EDGE AI: DEVELOPMENTS AND APPLICATIONS

Until now, Artificial Intelligence has largely relied on cloud computing, but sometimes it cannot be considered as an option. Therefore, the edge emerges as the ideal solution, thanks to its autonomy from network connections, data security guarantees and low costs. Reply, which gained considerable experience and technical expertise in the Edge AI landscape through numerous projects, is ready to support its customers in different edge usage scenarios.

REPLY specialises in the design and implementation of solutions based on new communication channels and digital media. As a network of highly specialised companies, Reply defines and develops business models enabled by the new models of AI, big data, cloud computing, digital media and the internet of things. Reply delivers consulting, system integration and digital services to organisations across the telecom and media; industry and services; banking and insurance; and public sectors.

www.reply.com

EDGE AI: WHY DO WE NEED INTELLIGENT EDGE DEVICES?

Historically, AI applications have almost always been deployed on the cloud due to the many favorable characteristics of those environments.

First, there is the computational power that is vital for being able to train AI models in a reasonable amount of time and to make inferences with reduced latency - cloud services can provide virtually without limits.

The second advantage is that all that power can be rented for exactly the amount of time that it is needed, without wasting money on very powerful and costly hardware that would be used very few times. Third, the cloud is known for its high availability that allows for nearly no downtime in your AI services.

AUTONOMY FROM NETWORK CONNECTIONS

As the fields of deep learning and artificial intelligence continue to advance, so does the complexity of the models they produce. The training, validation, and running of these models is a very computationally expensive and resource-heavy task - especially when real time inference is required, such as video stream computer vision (including object detection and classification), or real-time data analytics applications (including regressions and pattern mining).

With time, some drawbacks of cloud environments have emerged and have become bulkier and bulkier, especially when talking about IoT applications.

The first one is the need for a constant and reliable connection to the cloud. Even if most IoT devices are getting fast access to the network, often such connections are not reliable enough for real-time applications. If autonomous vehicles would entirely depend on AI in the cloud, temporary disconnections could have fatal consequences. Moreover, there are particular scenarios where a network connection is not available at all. One example would be space travel, or the use of drones for damage detection in remote wind parks.

Another consideration is the cost factor: deploying AI algorithms directly on the edge device can be more cost efficient than setting up and maintaining a network connection.

Finally, latency can be a source of concern. In some scenarios even a 10 ms delay caused by the roundtrip from edge device to the cloud and back can be a deal breaker. A 10 ms in music processing during a live event can easily destroy the experience.

DATA PRIVACY

Apart from purely technical aspects, other considerations like user experience, security and regulatory concerns can play a role as well. Processing sensitive, personal data directly on the device rather than transferring it to a central cloud service, has distinctive advantages in terms of compliance with applicable data privacy regulation, it decreases the risk of large scale data breaches and thus reputational risks and helps to increase the user acceptance of a service.

On the edge, this data is only used for the absolute minimum amount of time required to complete inference locally on the device. The concept of federated learning is emblematic for this approach. Particularly in the context of applying machine learning models to personal or identifiable data, a very common scenario in mobile phones, tablets, but also personal computers, federated learning can help to meet high data protection standards and build user trust in services.

ECONOMIC CONSIDERATIONS

Besides privacy concerns, the reduction of costs is another driver for edge AI services. If it is true that cloud costs can simply be modulated based on the computation demand, it is also true that the computational power of the edge device is ‘free’. For example, nowadays smartphones have high computational power onboard, sometimes also with ASIC processors specifically designed for AI, that can be leveraged to do all or some of the AI work directly on the device at no cost.

With the exception of federated learning models, the model training remains predominant in the realm of cloud services, due to its need for high computational power and its discontinuous execution. Once a model has been trained on the cloud, it can be deployed on edge devices by using various tools (e.g.: TensorFlow Lite) with relative simplicity.

This way, most of the AI algorithms usually executed in the cloud, can run on edge devices: AI-powered classifiers and regressors that can be useful for a wide spectrum of applications, ranging from simple face recognition to advanced robotic arm visual control loops.

Also for anomaly detection and predictive maintenance tasks, for example by monitoring the vibrations of train axles to estimate railway degradation, edge AI is already widely in use.

Even in cases, where the AI on the edge device is not sufficient, the edge AI can serve as a preprocessor cleaning or filtering the data to reduce payloads for the cloud service, hence accelerating the process and reducing costs.

AI as a preprocessor cleaning or filtering the data

- ✓ Reduce payloads for the cloud service
- ✓ Accelerating the process
- ✓ Reducing costs

A voice assistant may not be capable of understanding human language, but they can run highly specialised AI algorithms that identify short sentences, like “Hey, Google”, that are used as markers for the beginning of the interaction, leading to a considerable reduction in network usage and cloud costs.

EDGE AI DEVICES PANORAMA

The term edge device refers to any piece of hardware that lies on a terminal node of a network like the Internet. There is a wide spectrum of different devices that fall into this category, so a finer classification is needed to for a better understanding of possible applications and use cases.

Edge devices roughly fall in two macro-categories: mist devices and real edge devices.

Mist devices

Mist devices are informally described as all those devices which are not in the cloud but are not sufficiently small and power-efficient to be integrated into IoT devices. This category includes AI appliances and GPU-equipped workstations. These are the best choice when there are no space constraints, but low latency is required. An example can be the implementation of a visual control loop for a collaborative robot.

VS

“Real” Edge devices

Small and power-efficient chips that can be easily integrated into everyday objects. While training and deploying AI algorithms on mist devices requires no specific techniques, for edge devices particular attention must be paid to the available resources and specific algorithms that must usually be applied.

