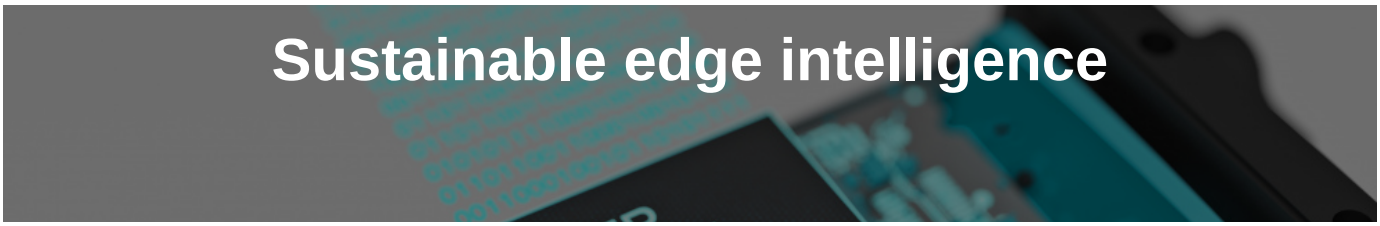
Sustainable edge intelligence



Inference accelerator for the AI of tomorrow

Artificial intelligence - both training and the actual inference - could so far mainly be developed for data centres. This trend is currently changing with the new emerging field of "Edge AI". Smartphones, robots, drones, surveillance cameras and industrial cameras - all of these devices will have some kind of AI processing built into them in the near future. When inference is to take place directly on the imaging device itself, things get interesting. How can such a high-power technology be used efficiently and sustainably outside of large data centres in the small resource-optimised embedded devices? There are already some working approaches and solutions to efficiently accelerate neural networks on edge devices. But only a few are flexible enough to keep pace with the rapidly advancing AI development.

Edge intelligence

Simply the term describes a class of devices that can solve inference tasks “on the edge” of networks with the help of neural networks and machine learning algorithms. One question that should be asked in this context is why artificial intelligence should increasingly be used in embedded devices and why Deep Learning and Deep Neural Networks are now moving into the industry's focus?

The answers to this question are not so much about AI per se, but about topics such as bandwidth, latency, security or decentralised data processing. So rather the core topics and challenges of modern Industry 4.0 applications. One important task is to reduce the inherent competition for the bandwidth of the shared communication channel by filtering or converting large amounts of sensor or camera data into usable information already on the edge devices themselves. Immediate data processing also enables process decisions to be made directly at the point of image capture without the latency of data communication. From a technical or security perspective, it may even be that reliable and continuous communication with a central processing unit, perhaps even in the cloud, is difficult to achieve or undesirable. Encapsulating acquired data on edge devices in this way also helps to decentralise data storage and processing, reducing the likelihood of a potential attack on the entire system. After all, the security of the data generated and transmitted is of great importance for every organisation.

Distributed system intelligence also creates a clear separation of job-specific tasks. For example, in a factory there may be hundreds of workstations that require an image classification service to analyse different sets of objects at each station. However, hosting multiple classifiers in the cloud comes at a cost. A cost-effective solution is desirable to train all classifiers in the cloud and send the models to the edge devices, adapted to the respective workstation. The specialisation of each model also has better performance than a classifier that makes predictions across all workstations. In addition, the simple special solutions, in contrast to data centre realisation, also reduce valuable development time. This all argues for outsourcing inference execution to edge devices.

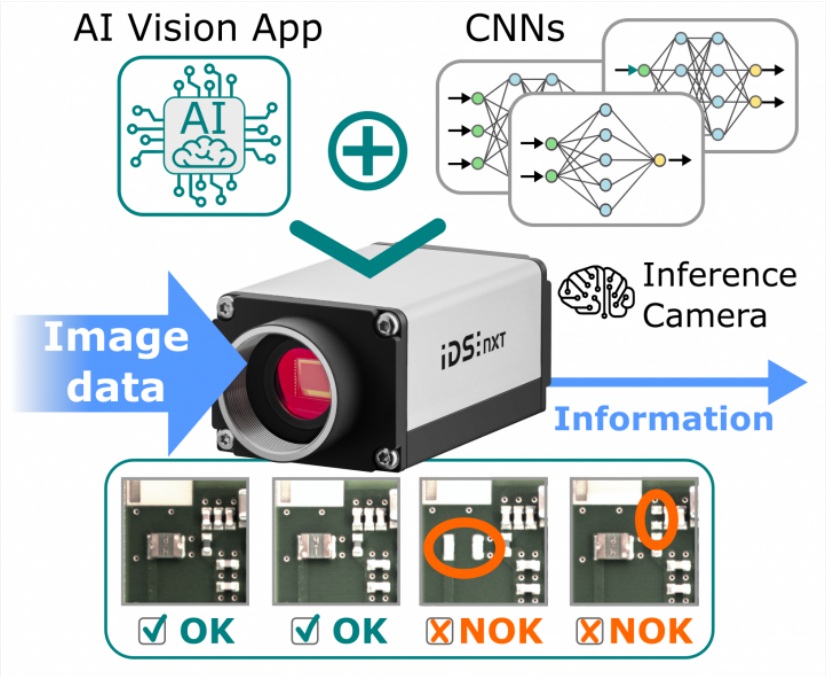


Figure 1 - Intelligent edge devices reduce large amounts of sensor and image data. They generate directly usable information on the edge and communicate only this to the control unit.

