operations. The authors tested various programming interfaces such as CUDA WMMA, CUTLASS, and cuBLAS. Otterness and Anderson [44] analyzed the suitability of AMD graphics cards (AMD RX 570) as an alternative to NVIDIA in real-time systems. They discussed the advantages and disadvantages of the AMD platform and highlighted the benefits of using the open-source ROCm software for real-time GPU management research, especially in the context of flexibility and repeatability of experiments. You et al. [45] presented an analysis of the scalability of learning deep neural networks (DNNs) using the example of training AlexNet and ResNet-50 models on the Ima-geNet dataset, using very large batch sizes and the adaptive learning rate scaling algorithm LARS. Different types of NVIDIA graphics cards (Tesla P100, K20, and M40) were used in the study.

Gyawali et al. [46] analyzed the differences in runtime, memory usage, and CPU and GPU efficiency when training deep learning models using the PyTorch framework. In the experiment, they used the DenseNet121 model to classify images of cats and dogs, training it on an Intel Xeon Gold 6256 CPU and two NVIDIA A6000 graphics cards (48 GB VRAM each). Pallipuram et al. [47] compared two popular GPU architectures, NVIDIA Fermi (Tesla C2050) and AMD/ATi Radeon 5870, and two GPU programming models, CUDA and OpenCL, using them to implement and accelerate spiking neural networks (SNNs). The performance of the different implementation versions (with memory and instruction optimizations) was compared and analyzed, among other things, including computation time, data transfer, cache, and GPU resource utilization.

When analyzing the studies, it is noticeable that many of them focus on the performance of enterprise-class GPUs. As a result, there is a noticeable gap in systematic, multi-criteria analyses of consumer-grade GPUs, especially those that prioritize real-time performance, energy efficiency, and affordability. In most cases, the tests involve advanced GPU hardware architectures that are accessible to a small group of researchers. The second gap is the lack of sustainability of the tests. Given the constant technological development of GPUs, developing a methodology for their post-comparison, taking into account a wide range of criteria, would make it possible to bring a sustainable aspect to research work.

Filling the above gaps should ensure that practitioners and researchers can choose hardware configurations best suited to the performance, energy consumption, and cost constraints in the ever-changing consumer card environment. The motivation for undertaking the work in this article was to fill the aforementioned research gaps in the field:

- Consumer-grade GPUs: high-end GPUs are not always available or necessary for medium-sized research labs, small and medium-sized companies, or individuals developing real-time systems.
- There is a lack of practical multi-criteria analysis that systematically compares consumer card GPUs along multiple dimensions (performance, cost of entry, power consumption, and broader compatibility), which are critical factors in real system solutions. This is especially true for methodologies that make a lasting contribution to the post-comparison process of graphics cards.
- Requirements for specific applications: many of the existing comparisons are not directly applicable to real-time or near-real-time tasks, where both the speed of conversion and energy efficiency are most important (e.g., USV-type autonomus navigation systems).

The aim of the article was to conduct a comparative analysis of the usefulness of consumer graphics cards in the process of deep learning in the detection of navigational marks in inland navigation. The analysis was based on the WSM (weighted scoring method) [48], which allows many criteria to be taken into account when assessing the efficiency and effectiveness of the tested devices. The criteria used included compatibility, quiescent power, energy efficiency of the learning process, energy efficiency of the inference process, and the threshold for entering the technology. A total of 21 features of graphics cards were analyzed. The study included devices from three leading manufacturers, AMD, Intel, and Nvidia, tested on a set of image data related to the detection of notice marks. The main contributions of the present work are as follows:

1. Methodological contribution: We introduce a multi-criteria framework to objectively evaluate GPUs using 21 different characteristics, including performance in the learning and inference phases and energy efficiency. The framework provides a repeatable and robust approach that others can adapt and extend to different tasks or hardware configurations.
2. Application to inland navigation: We demonstrate the practical utility of our approach by applying it to the detection of navigational marks in inland navigation. Although navigational sign detection serves as a case study, the methodology and findings can be generalized to many real-time image-based tasks where timely and efficient processing is crucial.
3. Sustained comparison value: Covering devices from three leading GPU manufacturers, AMD, Intel, and Nvidia, our comparison provides a sustainable benchmark for researchers and practitioners seeking consumer-oriented GPU solutions. Because we evaluate factors such as power consumption and compatibility that are not quickly becoming obsolete, our analysis maintains validity beyond mere peak performance metrics.

2. Materials and Methods

The research process concerning the usability analysis of graphics cards in vision systems, presented in Figure 2, begins with dataset preparation (STAGE 1) and concludes with the formulation of usability conclusions (STAGE 6). In the first stage, a dataset comprising images and their annotations is prepared. The next step (STAGE 2) involves setting up the research environment, including both hardware and software configurations. STAGE 3 focuses on the selection of evaluation criteria and determining their relevance. During STAGE 4, performance tests are conducted across five groups of criteria: compatibility, idle

power consumption, training, inference, and the entry threshold to the technology. The fifth stage concentrates on a detailed analysis of the test results, with each criterion group discussed in a separate subsection in part three of the thesis. Finally, STAGE 6 presents the usability conclusions, encompassing a summary, the impact of results, and general findings. The diagram below illustrates the interconnections between the individual stages of the research process, leading to conclusions regarding the usability of consumer GPUs in vision-based applications.

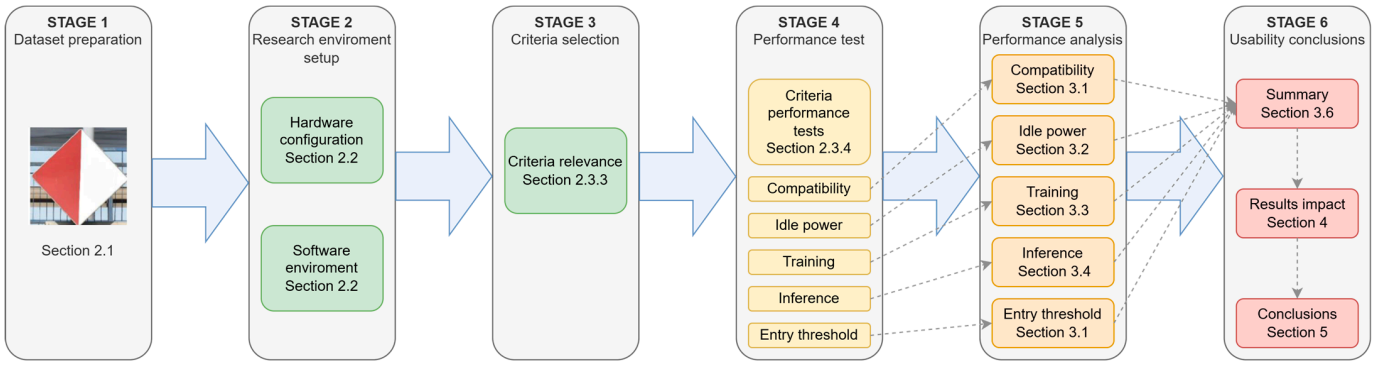


Figure 2. Workflow of the research process for GPU usability evaluation.

2.1. Description of the Datasets Used for the Analysis

All the networks used in this work for object detection were trained using a set of real photographs depicting navigational signs of inland navigation. The photographs show signs on various backgrounds, including bridges, buildings, elements of coastal infrastructure, and vegetation. The collection includes intakes taken both from the water and wharves, all of which were taken during the day in various meteorological conditions. The following situations can be specified: sunny days (normal and against the sun photos), slightly cloudy days, or foggy and heavily cloudy days. A total of 1923 photos with resolutions from 2304 × 3072 to 6000 × 4000 were collected, on which a total of 4153 objects were marked. The image dataset used in the study includes high-resolution images in the JPEG format. The dimensions of the images range from 2304 × 3072 to 6000 × 4000 pixels, with an average size of 4543.7 × 3100.4 pixels and an average aspect ratio of 1.45. File sizes range from 746.5 KB to 22 MB, with an average of 6.9 MB, reflecting the detailed visual content of the dataset. The images were taken with different cameras: Nikon D5000 (558 images), Canon PowerShot A710 IS (544), Nikon D7200 (556), DSC-RX100M3 (206), Sony Xperia Z3 Compact (23), and uTough-8010 (36). The time distribution covers the years 2008–2022, with the highest number of images recorded in 2022 (913) and 2010 (502). The images were acquired on the Oder River basin, covering the range from Szczecin to the tributary of the Nysa Łużycka River. The pictures were taken from boat and on land, often using changes in zoom to differentiate the scale of the pictures. Due to image acquisition within urbanized and non-urbanized areas, signs had various backgrounds: high and low vegetation, buildings, bridges, and often a mix of them.

Labeled marks are in various states, from looking new to old and faded. Some have been spray-painted, and some show signs of rust. The signs are visible from different angles and in different positions, and some of them are partially covered by vegetation. Examples of photos of signs are shown in Figure 3.

The collection of images intended for working with classifiers was created on the basis of the previous collection of photos. Labeled navigation marks were automatically cut out and then manually classified. The resulting set was subjected to argumentation using rotation, vertical and horizontal cropping in the range, the modification of contrast and brightness, and the addition of random noise. A total of 1000 images were generated for

each of the 50 mark classes. An example of the signs detected by the YOLO 11 [49] neural network is presented in Figure 4.

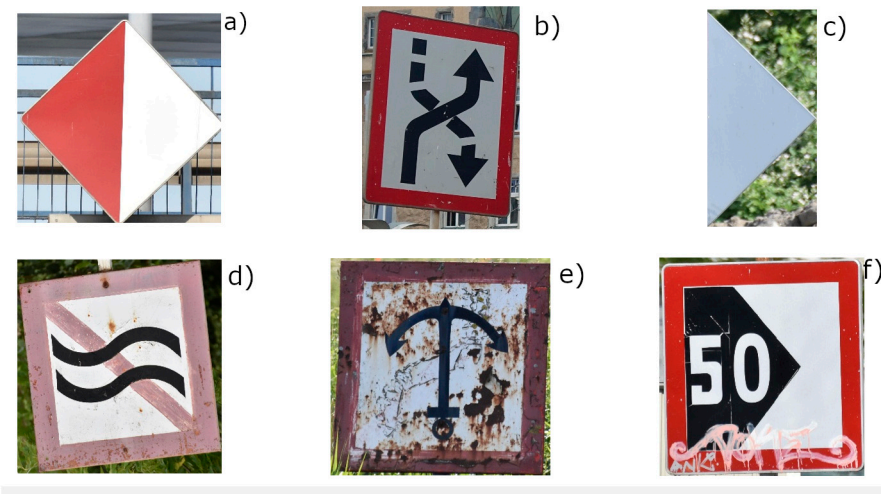


Figure 3. Example of notice marks: (**a**) notice mark against the background of a bridge, (**b**) notice mark against the background of buildings, (**c**) notice mark against the background of vegetation, (**d**) old notice mark, (**e**) rusty notice mark, and (**f**) sprayed notice mark.



Figure 4. Detected signs by the YOLO 11 neural network situated on the bridge over the Odra River in Szczecin (Poland).

2.2. Description of the Hardware and Software Platforms on Which the Tests Were Conducted

The hardware basis of the test platform is a desktop PC composed of typical components available on the consumer market. One of the goals of this publication is to validate the adaptability of a typical desktop computer to work related to neural networks, so a working unit with parameters slightly higher than a typical gaming computer was chosen as the basis (based on a Steam Hardware Survey [50]). The full hardware configuration is presented in Table 1.

The latest models of graphics cards (at the time of research) from the main manufacturers of graphics chips available on the European market, such as Intel, Nvidia, and AMD, were prepared for the test. Each of the graphics cards used in the test is intended for the consumer market to be installed on a desktop computer. The cards used in the test were the following: ASUS TUF Gaming GeForce RTX 4090 (with a Nvidia chip), ASRock Radeon RX 7900XT Phantom Gaming (with an AMD chip), and Intel ARC A750 8 GB Limited Editon. The technical specifications of the graphics cards used are listed in Table 2, using

the technology names used in the official technical documentation for AMD [51], Intel [52], and Nvidia [53].

Table 1. The hardware configuration of the test platform.

Component Type:	Model:
Processor	AMD Ryzen 7 5800X3D
CPU cooling	Arctic Liquid Freezer 2 360
Motherboard	MSI MPG B550 Gaming Plus
RAM	G.Skill AEGIS 3200 CL16 64 GB–4 × 16 GB
Disk	Lexar NM790 2TB
Power supply	SeaSonic Focus GX 850 ATX 3.0
Case	Be Quiet! Silent Base 802

Table 2. Technical parameters of the graphics cards.

Chip Manufacturer:	AMD	Intel	Nvidia
Model name	ASRock Radeon RX 7900XT Phantom Gaming	Intel ARC A750 8 GB Limited Editon	ASUS TUF Gaming GeForce RTX 4090
Chip name	Navi 31 XT	DG2-512	AD102-300-A1
Architecture	RDNA 3.0	Xe HPG	Ada Lovelace
Number of computer units	84 (computer units) 5376 (stream processors)	28 (X^e cores) 448 (X^e vector engines)	128 (streaming multiprocessors) 16384 (CUDA cores)
Number of dedicated AI computer units	168 (AI accelerators)	448 (XMX engines)	512 (Tensor cores)
VRAM	20 GB GDDR6 320 bit	8 GB GDDR6 256 bit	24 GB GDDR6X 384 bit
VRAM bandwidth	800 GB/s	512 GB/s	1008 GB/s
Typical board power (TBP)	315 W	225 W	450 W

The operating system used for testing is Ubuntu version 22.04, which was selected as it is the only one compatible with software designed for machine learning with all graphics cards used. The AMD graphics card currently does not have support for the main library dedicated to AI under Windows [54]; however, there is support for Ubuntu versions 22.04 and 24.04, and the Nvidia card provides support for Windows and Ubuntu versions 20.04, 22.04, and 24.04 [55]. Furthermore, the Intel card has drivers and APIs for Windows and Ubuntu versions 22.04 and 23.04 [56]. Ubuntu 22.04 is, therefore, the latest operating system on which it is officially possible to use all the graphics cards used for machine learning purposes. In order to avoid compatibility issues between different libraries and drivers, a separate instance of the operating system was created for each graphics card. Graphics card drivers and companion software were installed using the latest available version of Ubuntu. For driver and companion software versions, see Table 3.

Table 3. AI driver and library versions.

	AMD	Intel	Nvidia
Graphics driver version	6.8.5	2:2.99.917	550.107.02
Framework for running calculations on the GPU	HIP 6.2.41134	OneAPI 2024.2.1	CUDA Toolkit 12.4
AI library	ROCm 6.2.2	OpenVINO 2024.4.0	CuDNN 8.9.7

Performance testing was conducted on Python 3.10.12 using the same or close to the same version of the libraries for all graphics cards when not available. Where applicable, the configuration/extensions of the library prepared for a given graphics card were used. A list of the versions of libraries used for performance tests is presented in Table 4. Compatibility with neural network training and inference frameworks was checked based on the latest versions of libraries available for a given device, if any. Compatibility was checked for the following libraries: PyTorch, Tensorflow, Google JAX, SciKit, ONNX, OpenCV, and Numba.

Table 4. List of the versions of libraries used for performance tests.

	AMD	Intel	Nvidia
ONNX runtime	onnxruntime-rocm 1.19.0	onnxruntime-openvino 1.19.0	onnxruntime-gpu 1.19.0
OpenCV	4.10.0 built with OpenCL 1.2	4.10.0 built with OpenVINO 2024.4.0	4.10.0 built with CUDA 12.4 & cuDNN 8.9.7
Tensorflow	tensorflow-rocm 2.15.0	tensorflow 2.15.0	tensorflow 2.15.0
PyTorch	torch 2.3.0 + rocm6.2.3	torch 2.3.0 + cxx11.abi	torch 2.3.0

2.3. Research Methodology

2.3.1. Multi-Criteria Decision-Making Method

Comparing the usability of graphics cards in the context of use in work with vision systems requires comparing many criteria, each of which may have a different impact on the final result. This is due to factors such as the dependency of the graphics card performance on the framework used (some libraries are better suited for use with given graphics cards than others), and providing high performance when moving applications between different devices is a challenge [57]. To assess the usefulness of the cards, one of the MCDM multi-criteria decision-making methods used in the assessment of software usefulness [48] was used, i.e., the weighted scoring method. This method uses expert knowledge to determine the W_i weight of each A_i criterion. The score is calculated by adding up the value of the criterion and its weight for the given option according to the formula below:

$$S(A_i) = \Sigma W_j S_{ij} \quad (1)$$

where

$S(A_i)$ is the score of option A_i;
W_j is the weight and significance of criterion C_j;
S_{ij} is the A_i option performance for the C_j criterion.

2.3.2. Result Normalization Against the Most Efficient Solution

The performance measurement results of each criterion have different scales and ranges, so the performance of each option is evaluated in a standardized measurement for each criterion. Thanks to normalization, the results obtained by a given graphics card can be compared with others in an easier way. The results of the measurements of each P_{ij} criterion were normalized to the range of <0, 1> based on the score obtained by the best option. Two formulas were used for normalization, depending on whether the best value was the highest value (e.g., energy efficiency of image processing) or the lowest value (e.g., energy consumed by the card during idle operation) for a given criterion C*j*.

For criteria where greater is better,

$$S_{ij} = \frac{P_{ij}}{\max(P_j)} \quad (2)$$

For criteria where less is better,

$$S_{ij} = \frac{\min(P_{ij})}{P_j} \quad (3)$$

where

S_{ij} is the normalized value of the C_j criterion of the A_i option;
P_i is the set of all values of criterion C_j;
P_{ij} is the result obtained by the A_i option in the C_j criterion.