Challenge

Why are neural networks "actually" unsuitable for embedded use, or what are the challenges using them effectively "on the edge"? Performing AI inference tasks on edge devices is not that easy. Edge computing in general is all about efficiency. Edge devices usually only have limited amounts of computing, storage and energy resources available. So calculations have to be executed with high efficiency, but at the same time they should deliver high performance values and the whole thing with low latency times - which seems somehow incompatible. With the execution of CNNs, we are also dealing with the supreme discipline. CNNs in particular are known for being extremely computationally intensive, requiring billions of calculations to process an input. With millions of parameters describing the CNN architecture itself, they are in principle not a good candidate for edge computing. So-called "parameter-efficient" networks, such as MobilNet, EfficientNet or SqueezeNet, which require a smaller number of parameters to describe them, are suitable for embedded use. This significantly reduces memory requirements and computing demands. But that's not all. To further reduce storage requirements, the networks must be compressed. For example, unimportant parameters can be removed after training by so-called pruning, or the number of bits for describing the parameters can be reduced by quantisation. The reduced CNN memory size also has a positive effect on its processing time. And that leads us to the last aspect of optimisation.

Despite the use of parameter-efficient and compressed networks, a specially tailored computational system must still be used, customised for these architectures, to efficiently run AI on the edge. For this purpose, two basic system properties need to be considered. In addition to the efficiency already mentioned, the system should be flexible enough to support new developments of CNN architectures. This is important because, especially in the field of AI, new architectures and new layer types leave the development and research area every month. Things that are current and new today may already be outdated tomorrow. So what are the platform options?

Platform selection

- A **CPU-based system** undoubtedly brings the greatest flexibility. At the same time, CPUs are unfortunately extremely inefficient when running CNNs and are not very power efficient either.
- A **GPU platform** does the CNN execution with a lot of power through its compute cores working in parallel. They are more specialised than CPUs, but still have great flexibility. Unfortunately, GPUs are very power-hungry and thus rather problematic on the edge.
- The architecture of programmable **FPGAs** can be reconfigured in the field and thus adapted to new CNN architectures. Due to their parallel mode of operation, FPGAs also function very efficiently. However, their programming requires a high level of hardware knowledge.
- A full **ASIC solution** is the obvious winner in terms of efficiency as a customised integrated circuit, as it is specifically optimised to efficiently execute a given CNN architecture. However, flexibility could be a problem if new or changed CNN architectures are no longer supported.

With the characteristics of "high performance, flexible and energy efficient", the use of FPGA technology is best suited for the realisation of a CNN accelerator on edge devices at the current stage of AI development.

The ability to modify it anytime during the operation of the unit by updating with a new configuration file for special applications or CNNs makes it a solution that works long-term and is therefore suitable for industrial use. The biggest challenge using FPGA technology is the fact that programming is very complex and can only be done by specialists.

Development strategy

In order to execute neural networks in a "vision edge device", i.e. our IDS NXT cameras, we decided to develop a CNN accelerator based on FPGA technology. We call it "deep ocean core". However, in order to keep the handling of the FPGA as simple as possible in later use, not several specially optimised configurations for different CNN types should be developed, but one universally applicable architecture. This allows the accelerator to run any CNN network, if it consists of supported layers. However, since all regular layers, such as convolutional layers, addition layers, various types of pooling layers or squeezing excite layers are already supported, basically every important layer type can be used. And this completely eliminates the problem of difficult programming, because the user does not need to have any specific knowledge to create a new FPGA configuration. Through firmware updates of the IDS NXT camera, the deep ocean core is constantly evolving to support any new development in the CNN field.

deep ocean core

How does the universal CNN accelerator work or which steps in order to run a trained neural network are necessary? The accelerator only needs a "binary description" that shows which layers make up the CNN network. No programming is necessary for this. However, a neural network trained with Keras, for example, is in a special "Keras high-level language" that the accelerator does not understand. Therefore it has to be translated into a binary format that resembles a kind of "chained list". Each layer of the CNN network becomes a node end descriptor that precisely describes each layer. The end result is a complete concatenated list of the CNN in binary representation. The entire translation process is automated by a tool. No special knowledge is necessary for this either. The generated binary file is now loaded into the camera's memory and the deep ocean core begins processing. The CNN network is now running on the IDS NXT camera.