2.3.3. Criteria and Their Relevance in the Context of Comparison

A total of 21 criteria were selected, among which weights with a total of 100 were distributed. The first group of criteria concerns compatibility with popular frameworks used to work with AI in the context of vision systems. Ensuring the widest possible compatibility with popular frameworks is important because a lack thereof may expose the user to increased time needed to complete the project, e.g., by forcing a change in the main work tool. Therefore, popular frameworks for working with AI were chosen to assess the usefulness of graphics cards when working on vision systems. Only frameworks with support for performing calculations using hardware acceleration on graphics cards currently being developed were selected for testing. The selected frameworks for evaluation are as follows: PyTorch, Tensorflow, Google JAX, SciKit, ONNX, OpenCV, and Numba. The grading method provides for a scale containing three degrees of compatibility, which are presented in Table 5.

Table 5. Compatibility scale.

Point Scale	Degree of Compatibility
2	Full graphics card compatibility is ensured. It was possible to run calculations using hardware acceleration, and no significant usability limitations resulting from the graphics card used were found.
1	Compatibility is ensured, and it is possible to run calculations using hardware acceleration. Significant limitations in the functionality of a given framework using a given graphics card were found.
0	It is not compatible with a given GPU, so there is no possibility of running calculations using a framework with hardware acceleration.

Another criterion is the power consumed by the graphics card when idle. This criterion is important because it has a large impact on the cost of using a computer when performing non-GPU computing tasks. Some workstations can be used most of the time in a way that uses the graphics card only to display the image without significant load on it. Therefore, it is important that the additional component does not cause significant electricity consumption when it is not used.

Here, we present the energy efficiency criteria for training popular AI models using a collection of images described earlier. Within these criteria, the YOLO11 and ResNet50 neural network models in the PyTorch framework and MobileNetV2 [58] and ResNet50 in the Tensorflow framework were trained. Energy efficiency was adopted as a criterion due to the more universal nature of this parameter. A graphics card that is significantly faster does not necessarily have to be economical and may cause an increased cost related to electricity consumption, so the assessment took into account the speed of calculations and the demand for electricity at the same time.

Inference energy efficiency defines how efficiently a given graphics card is able to process an image using an already-trained AI model to perform object detection and classification tasks. Criteria related to the energy efficiency of inference are particularly important when processing large sets of measurement data, which can take a significant

amount of time to process. The following are the determined energy efficiency criteria in models and frameworks: YOLO11 and PesNet50 in PyTorch, MobileNetV2 and ResNet50 in Tensorflow, YOLO11 and ResNet50 in ONNX, and YOLOv4 [59] and EfficientNet B0 [60] in OpenCV.

The last criterion is the threshold for entry into a given technology, which represents the price of the cheapest model of the same generation graphics card as the tested GPU. This parameter is important when equipping the laboratory for the purpose of training the team to use a given technology.

2.3.4. Description of the Criteria Performance Tests

All tests were conducted with one connected IIyama PL2770QS monitor with a resolution of 2560 × 1440 and a refresh rate of 165 Hz via a DisplayPort cable. The operating system used is Ubuntu 22.04. The graphics card monitoring software used is amd-smi [61] for AMD, xpumcli [62] for Intel, and nvidia-smi [63] for Nvidia. A description of the criteria benchmarking is provided below.

1. Compatibility criteria: Compatibility validation was performed based on the available documentation for the framework, and the AI library was designed to work with the graphics card. An attempt was made to run calculations for each of the frameworks to confirm compatibility.
2. Idle power criterion: The computer launched the PyCharm Community IDE and the terminal. Then, a tool was run to monitor the status of the graphics card at a frequency of 1 Hz and write to a file. The test was conducted for 10 min, after which the measurement was manually interrupted. The result of the measurement is the average power reported by the monitoring software.
3. Learning energy efficiency criteria: Two instances of the terminal were launched; the first activated the virtual Python environment [64] with the necessary libraries installed. Then, a program written in Python was launched, which used the framework to start the learning process. To measure the learning rate, tools built into the framework were used, which return the calculation rate of one epoch of the neural network model's training. The time used for calculation was the arithmetic average of all epoch times. Training of each neural network model was conducted for 30 epochs. During the learning process, a record of electricity consumption measurement was launched on the second terminal. Energy efficiency was determined according to the following formula:

$$E_{ij}\left[\frac{\text{epochs}}{\text{kJ}}\right] = \frac{1000}{P_{ij}[\text{W}] \cdot T_{ij}[\text{s}]} \quad (4)$$

where

E_{ij} is the energy efficiency expressed in epochs per kilojoule of energy;

P_{ij} is the average power of the graphics card reported by the monitoring software in watts;

T_{ij} is the average time of one epoch in seconds.

4. Inference energy efficiency criteria: The tests were carried out similarly to the learning efficiency tests, with the Python virtual environment running on one terminal for each criterion and the measurement of electricity consumption on the other. The program written in Python was designed to check a test set of photos using an already trained neural network model, determining the time allocated to the inference of each file. In order to minimize the impact of the speed of reading images from a disk, the entire test set was first loaded into RAM, and only then was the processing speed measured.

The time used to determine the energy efficiency was the average of all inference times of the test set. Energy efficiency was determined according to the following formula:

$$E_{ij}\left[\frac{\text{images}}{\text{kJ}}\right] = \frac{1000}{P_{ij}[\text{W}] \cdot T_{ij}[\text{s}]} \quad (5)$$

where

E_{ij} is the energy efficiency of inference, i.e., the number of processed images per kilojoule of energy;

P_{ij} is the average power reported by the monitoring software in watts;

T_{ij} is the average inference time of one image expressed in seconds.

5. The criterion of the technology entry threshold: Based on the list of available graphics card models in the model line of a given manufacturer, the lowest model of the graphics card offered by the manufacturer of the graphics chip was determined. These were AMD Radeon RX 7600 [65], Intel ARC A310 [66], and Nvidia–GeForce RTX 4060 [67]. Then, the price of the cheapest graphics card was determined for these chips using a price comparison website [68].

3. Results

3.1. Framework Compatibility

At this stage of the research, a comparative analysis was carried out using seven frameworks: Pytorch, Tensorflow, JAX, SciKit, ONNX, OpenCV, and Numba. The compatibility criteria are defined in Table 5. A summary of the compatibility tests is provided in Table 6, and a detailed description is presented in Sections 3.1.1–3.1.3

Table 6. Summary of compatibility with frameworks.

	AMD	Intel	Nvidia
Pytorch	2	1	2
Tensorflow	2	1	2
JAX	2	1	2
SciKit	0	1	2
ONNX	2	1	2
OpenCV	2	1	2
Numba	1	1	2

3.1.1. Nvidia Card Compatibility

The performed tests have shown full compatibility of the Nvidia product with all proven frameworks. During the tests, no functional gaps were found due to the use of the RTX 4090 graphics card, so the maximum number of points was awarded in all compatibility criteria. Preparing each framework to perform calculations using hardware acceleration was limited to downloading or compiling the appropriate version compatible with Nvidia CUDA and then indicating the graphics card as the execution device.

3.1.2. AMD Card Compatibility

The AMD graphics card is fully compatible with the Pytorch, Tensorflow, JAX, and ONNX frameworks. In these frameworks, no functional weaknesses were found due to the use of the RX 7900XT graphics card, so the maximum number of points was awarded for these criteria. Running libraries looks identical to Nvidia cards. Software using these libraries can be transferred without modifying the code, thanks to the use of an identical

initialization method. AMD graphics cards in the Pytorch, Tensorflow, and JAX frameworks are detected as CUDA devices, while in the ONNX library, the names of the runtime devices are required. OpenCV does not have dedicated support for AMD graphics cards, so the only way to run calculations on the GPU is to compile them with the OpenCL library. This solution is characterized by the low speed of calculations compared to other solutions (based on tests carried out, the results of which are presented in later sections), but it provides the ability to perform calculations using hardware acceleration and no shortcomings in functionality were found. This criterion evaluates the level of compatibility, which is why the card obtained the maximum score despite the low processing speed. SciKit was not originally designed to use graphics cards for computational acceleration [69] and never supported AMD graphics cards or the OpenCL framework. RX 7900XT was found to be completely incompatible with the SciKit library, resulting in a score of zero points for this criterion. The Numba framework used to support AMD graphics cards, but based on the official Numba repository [70], it is no longer developed. Instead, it uses an outdated version of the ROCm library, which is not compatible with the GPU used. Support for AMD cards was also provided by Intel one API [71], but there is no support for RX 7000 series graphics cards. AMD released a git repository that provides the ability to run Numba on AMD MI series professional cards [72], which, despite the lack of official consumer card support, works fine with the Radeon RX 7900XT card. In the case of this library, it is also possible to run code prepared for Nvidia cards, thanks to the function that allows you to mimic a CUDA device. The library documentation mentions significant limitations such as a lack of support for half-precision operations (fp16), no ability to debug code executed on the card, no simulator similar to the one from CUDA, and no support for atomic operations using tuples and arrays. Despite the lack of official support and functionality, it is possible to run the code in the NUMBA framework on an AMD graphics card, which is why the card received one point for this criterion.

A comparison of the results presented in Table 1 for AMD and Nvidia cards indicates that the use of an AMD graphics card is associated with compatibility limitations compared to a competing product. These limitations apply to two frameworks, SciKit and Numba, which can make it difficult to work with this system. In other criteria, the AMD card provides compatibility at a level similar to the Nvidia product.

3.1.3. Intel Card Compatibility

Intel ARC A-series cards do not support double-precision floating-point (FP64) calculations, which is a direct result of the technical specification [73], which does not specify performance for this type of operation. Intel's documentation for the PyTorch [74] and TensorFlow [75] frameworks in the "Known Issues" section also indicates the lack of compatibility with FP64. The lack of support for this type of data prevents the use of neural network models that use FP64 in their structure or during the training process. To confirm this incompatibility, tests were carried out on all frameworks used. The test results confirmed the lack of support, and an attempt to perform calculations on FP64 data ended in an error. Due to this compatibility limitation, the Intel card was awarded one point in all compatibility-related evaluation criteria. The results presented in Table 1 indicate the significantly lower compatibility of the Intel card compared to the Nvidia card.

The use of the Intel graphics card in the PyTorch, TensorFlow, and JAX frameworks requires partial modification of the code intended for training on Nvidia cards, which is related to the need to use dedicated extensions, such as IPEX (Intel PyTorch Extension) in the case of PyTorch [74]. Direct execution of code prepared for Nvidia graphics cards on Intel cards without applying these extensions is not possible in the mentioned frameworks. However, code using the ONNX framework can run on Intel cards without making changes,

provided that the appropriate device is specified during initialization. On the other hand, the OpenCV library, built with OpenCL and OpenVINO extensions, only requires you to indicate the right device when running a neural network.

3.2. Measurement of Idle Power

The lowest idle power demand was recorded for the Nvidia graphics card, which achieved a result of 5.38 W, despite the highest claimed TGP of 450 W. AMD, with a lower TGP of 315 W, consumed 13.25 W in the idle test, which is more than double the power of the Nvidia card. The highest idle consumption was shown by the Intel card, which consumed 33.87 W, more than six times that of the Nvidia card. The exact measurement data are shown in Table 7. Tests were carried out with one monitor connected according to the procedure described in Section 2.3.4.

Table 7. The power drawn by the graphics card at idle power.

	AMD	Intel	Nvidia
Power [W]	13.25	33.87	5.38

During the tests, the behavior of the cards was also observed: AMD and Nvidia devices worked in passive mode with the fans turned off. The Intel card did not switch to passive mode, but the fans worked at minimum speeds, which did not cause bothersome noise.

3.3. Training Energy Efficiency

The results of the efficiency measurements are presented in Table 8. Additional time and power columns have been included to help identify the causes of high or low energy efficiency in a particular card in the test. Analysis of the measurement results shows a significant advantage of the Nvidia card when training neural networks. This card achieved energy efficiency more than double that of AMD and more than four times that of Intel in all tests conducted.

Table 8. Measure machine learning speed (T), power input (P), and energy efficiency (E).

	AMD		Intel		Nvidia		AMD	Intel	Nvidia
	T [s]	P [W]	T [s]	P [W]	T [S]	P [W]	E [ep/kJ]	E [ep/kJ]	E [ep/kJ]
YOLO 11 PyTorch	45.33	256.50	148.03	99.64	13.33	200.02	0.09	0.07	0.37
ResNet50 PyTorch	129.79	259.88	912.41	105.96	54.71	305.33	0.03	0.01	0.06
MobileNetV2 Tensorflow	9.84	229.80	47.90	98.67	3.30	262.65	0.44	0.21	1.15
ResNet50 Tensorflow	2.90	245.08	13.40	107.46	0.89	351.95	1.41	0.69	3.19

During the ResNet50 PyTorch, MobileNetV2 TensorFlow, and ResNet50 TensorFlow tests, the Nvidia card consumed the highest power (280 W on average) while achieving the shortest training times (13.33 s for YOLO 11 PyTorch, 54.71 s for Resnet50 PyTorch, 3.3 s for Mobilenet V2 Tensorflow, and 0.89 s for ResNet50 Tensorflow). The Intel card, despite the lowest power consumption during training (103 W on average), showed too low computing performance to achieve sufficiently short training times. As a result, the Intel card achieved very long learning times (148.03 s for YOLO 11 PyTorch, 912.41 s for ResNet50 PyTorch, 47.90 s for MobileNetV2 Tensorflow, and 13.40 s for ResNet50 Tensorflow), resulting in the lowest power efficiency in all tests performed.

The AMD card, during testing, consumed similar power to the Nvidia card (248 W on average) but achieved significantly worse training times compared to the Nvidia card (45.33 s for YOLO 11 PyTorch, 129.79 s for ResNet50 PyTorch, 9.84 s for MobileNetV2

Tensorflow, and 2.90 s for ResNet50 Tensorflow). Despite this, the achieved times were short enough for the AMD card to achieve better energy efficiency than the Intel card.

3.4. Inference Energy Efficiency

Neural network inference energy efficiency tests showed a much greater variety of results compared to previous tests, both in the context of the most and least effective product. The results of all the tests carried out are presented in Table 9, where, as in the previous section, the source data (inference time and power during inference) used to calculate the energy efficiency are also included.

Table 9. Measurement of operating speed (T), power consumption (P), and inference energy efficiency (E).

	AMD		Intel		Nvidia		AMD	Intel	Nvidia
	T [ms]	P [W]	T [ms]	P [W]	T [ms]	P [W]	E [im/kJ]	E [im/kJ]	E [im/kJ]
YOLO 11 PyTorch	8.42	243.43	19.99	106.17	6.62	118.00	487.98	471.25	1279.52
ResNet50 PyTorch	5.10	193.44	11.41	97.57	4.01	84.92	1014.09	898.14	2938.52
MobileNetV2 Tensorflow	36.91	36.16	40.74	56.28	38.88	60.34	749.19	436.11	426.21
ResNet50 Tensorflow	40.13	32.98	44.68	60.60	41.29	46.54	755.70	369.31	520.43
YOLO 11 ONNX	8.18	248.33	15.61	77.63	5.68	103.00	492.14	825.01	1710.56
ResNet50 ONNX	3.29	267.27	10.27	80.86	2.67	151.73	1137.19	120.69	2468.07
YOLO v4 OpenCV	605.62	235.14	110.08	80.52	34.59	313.50	7.22	112.81	92.22
EfficientNet B0 OpenCV	274.75	98.42	3.35	71.66	1.48	100.00	36.98	4159.85	6773.51

Measurements obtained for the PyTorch framework (Table 9: YOLO11 PyTorch and ResNet50 PyTorch) showed more than twice the power efficiency advantage of the Nvidia card over other solutions. AMD and Intel cards in these networks worked at much higher power while generating longer inference times.

Tests of inference in the Tensorflow framework (Table 9: MobileNetV2 Tensorflow and ResNet50 Tensorflow), in turn, showed the high efficiency of the AMD card. All of the measured cards in this framework achieved a similar inference time, and none of the cards came close to the manufacturer's power limit (TGP). This indicates that the speed limit is a factor other than the speed of the graphics chip. In such a situation, the driver that manages the energy states [76] where the graphics card is located during inference has a greater impact on energy efficiency. The AMD graphics card was in the lowest energy state during the Tensorflow inference, which allowed for the highest power efficiency.

A reference to the network models in the ONNX framework (Table 9: YOLO 11 ONNX and ResNet50 ONNX) showed the significantly higher power efficiency of the Nvidia card compared to other cards. Nvidia achieved the shortest inference times: 5.68 ms for YOLO 11 and 2.67 ms for ResNet50, with low power consumption: 103 W for YOLO 11 and 151 W for ResNet50. The AMD card achieved significantly longer inference times: 8.18 ms for YOLO 11 and 3.29 ms for ResNet50. In addition, the high power consumption during the inference, 248.33 W for YOLO 11 and 267.27 W for ResNet50, resulted in more than three times worse energy efficiency in the YOLO 11 test and more than twice as bad in the ResNet50 test compared to the Nvidia card. The Intel card worked with the lowest power consumption: 77.63 W for YOLO 11 and 80.86 W for ResNet50. At the same time, the longest inference times of 15.61 ms for YOLO 11 and 10.27 ms for ResNet50, respectively, resulted in lower power efficiency compared to Nvidia's card. Despite this, in the YOLO 11 test, the Intel card was 1.68 times more energy efficient than the AMD card. On the other hand, the ResNet50 network turned out to be 9.42 times less energy efficient than AMD.