Execution flexibility

Using a CNN representation as a chained list has clear advantages in terms of accelerator flexibility. This makes it possible to switch between networks on the fly. And seamlessly, without delay. Several "linked list representations" of different neural networks can be loaded in the camera's working memory. To select a CNN for execution, the deep ocean accelerator must point at one of these lists beginning. The only thing that needs to be done is changing a "pointer value" to one of the list memories. We are talking about a simple write operation of an FPGA register, which can be done extremely fast at any time.

The following example explains why this fast switching of CNN networks can be important. Let's say you have a production line with two types of products running on it. You want to inspect the quality of the products. To do this, you must first recognise their position and then classify the quality according to product-specific defects on the basis of the recognised product category.

You could solve the task by training a huge CNN network to find the objects and classify them at the same time by pre-training every single possible failure case for each of the product groups. This is very costly and the network would become very large and possibly work only slowly, but it could work. The difficulty will be to achieve a sufficiently high accuracy. With the ability to change the active CNN network on the fly, you could decouple the localisation and classification of the different objects with the consequence that the individual CNNs are easier to train. The object recognition only has to distinguish between two classes and provide their positions. Two further networks are trained only for the respective product-specific properties and defect classes. Depending on the localised product, the camera application then decides fully automatically which classification network is activated as a result in order to also determine the respective product quality. Through this approach, the edge device works with relatively simple tasks with few parameters. As a result, the individual networks are much smaller, need to differentiate much fewer features and thus work much faster and consume less energy, making them ideally suited for execution on an edge device.

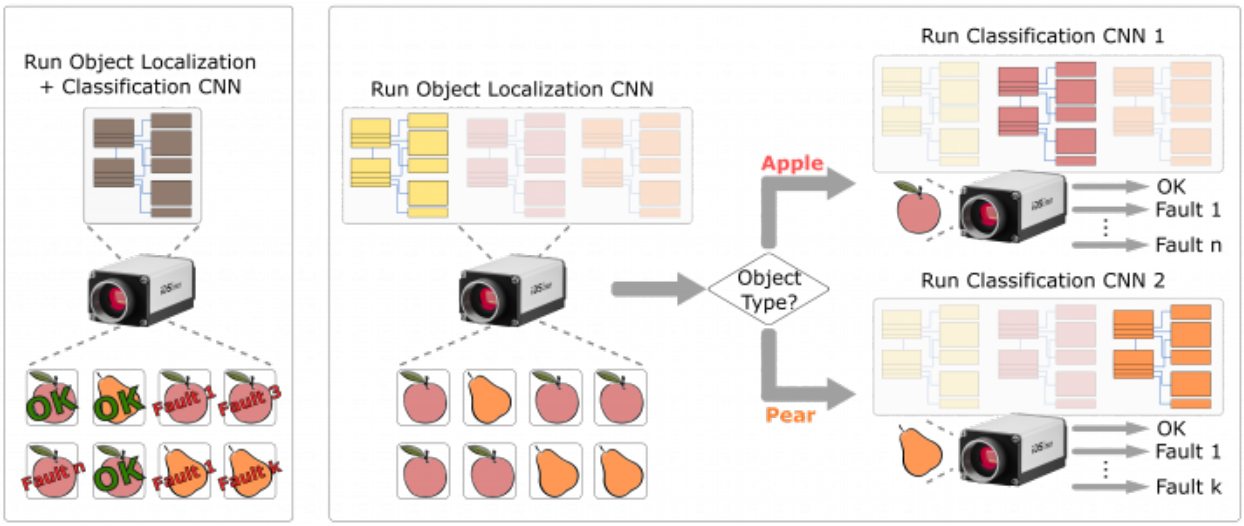


Figure 2 - Being able to change the execution of neural networks on the fly allows image analysis to be divided into simpler inference workflows that are more efficient, faster and more stable on-camera.

Performant and efficient

The FPGA-based CNN accelerator in our IDS NXT inference cameras operates on a Xilinx Zynq Ultrascale SoC with 64 compute cores. In many well-known image classification networks, such as MobileNet, SqueezeNet or EfficientNet, frame rates of up to 67 frames per second are achieved. Even on network families such as Inception or ResNet, which are considered too complex for edge computing, 20 frames per second are possible, which is quite sufficient for many applications. The FPGA implementation also allows us to further develop the performance of the deep ocean accelerator. Firmware updates will also benefit all cameras already in the field.