The OpenCV library does not support AMD HIP and ROCm, which requires the use of OpenCL to perform calculations on AMD graphics cards. This results in a significant

increase in inference times, which amounted to 605.62 ms for YOLO v4 and 247.75 ms for EfficientNet B0, as shown in Table 4. During the tests, the AMD card ran at 235.14 W for YOLO v4 and 98.42 W for EfficientNet B0. The result was the lowest power efficiency compared to other cards. AMD was 12.77 times less energy efficient on YOLO v4 and more than 183 times less efficient on EfficientNet B0 compared to Nvidia. Intel achieved the highest power efficiency in the YOLO v4 network at 112.81 images per kilojoule, which is approximately 22% better than Nvidia's result (92.22 images per kilojoule). With EfficientNet B0, Nvidia achieved the highest power efficiency of 6773.51 images per kilojoule.

3.5. Technology Entry Threshold

Table 10 presents a list of the cheapest graphics cards [68] using the lowest model of current generation graphics chips that manufacturers offer. In addition to the price, there are also basic elements of graphics chip specifications.

Table 10. Prices and specifications of the cheapest models of current generation consumer graphics cards.

Model	Price [PLN]	VRAM [GB]	AI Compute Units	TGP [W]
AMD: Acer Predator BiFrost Radeon RX 7600	1049	8	64 (AI cores)	165
Intel: ASRock Intel ARC A310 Low Profile	415	4	96 (XMX Engines)	75
Nvidia: MSI GeForce RTX 4060 Ventus 2X Black	1257	8	96 (Tensor Cores)	115

The cheapest model on the list is the Intel ARC A310, equipped with 96 XMX (matrix extension) units and 4 GB of VRAM. The limited TGP design of 75 W eliminates the need for an additional power connector, resulting in low power requirements. However, the significant reduction in computing resources compared to the Intel ARC A750 (Ch. 2.2, Table 2) results in a significant performance limitation. The ARC A750 model already showed the lowest performance among the tested cards in most of the tests (Section 3.3, Table 8; Section 3.4, Table 9). Therefore, the low price threshold for entering the technology is associated with significant usability limitations, which makes it difficult for this solution to compete with the presented alternatives.

In the case of AMD, the lowest-priced offer is the Radeon RX 7600, which is equipped with 64 AI units and 8 GB of VRAM. With a TGP of 165 W, the card requires one eight-pin PCI-E power connector. Compared to the RX 7900 XT test model, the reduction in computing resources is relatively smaller than that of the Intel card. In most of the tests conducted on RX 7900 XT (Sections 3.3 and 3.4), higher performance was demonstrated compared to that of ARC A750. RX 7600's higher entry price threshold is reflected in better performance and twice the amount of VRAM, which may justify the higher cost of purchase.

The highest entry threshold is represented by the Nvidia RTX 4060, equipped with 96 Tensor Cores and 8 GB of VRAM. The TGP of the card is 115 W, which requires a PCI-E 8 PIN connector. The compute resources of the RTX 4060 have been reduced to a similar extent to that of the Intel ARC A310 with respect to higher-end models. Nevertheless, the RTX 4090 performed significantly higher in tests, both in inference tasks (Section 3.4, Table 9) and in training (Section 3.3, Table 8). For RTX 4060, the higher entry price threshold is due to the larger VRAM available and the anticipated higher performance compared to Intel cards, making this model more attractive for AI imaging applications.

3.6. Compilation of Standardized Results for Measuring the Performance of the Criteria

Table 11 shows a summary of the normalized scores, weights, and weighted performance values for the criteria analyzed. The normalization process of the results was

carried out as described in Section 2.3.2, while the final results based on the weighted scoring method are presented in the last row of Table 11, following the procedure described in Section 2.3.1.

Table 11. Summary of standardized results and weights.

	Normalized Performance			Weight	Weighted Performance		
	AMD	**Intel**	**Nvidia**		**AMD**	**Intel**	**Nvidia**
Compatibility							
Pytorch	1.00	0.50	1.00	4.00	4.00	2.00	4.00
Tensorflow	1.00	0.50	1.00	4.00	4.00	2.00	4.00
JAX	1.00	0.50	1.00	4.00	4.00	2.00	4.00
SciKit	0.00	0.50	1.00	4.00	0.00	2.00	4.00
ONNX	1.00	0.50	1.00	4.00	4.00	2.00	4.00
Opencv	1.00	0.50	1.00	4.00	4.00	2.00	4.00
Numba	0.50	0.50	1.00	4.00	2.00	2.00	4.00
Idle Power Draw	0.41	0.16	1.00	8.00	3.25	1.27	8.00
Training Energy Efficiency							
YOLO 11 PyTorch	0.24	0.19	1.00	6.00	1.46	1.14	6.00
ResNet50 PyTorch	0.50	0.17	1.00	6.00	3.00	1.00	6.00
MobileNetV2 TensorFlow	0.38	0.18	1.00	6.00	2.30	1.10	6.00
ResNet50 TensorFlow	0.44	0.22	1.00	6.00	2.65	1.30	6.00
Inference Energy Efficiency							
YOLO 11 PyTorch	0.38	0.37	1.00	4.00	1.53	1.47	4.00
ResNet50 PyTorch	0.35	0.31	1.00	4.00	1.38	1.22	4.00
MobileNetV2 Tensorflow	1.00	0.58	0.57	4.00	4.00	2.33	2.28
ResNet50 Tensorflow	1.00	0.49	0.69	4.00	4.00	1.95	2.75
YOLO 11 ONNX	0.29	0.48	1.00	4.00	1.15	1.93	4.00
ResNet50 ONNX	0.46	0.05	1.00	4.00	1.84	0.20	4.00
YOLO v4 OpenCV	0.06	1.00	0.82	4.00	0.26	4.00	3.27
EfficientNet B0 OpenCV	0.01	0.61	1.00	4.00	0.02	2.46	4.00
Technology Entry Threshold	0.40	1.00	0.33	8.00	3.16	8.00	2.64
				Total:	52.00	43.36	90.94

The highest score of 90.94 was achieved by the Nvidia graphics card, achieving the maximum normalized value in 17 out of 21 criteria. The only criterion in which this card showed by far the weakest performance was the threshold for entering the technology. The high initial cost of this card means that choosing this solution is associated with increased financial outlays to equip additional workstations with lower-segment devices. Despite this, a clear advantage in criteria related to energy efficiency, computing performance,

and compatibility with popular frameworks made this card the most effective in the overall ranking.

The second-place ranking was taken by the AMD graphics card, with a score of 52.00 points. The main differences to the detriment of this solution were observed in the criteria related to the energy efficiency of inference in popular frameworks. In the case of the TensorFlow framework, the AMD card achieved a result similar to the Nvidia card, but in other solutions (PyTorch, ONNX, and OpenCV), the power efficiency turned out to be significantly lower. In addition, the results for the energy efficiency criteria when training the models were also noticeably worse than those obtained by the Nvidia card. As a result, the AMD card gave way in most key areas of performance in favor of the solution proposed by Nvidia.

The lowest score, equal to 43.36, was obtained by the Intel graphics card. This product was characterized by significantly lower compatibility with popular frameworks, obtaining the lowest scores in all related criteria. In addition, the Intel card showed the highest idle power consumption and the lowest scores on the training energy efficiency criteria. The only criteria by which this card achieved the maximum score were the energy efficiency of YOLO v4 and the threshold for entry into the technology. However, achieving the highest score in only 2 of the 21 criteria was not enough to significantly improve the final score.

4. Discussion

The aim of the study was to conduct a comparative analysis of the usefulness of consumer graphics cards in the process of deep learning in the field of detection of navigational marks in inland navigation. Using WSM methods, a methodology for comparing cards was developed using various criteria and expert knowledge. The detection of navigational marks in inland navigation was only employed as a practical case study to demonstrate the methodology.

Based on the results of the analysis, it can be concluded that the highest performance in terms of learning and inference was achieved by the Nvidia graphics card, which, at the same time, was characterized by the highest power demand. The results in Section 3.3 (Table 8) show that it often outperformed other designs in terms of speed, even many times over, while achieving the highest power efficiency of all the models tested. This combination of parameters makes Nvidia's solutions particularly beneficial for processing large data sets, such as video surveillance data for river navigational markings. For example, processing a 6 h video recorded at 25 frames per second (equivalent to 540,000 images) with YOLO 11 takes approximately 1 h and requires 422 kJ of electricity using the Nvidia RTX 4090. By comparison, the Intel ARC A750 performs this operation in approximately 3 h and consumes 1146 kJ of power, while the AMD RX 7900XT needs approximately 1 h and 16 min and consumes 1107 kJ of electricity.

Using two graphics cards in the case of AMD or Intel would reduce the processing time, but even with this solution, the energy efficiency would remain lower compared to a single Nvidia card, resulting in higher operating costs. Therefore, in the context of frequent processing of large data sets, the choice of a solution characterized by higher energy efficiency is crucial.

Analysis of the performance results also indicates that individual graphics cards handle different frameworks differently. For example, the AMD RX 7900XT shows significantly worse results in both speed and energy efficiency in the OpenCV framework (Section 3.4, Table 9) despite its formal compatibility with this software. This suggests that the choice of graphics card should depend not only on its overall performance but also on the specifics of the development environment used. When processing large image sets

with OpenCV, the AMD card can generate excessive operating costs and extend the time of project implementation, which in extreme cases can influence their timely implementation.

Our studies also enable more detailed analysis. Based on the results obtained from the performance measurements, it is possible to not only identify the overall best-performing graphics card for research related to vision systems in inland navigation according to the adopted criteria but also to pinpoint specific solutions to challenges associated with the future deployment of such systems. For instance, the data presented in Table 11 indicate that the Intel graphics card achieved the highest energy efficiency during inference using the YOLO v4 model within the OpenCV framework. Therefore, if the development of a vision system involves the use of this specific neural network and framework, the Intel graphics card appears to be the most advantageous in terms of energy consumption. Additionally, the inference time of 110 ms (Table 9) enables its use in real-time systems, such as sign detection at a frequency of nine frames per second. Consequently, the use of the Intel card may not only improve the system's energy efficiency but also reduce overall production costs due to the significantly lower price of the hardware. In turn, for the MobileNetV2 and ResNet50 models within the TensorFlow framework, the AMD graphics card demonstrates high inference energy efficiency (Table 11), which also supports its use in the design of cost-effective and energy-efficient systems. These findings facilitate both the decision-making process related to the construction of a development environment and the selection of optimal hardware for system deployment, depending on the chosen neural network model and software framework.

A certain limitation of the research is the constantly advancing technological development of graphics cards, but the comparative methodology for new products presented in the paper can be used for future analyses. In the near future, attention should be paid to modern CPUs such as the Intel Core Ultra 285K [77], equipped with integrated NPUs (neural processing units), which open up new possibilities in the field of data processing using artificial intelligence models. According to the manufacturer's assurances, these chips are compatible with frameworks such as OpenVINO or ONNX runtime, which can significantly increase the energy efficiency of computations while eliminating the need to purchase a dedicated graphics accelerator.

Currently, Intel's processor is the only consumer desktop CPU that has an integrated NPU. Certainly, in the near future, more solutions of this type are expected to be introduced to the market, where the CPU will play a leading role in AI processing. Analogous chips supporting AI processing are already present in mobile devices, such as laptops (e.g., AMD Ryzen AI 9 HX 370, Intel Core Ultra 185H, and Qualcomm Snapdragon X Elite X1E) or mobile phones (e.g., MediaTek Dimensity 9300, Google Tensor G4, and Qualcomm Snapdragon 8 Gen 3). Therefore, future research should focus on comparing the performance of integrated NPU–CPUs with graphics cards in terms of image processing for neural network vision systems. A key aspect of this analysis will be to determine the energy efficiency of the entire workstation, as changing the computing unit will also force modifications to other system components, making direct comparison difficult.

An important role in further research will be played by tools designed for code migration between platforms [57], which will also allow for satisfactory performance without the need to redesign the code. The issue of performance portability may have a decisive impact on the implementation of new technological solutions in the near future. The lack of dedicated support for a framework or library can result in extremely slow performance, as seen for the AMD GPU in OpenCV. Migrating the neural network to another framework, in this case, is the only option to achieve the expected performance. A properly selected method should take into account both the properties of the hardware and the software included by the manufacturer, of which it is an integral part of the ecosystem.

Measuring the efficiency of a GPU is significantly more difficult than a CPU precisely because of the need to take into account software designed for a specific chip manufacturer and sometimes even for a specific graphics card.

The results of this analysis can contribute to a better understanding of hardware requirements and their impact on the development of machine vision systems used in navigation. The proposed benchmarking scheme can also be useful in evaluating future GPU solutions and adapted to dynamically changing GPU technologies, especially in the context of neural network training. The results of the research can support the process of choosing the right graphics card, taking into account both technical and economic aspects, such as the price–performance ratio. This analysis will be used both on smaller scales, such as single-user workstation configurations, and in larger projects based on multi-tenant workstation solutions. With the growing need for AI technology, we can expect an increase in demand for optimized hardware solutions that combine high computing performance with energy efficiency. In practice, the results of the research can be used to develop vision systems using deep learning technology for the automatic detection of navigational objects.

Moreover, in the context of climate change, energy efficiency and the carbon footprint of computing infrastructure are becoming increasingly important issues [78]. Machine learning and high-performance computing (HPC) can consume large amounts of energy, which contributes to greenhouse gas emissions. Therefore, choosing GPUs with a favorable performance-to-power ratio directly supports efforts to reduce the overall environmental impact of AI operations. By carefully assessing the power consumption of hardware, it is possible to mitigate some of the negative effects of increasing computing demands. This is particularly important in a variety of HPC environments to balance the rapidly growing workloads associated with artificial intelligence. Accordingly, the insights gained from the study not only support the technical and economic dimensions of artificial intelligence deployment but also fit into the list of climate change challenges through more efficient and sustainable use of computing resources.

5. Conclusions

Studies have shown that the Nvidia RTX 4090 is the most suitable consumer graphics card for processing large data sets, especially in neural network applications. This model stands out for both the highest performance and the best energy efficiency, which in many cases translates into significantly lower energy consumption compared to other tested solutions. Choosing a suboptimal graphics card can lead to a significant increase in calculation time and operating costs, which can negatively affect the implementation of the project. The AMD RX 7900XT presents higher performance than the Intel ARC A750 and, in many cases, better energy efficiency. Nevertheless, its performance is not good enough to compete with the RTX 4090. The Intel ARC A750 performed the worst in the tests, which turned out to be the least suitable card for working with large collections of images. Its main limitations are compatibility problems with popular frameworks, slow speed of operation, and the worst energy efficiency among the tested models.

In inland navigation vision systems, graphics cards will play an increasingly important role in object detection. By selecting a high-performance card, vision systems will be able to provide fast, accurate, and reliable obstacle recognition, strengthen navigation safety, and support autonomous navigation systems. With an aim to better manage risk and optimize real-time navigation, selecting the right graphics card will, therefore, be a key element in the development of intelligent navigation systems.

To summarize the studies, the best solution for a vision system for analyzed applications is graphic cards based on Nvidia chipsets. It could be the most effective in computa-

tion, AI application, and energy efficiency. The crucial issue is also their impact on carbon footprint, which also contributes to the sustainability of the use of artificial intelligence.

Author Contributions: Conceptualization, P.A. and J.L.; methodology, P.A. and J.L.; software, P.A.; validation, P.A.; formal analysis, P.A.; investigation, P.A.; resources, P.A.; data curation, P.A. and J.L.; writing—original draft preparation, P.A. and J.L.; writing—review and editing, P.A. and J.L.; visualization, P.A. and J.L.; supervision, J.L. and P.A.; project administration, J.L.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Maritime University of Szczecin, grant number 2/S/KGiT/25.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
AMD	Advanced Micro Devices
API	Application Programming Interface
ATX	Advanced Technology eXtended
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
CuDNN	CUDA Deep Neural Network library
E	Energy Efficiency of Inference
ep	Epoch (one complete pass through the entire training dataset during the model training process)
FP64	Floating Point 64
GB	Gigabyte
GPU	Graphics Processing Unit
HIP	Heterogeneous Computer Interface for Portability
HPC	High-Performance Computing
Hz	Hertz
IDE	Integrated Development Environment
IPEX	Intel® Extension for PyTorch
kj	Kilojoule
ms	Millisecond
MSI	Micro-Star International
NPU	Neural Processing Unit
ONNX	Open Neural Network Exchange
PLN	Polish Zloty
RAM	Random Access Memory
ROCm	Radeon Open Compute
s	Second
SciKit	Scientific toolkit
T	Time in seconds
TB	Terabyte
TBP	Total Board Power

TGP	Total Graphics Power
VRAM	Video Random Access Memory
W	Watt
WSM	Weighted Scoring Method
XMX	Xe Matrix eXtensions