Even more important when it comes to edge computing, however, is the power efficiency. It indicates how many images per second a system can process per watt of energy. This makes power efficiency a good metric to compare different edge solutions. The following diagram compares different CNN accelerators. The deep ocean core as FPGA implementation, the GPU solution with a Jetson TX 2A, the classic CPU solution by a current Intel Core-i7 CPU, a Raspberry Pi as embedded CPU solution and a fully ASIC solution represented by the Intel Movidius AI chip.

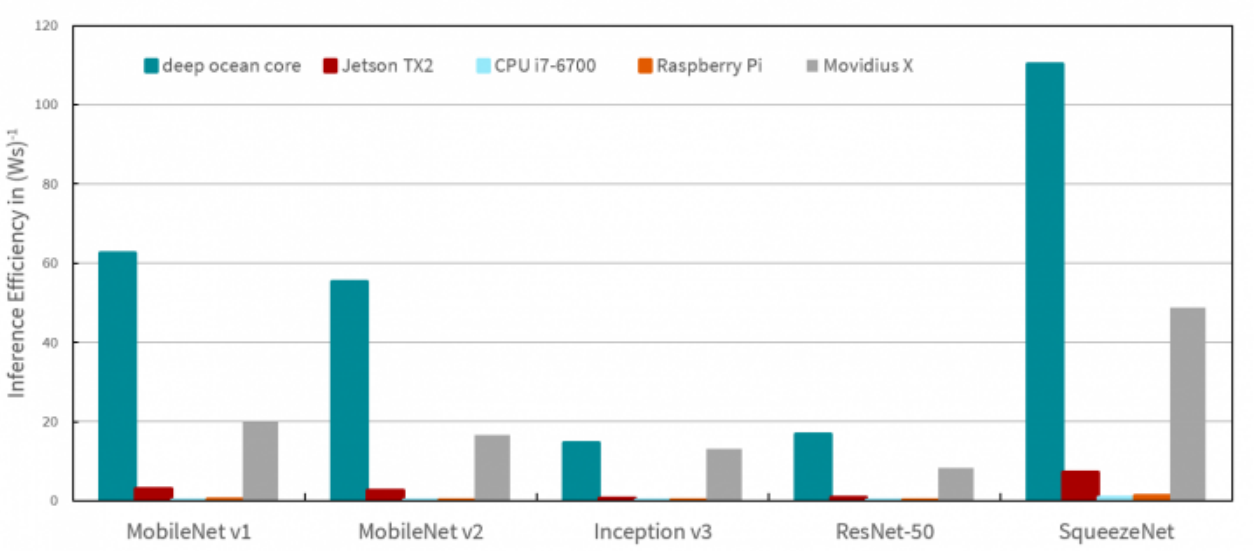


Figure 3 - Especially for parameter-efficient networks, such as the MobilNets or the SqueezeNet, you can see that the FPGA architecture is a clear winner. It has the highest power efficiency among the systems compared. This makes the deep ocean core the preferred candidate for edge intelligence.

All-in-one inference camera solution

To make the FPGA-based CNN accelerator even easier to use, IDS offers a complete inference camera solution to make the technology easily accessible to everyone. Users do not need any expertise in deep learning, image processing or camera / FPGA programming to train and run a neural network and can start AI-based image processing immediately. Easy to use tools lower the entry barrier to create inference tasks in minutes and run them immediately on a camera. In addition to the intelligent camera platform IDS NXT with the FPGA-based CNN accelerator "deep ocean core" , the overall concept also includes easy-to-use training software for neural networks. All components are developed directly by IDS and are designed to work seamlessly together. This simplifies the workflows and makes the overall system very efficient.

Sustainable edge intelligence

Each of the possibilities to accelerate neural networks mentioned in the article has individual advantages and disadvantages. If end users have to deal with the necessary components themselves in order to use AI for machine vision tasks, they like to turn to fully integrated AI accelerators, such as the Intel Movidius. Ready-to-use chip solutions work efficiently, enable unit prices that are only possible in large quantities and can be integrated into systems quickly and relatively easily thanks to extensive documentation of the functional scope. Unfortunately, there is a catch. Their long development time is unfortunately a problem in the AI environment, which has now gained enormous momentum and is changing every day. In order to develop a universally and flexibly working "edge intelligence" today, the system components must fulfil other requirements. An FPGA base is the optimal combination of flexibility, performance, energy efficiency and sustainability. After all, one of the most important requirements of an industrial product is its "industrial suitability", which is ensured, among other factors, by long availability and easy and long-term maintainability. Today, the easy-to-use IDS NXT inference camera platform combined with an FPGA CNN accelerator provides a sustainable edge intelligence end-to-end solution that eliminates the need for end users to worry about individual components and AI updates.

Further information

- Learn more about the IDS NXT embedded vision AI platform on the **product website**.
- In the technical article "**AI for all**" you will learn more about the easy entry into deep learning technology with the all-in-one inference camera solution IDS NXT ocean.