References

1. You, L.; Xiao, S.; Peng, Q.; Claramunt, C.; Han, X.; Guan, Z.; Zhang, J. ST-Seq2Seq: A Spatio-Temporal Feature-Optimized Seq2Seq Model for Short-Term Vessel Trajectory Prediction. *IEEE Access* **2020**, *8*, 218565–218574. [CrossRef]
2. Mozaffari, S.; Al-Jarrah, O.Y.; Dianati, M.; Jennings, P.; Mouzakitis, A. Deep Learning-Based Vehicle Behavior Prediction for Autonomous Driving Applications: A Review. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 33–47. [CrossRef]
3. Azimi, S.; Salokannel, J.; Lafond, S.; Lilius, J.; Salokorpi, M.; Porres, I. A Survey of Machine Learning Approaches for Surface Maritime Navigation. 2020. Available online: http://hdl.handle.net/2117/329714 (accessed on 12 February 2025).
4. Donandt, K.; Böttger, K.; Söffker, D. Short-Term Inland Vessel Trajectory Prediction with Encoder-Decoder Models. In Proceedings of the 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022; pp. 974–979. [CrossRef]
5. Agorku, G.; Hernandez, S.; Falquez, M.; Poddar, S.; Amankwah-Nkyi, K. Traffic Cameras to Detect Inland Waterway Barge Traffic: An Application of Machine Learning, Computer Vision and Pattern Recognition. *arXiv* **2024**, arXiv:2401.03070.
6. Hart, F.; Okhrin, O.; Treiber, M. Vessel-Following Model for Inland Waterways Based on Deep Reinforcement Learning. *Ocean. Eng.* **2023**, *281*, 114679. [CrossRef]
7. Vanneste, A.; Vanneste, S.; Vasseur, O.; Janssens, R.; Billast, M.; Anwar, A.; Mets, K.; De Schepper, T.; Mercelis, S.; Hellinckx, P. Safety Aware Autonomous Path Planning Using Model Predictive Reinforcement Learning for Inland Waterways. In Proceedings of the IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society, Brussels, Belgium, 17–20 October 2022; pp. 1–6. [CrossRef]
8. Qiao, Y.; Yin, J.; Wang, W.; Duarte, F.; Yang, J.; Ratti, C. Survey of Deep Learning for Autonomous Surface Vehicles in Marine Environments. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 3678–3701. [CrossRef]
9. SIGNI. *European Code for Signs and Signals on Inland Waterways: Resolution No. 90*; United Nations: New York, NY, USA, 2018.
10. Jerzyło, P.; Rekowska, P.; Kujawski, A. Shipping Safety Management on Polish Inland Waterways. *Arch. Transp. Syst. Telemat.* **2018**, *11*, 18–22.
11. Fan, W.; Zhong, Z.; Wang, J.; Xia, Y.; Wu, H.; Wu, Q.; Liu, B. Vessel-Bridge Collisions: Accidents, Analysis, and Protection. *China J. Highw. Transp.* **2024**, *37*, 38–66.
12. Agorku, G.; Hernandez, S.; Falquez, M.; Poddar, S.; Amankwah-Nkyi, K. Real-Time Barge Detection Using Traffic Cameras and Deep Learning on Inland Waterways. *Transp. Res. Rec.* **2024**. [CrossRef]
13. Wawrzyniak, N.; Stateczny, A. Automatic Watercraft Recognition and Identification on Water Areas Covered by Video Monitoring as Extension for Sea and River Traffic Supervision Systems. *Pol. Marit. Res.* **2018**, *25*, 5–13. [CrossRef]
14. Restrepo-Arias, J.F.; Branch-Bedoya, J.W.; Zapata-Cortes, J.A.; Paipa-Sanabria, E.G.; Garnica-López, M.A. Industry 4.0 Technologies Applied to Inland Waterway Transport: Systematic Literature Review. *Sensors* **2022**, *22*, 3708. [CrossRef]
15. Wawrzyniak, N.; Hyla, T.; Popik, A. Vessel Detection and Tracking Method Based on Video Surveillance. *Sensors* **2019**, *19*, 5230. [CrossRef] [PubMed]
16. Liu, X.; Hu, Y.; Ji, H.; Zhang, M.; Yu, Q. A Deep Learning Method for Ship Detection and Traffic Monitoring in an Offshore Wind Farm Area. *J. Mar. Sci. Eng.* **2023**, *11*, 1259. [CrossRef]
17. Li, J.; Sun, J.; Li, X.; Yang, Y.; Jiang, X.; Li, R. LFLD-CLbased NET: A Curriculum-Learning-Based Deep Learning Network with Leap-Forward-Learning-Decay for Ship Detection. *J. Mar. Sci. Eng.* **2023**, *11*, 1388. [CrossRef]
18. Liu, B.; Wang, S.Z.; Xie, Z.X.; Zhao, J.S.; Li, M.F. Ship Recognition and Tracking System for Intelligent Ship Based on Deep Learning Framework. *TransNav Int. J. Mar. Navig. Saf. Sea Transp.* **2019**, *13*, 699–705. [CrossRef]
19. Shi, K.; He, S.; Shi, Z.; Chen, A.; Xiong, Z.; Chen, J.; Luo, J. Radar and Camera Fusion for Object Detection and Tracking: A Comprehensive Survey. *arXiv* **2024**, arXiv:2410.19872.
20. Wei, P.; Cagle, L.; Reza, T.; Ball, J.; Gafford, J. LiDAR and Camera Detection Fusion in a Real-Time Industrial Multi-Sensor Collision Avoidance System. *Electronics* **2018**, *7*, 84. [CrossRef]
21. Hoehner, F.; Langenohl, V.; Akyol, S.; el Moctar, O.; Schellin, T.E. Object Detection and Tracking in Maritime Environments in Case of Person-Overboard Scenarios: An Overview. *J. Mar. Sci. Eng.* **2024**, *12*, 2038. [CrossRef]
22. Rodríguez-Gonzales, J.L.; Niquin-Jaimes, J.; Paiva-Peredo, E. Comparison of Algorithms for the Detection of Marine Vessels with Machine Vision. *Int. J. Electr. Comput. Eng.* **2024**, *14*, 6332–6338. [CrossRef]
23. Heller, D.; Rizk, M.; Douguet, R.; Baghdadi, A.; Diguet, J.-P. Marine Objects Detection Using Deep Learning on Embedded Edge Devices. In Proceedings of the RSP 2022, Shanghai, China, 13 October 2022; IEEE: New York, NY, USA, 2022. [CrossRef]

24. Folarin, A.; Munin-Doce, A.; Ferreno-Gonzalez, S.; Ciriano-Palacios, J.M.; Diaz-Casas, V. Real Time Vessel Detection Model Using Deep Learning Algorithms for Controlling a Barrier System. *J. Mar. Sci. Eng.* **2024**, *12*, 1363. [CrossRef]
25. Alhattab, Y.A.; Abidin, Z.B.Z.; Faizabadi, A.R.; Zaki, H.F.M.; Ibrahim, A.I. Integration of Stereo Vision and MOOS-IvP for Enhanced Obstacle Detection and Navigation in Unmanned Surface Vehicles. *IEEE Access* **2023**, *11*, 128932–128956. [CrossRef]
26. Hao, G.; Xiao, W.; Huang, L.; Chen, J.; Zhang, K.; Chen, Y. The Analysis of Intelligent Functions Required for Inland Ships. *J. Mar. Sci. Eng.* **2024**, *12*, 836. [CrossRef]
27. Li, Y.; Hu, Y.; Rigo, P.; Lefler, F.E.; Zhao, G. (Eds.) *Proceedings of PIANC Smart Rivers 2022: Green Waterways and Sustainable Navigations*; Lecture Notes in Civil Engineering; Springer: Singapore, 2023; Volume 264. [CrossRef]
28. Hammedi, W. Smart River: Towards Efficient Cooperative Autonomous Inland Navigation. Ph.D. Thesis, Université Bourgogne Franche-Comté, Besançon, France, 2022. Available online: https://theses.hal.science/tel-03614827 (accessed on 12 February 2025).
29. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions. *J. Big Data* **2021**, *8*, 53. [CrossRef] [PubMed]
30. Sharma, O. Deep Challenges Associated with Deep Learning. In Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Perspectives and Prospects (COMITCon 2019), Faridabad, India, 14–16 February 2019; pp. 72–75. [CrossRef]
31. Tsimenidis, S. Limitations of Deep Neural Networks: A Discussion of G. Marcus' Critical Appraisal of Deep Learning. *arXiv* **2020**, arXiv:2012.15754. [CrossRef]
32. Nia, V.P.; Zhang, G.; Kobyzev, I.; Metel, M.R.; Li, X.; Sun, K.; Hemati, S.; Asgharian, M.; Kong, L.; Liu, W.; et al. Mathematical Challenges in Deep Learning. *arXiv* **2023**, arXiv:2303.15464. Available online: http://arxiv.org/abs/2303.15464 (accessed on 12 February 2025).
33. Patterson, D.A.; Hennessy, J.L. *Computer Architecture: A Quantitative Approach*, 5th ed.; Morgan Kaufmann: Waltham, MA, USA, 2012.
34. Owens, J.D.; Houston, M.; Luebke, D.; Green, S.; Stone, J.E.; Phillips, J.C. GPU Computing. *Proc. IEEE* **2008**, *96*, 879–899. [CrossRef]
35. McClure, N. *TensorFlow Machine Learning Cookbook*; Packt Publishing: Birmingham, UK, 2017.
36. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A System for Large-Scale Machine Learning. *arXiv* **2016**, arXiv:1605.08695. [CrossRef]
37. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv* **2019**, arXiv:1912.01703.
38. Manakitsa, N.; Maraslidis, G.S.; Moysis, L.; Fragulis, G.F. A Review of Machine Learning and Deep Learning for Object Detection, Semantic Segmentation, and Human Action Recognition in Machine and Robotic Vision. *Technologies* **2024**, *12*, 15. [CrossRef]
39. Bianco, S.; Cadene, R.; Celona, L.; Napoletano, P. Benchmark Analysis of Representative Deep Neural Network Architectures. *IEEE Access* **2018**, *6*, 64270–64277. [CrossRef]
40. Shi, S.; Wang, Q.; Chu, X. Performance Modeling and Evaluation of Distributed Deep Learning Frameworks on GPUs. *arXiv* **2016**, arXiv:1711.05979.
41. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
42. Ren, Y.; Yoo, S.; Hoisie, A. Performance Analysis of Deep Learning Workloads on Leading-edge Systems. In Proceedings of the 2019 IEEE/ACM Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS), Denver, CO, USA, 18 November 2019; pp. 103–113. [CrossRef]
43. Markidis, S.; Chien, S.W.D.; Laure, E.; Peng, I.B.; Vetter, J.S. NVIDIA Tensor Core Programmability. In Proceedings of the 2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Vancouver, BC, Canada, 21–25 May 2018; pp. 522–531. [CrossRef]
44. Otterness, N.; Anderson, J.H. AMD GPUs as an Alternative to NVIDIA for Supporting Real-Time Workloads. *Leibniz Int. Proc. Inform.* 2020, pp. 12:1–12:23. Available online: https://www.cs.unc.edu/~anderson/papers/ecrts20a.pdf (accessed on 12 January 2025).
45. You, Y.; Zhang, Z.; Hsieh, C.-J.; Demmel, J.; Keutzer, K. ImageNet Training in Minutes. In Proceedings of the 47th International Conference on Parallel Processing (ICPP), Eugene, OR, USA, 13–16 August 2018; pp. 1–10. [CrossRef]
46. Gyawali, D. Comparative Analysis of CPU and GPU Profiling for Deep Learning Models. *arXiv* **2023**, arXiv:2309.02521.
47. Pallipuram, V.K.; Bhuiyan, M.; Smith, M.C. A Comparative Study of GPU Programming Models and Architectures Using Neural Networks. *J. Supercomput.* **2012**, *61*, 673–718. [CrossRef]

48. Jadhav, A.; Sonar, R. Analytic Hierarchy Process (AHP), Weighted Scoring Method (WSM), and Hybrid Knowledge Based System (HKBS) for Software Selection: A Comparative Study. In Proceedings of the 2nd International Conference on Emerging Trends in Engineering and Technology (ICETET 2009), Nagpur, India, 16–18 December 2009; IEEE: Piscataway, NJ, USA; pp. 991–997. [CrossRef]
49. Sapkota, R.; Qureshi, R.; Flores-Calero, M.; Badgujar, C.; Nepal, U.; Poulose, A.; Zeno, P.; Vaddevolu, U.B.P.; Khan, S.; Shoman, M.; et al. YOLO11 to Its Genesis: A Decadal and Comprehensive Review of The You Only Look Once (YOLO) Series. *arXiv* **2025**, arXiv:2406.19407v5.
50. Valve Corporation. Steam Hardware Survey 2024. Available online: https://store.steampowered.com/hwsurvey/ (accessed on 15 November 2024).
51. Advanced Micro Devices, Inc. AMD RX 7900XT Datasheet. Available online: https://www.amd.com/en/products/graphics/desktops/radeon/7000-series/amd-radeon-rx-7900xt.html (accessed on 11 February 2025).
52. Intel Corporation. Intel ARC A750 Graphics Datasheet. Available online: https://www.intel.com/content/www/us/en/products/sku/227954/intel-arc-a750-graphics/specifications.html (accessed on 11 February 2025).
53. Nvidia Corporation. Nvidia ADA GPU Architecture Whitepaper, v2.1. Available online: https://images.nvidia.com/aem-dam/Solutions/Data-Center/l4/nvidia-ada-gpu-architecture-whitepaper-v2.1.pdf (accessed on 11 February 2025).
54. Advanced Micro Devices, Inc. ROCm Library Compatibility Matrix. Available online: https://rocm.docs.amd.com/en/docs-6.2.2/compatibility/compatibility-matrix.html (accessed on 11 February 2025).
55. Nvidia Corporation. CUDA Installation Guide for Linux. Available online: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/ (accessed on 11 February 2025).
56. Intel Corporation. Intel ARC A750 Graphics Driver Support. Available online: https://www.intel.com/content/www/us/en/products/sku/227954/intel-arc-a750-graphics/downloads.html (accessed on 11 February 2025).
57. Kwack, J.; Tramm, J.R.; Bertoni, C.; Ghadar, Y.; Homerding, B.; Rangel, E.; Knight, C.; Parker, S. Evaluation of Performance Portability of Applications and Mini-Apps across AMD, Intel and NVIDIA GPUs. In Proceedings of the 2021 International Workshop on Performance, Portability and Productivity in HPC (P3HPC), St. Louis, MO, USA, 14 November 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 45–56. Available online: https://api.semanticscholar.org/CorpusID:245542693 (accessed on 12 February 2025).
58. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. *arXiv* **2019**, arXiv:1801.04381v4.
59. Geetha, A.S. YOLOv4: A Breakthrough in Real-Time Object Detection. *arXiv* **2025**, arXiv:2502.04161v1.
60. Tan, M.; Le, Q.V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *arXiv* **2020**, arXiv:1905.11946v5.
61. Advanced Micro Devices, Inc. AMD System Management Interface (AMD SMI) Library. Available online: https://github.com/ROCm/amdsmi (accessed on 11 February 2025).
62. Intel Corporation. Intel XPU Manager and XPU System Management Interface. Available online: https://github.com/intel/xpumanager (accessed on 11 February 2025).
63. Nvidia Corporation. Nvidia System Management Interface Program. Available online: https://docs.nvidia.com/deploy/nvidia-smi/index.html (accessed on 11 February 2025).
64. Python Software Foundation. Venv–Creation of Virtual Environments. Available online: https://docs.python.org/3/library/venv.html (accessed on 11 February 2025).
65. Advanced Micro Devices, Inc. AMD Radeon Product Lineup. Available online: https://www.amd.com/en/products/graphics/desktops/radeon.html#specifications (accessed on 11 February 2025).
66. Intel Corporation. Intel Arc A-Series Graphics Product Lineup. Available online: https://www.intel.com/content/www/us/en/products/details/discrete-gpus/arc/arc-a-series/products.html (accessed on 11 February 2025).
67. Nvidia Corporation. Nvidia GeForce RTX 40 Series Product Lineup. Available online: https://www.nvidia.com/en-us/geforce/graphics-cards/40-series/ (accessed on 11 February 2025).
68. CENEO.PL. Available online: https://www.ceneo.pl/ (accessed on 11 November 2024).
69. Scikit-Learn Developers. Frequently Asked Questions. Available online: https://scikit-learn.org/stable/faq.html#will-you-add-gpu-support (accessed on 12 February 2025).
70. Numba. Tools for Using AMD ROCm with Numba. Available online: https://github.com/numba/roctools (accessed on 12 February 2025).
71. Unified Acceleration Foundation. Bringing Nvidia and AMD Support to oneAPI. Available online: https://oneapi.io/blog/bringing-nvidia-and-amd-support-to-oneapi/ (accessed on 12 February 2025).
72. Advanced Micro Devices, Inc. Numba HIP–Repository Providing ROCm HIP Backend for Numba. Available online: https://github.com/ROCm/numba-hip (accessed on 12 February 2025).
73. Intel Corporation. Introduction to the Xe-HPG Architecture. Available online: https://cdrdv2-public.intel.com/758302/introduction-to-the-xe-hpg-architecture-white-paper.pdf (accessed on 12 February 2025).

74. Intel Corporation. Intel Extension for PyTorch–GPU-Specific Issues. Available online: https://intel.github.io/intel-extension-for-pytorch/xpu/2.1.10+xpu/tutorials/performance_tuning/known_issues.html (accessed on 12 February 2025).
75. Intel Corporation. Intel Extension for TensorFlow–Release Notes. Available online: https://github.com/intel/intel-extension-for-tensorflow/releases (accessed on 12 February 2025).
76. The Kernel Development Community. GPU Power/Thermal Controls and Monitoring. Available online: https://docs.kernel.org/gpu/amdgpu/thermal.html (accessed on 12 February 2025).
77. Intel Corporation. Intel Core Ultra 9 285K Processor Datasheet. Available online: https://www.intel.com/content/www/us/en/products/sku/241060/intel-core-ultra-9-processor-285k-36m-cache-up-to-5-70-ghz/specifications.html (accessed on 12 February 2025).
78. Andrae, A.S.G.; Edler, T. On Global Electricity Usage of Communication Technology: Trends to 2030. *Challenges* **2015**, *6*, 117–157. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.