In the “Real” Edge devices category there are a plethora of different devices.

→ DEDICATED EDGE DEVICES

(e.g. Nvidia Jetson)

They are equipped with a smartphone-grade CPU and a dedicated AI coprocessor, which can be a simple GPU or a more sophisticated TPU. Even if small and low-power, these devices can be used for complex applications, like real-time object recognition on high-resolution images. They usually provide a Linux-based operating system, so they can run common PC software.

→ GENERAL-PURPOSE EDGE DEVICES

(e.g. Raspberry Pi)

Similar to dedicated edge devices, but they do not have an AI coprocessor. This limits the range of applications of these devices for AI to simple models unless some accelerator system is added.

→ AI ACCELERATORS

(e.g. Google Coral)

These devices come in the form of USB sticks that are thought to be connected to general-purpose edge devices to enhance their AI capabilities, bringing them to the same level as dedicated edge devices. However, bear in mind that it’s usually more cost-effective to select a dedicated edge device from the beginning.

→ MICROCONTROLLERS

(e.g. ARM Cortex M-55, Espressif ESP32)

These devices consume much less power and are smaller, but they have no OS and a very limited set of resources. This has prevented the use of power-hungry AI applications on microcontrollers in the past years, but recently some steps have been taken to bring AI even on chips like these. Also, there exist neural accelerators specifically built to enhance the AI capabilities of microcontrollers (like the ARM ETHOS-U55) that can help in going beyond the power limits. As of now, the applications of AI algorithms on these devices are just proof of concept, but things are rapidly changing and likely this will be one of the most active research fields for AI in the near future.

→ ASICS AND FPGAS

(e.g.: Intel Stratix)

These special and costly devices are meant for very specific applications where it is important to have hardware specifically tailored for the task at hand. Crafting custom hardware that beats already existing dedicated devices in performance is a hard task that requires high experience in the field, so if possible it is preferable to go with standard devices.

→ EDGE TPUS

Designed by Google from the ground up specifically for accelerated machine learning processing, the Tensor Processing Unit (TPU) is a processing chip used to provide extremely fast computation specifically when running neural networks, built around Google's industry-leading framework TensorFlow. This processing unit is specially designed for extremely fast high-volume processing of low precision computation, making them expertly suited for machine learning applications. TPU's cannot only be utilized within a Google Cloud Platform infrastructure, but they can also be used on standalone Edge or Mist devices such as the Coral Dev Board.

Having so many options is certainly positive, since it enables a wide range of applications, but it can be overwhelming. In general, it is better to first solve the problem on standard hardware with standard techniques and then select the cheapest device that fits the power needs, always leaving room for future improvements.

Cloud Computing

∞ PF
? €

Appliance

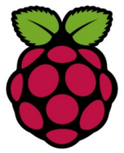
5 PF
200 K€

Workstation (w GPU)

0.03 PF
3 K€

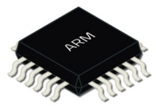
Dedicated edge device

0.002 PF
0.6 K€



Dedicated edge device

10^5 PF
0.03 K€



Micro controller

10^7 PF
0.002 K€

Cloud | **Mist** | **Edge**

EDGE AI MODEL DEVELOPMENT PROCESS AND TOOLS

The process of developing an AI model for edge devices can be decomposed into three main steps.

1

Model development

Data is analysed and the model topology is defined. The model is then implemented on a standard PC, trained and evaluated. The result of this step is a computational graph with frozen weights.

2

Model optimisation

Several techniques are applied to shrink the model size and make it more suitable to edge devices. The result can be a new computational graph or an automatically-written program.

3

Model deployment

The model deployment is quite transparent for dedicated and general-purpose edge devices: its deployment is installed on the target device and is used for inference.

MODEL DEVELOPMENT

For each of these steps, some frameworks and software tools help in simplifying the process. In particular, for step 1, the support is given by the usual AI framework: the most common tools are PyTorch and TensorFlow. Even if both are mature for common applications, as of now TensorFlow is widely regarded as the preferred choice providing many more features specifically designed for edge devices.

While designing the model architecture, it must be considered that usually RNNs have little to no support on edge devices, especially if accelerators are involved. Therefore, it is better to prefer CNNs and fully connected layers, when possible. Having a clear view on supported operations and restrictions of the chosen device can help in reducing the effort in later phases.

MODEL OPTIMISATION

The choice of the most appropriate tool for the model optimisation depends on several elements and cannot be generalised: often, different devices have different optimisation procedures, which is especially true if accelerators are involved.

TensorFlow Lite has clear advantages when dealing with mobile applications, since it integrates nicely with typical chipsets used in mobile devices.

To maximise performance on a Nvidia GPU accelerated device Tensor Rt is a must, while Open VINO allows to perfectly optimise the AI models in Intel microprocessors. There are several strategies to compress and optimise performances: model quantisation and layer fusion are widely used.

MODEL DEPLOYMENT

The model deployment is quite transparent for dedicated and general-purpose edge devices: it is sufficient to take the optimised model and run it on the device by embedding it in a program (usually written in python or C++) that reads data and feeds them into the model inputs. A different approach must be taken for non-traditional devices like accelerators, FPGAs and microcontrollers. Accelerators usually have their interface libraries that are often low level and quite unstable. For that reason, some frameworks have been developed to hide all that complexity behind a common interface. An example is Intel's Open VINO, which is capable of distributing the computation load on all available AI-enabled processors. It can even split a computational graph by placing each step on the most capable processor it finds.

For the microcontrollers field, TensorFlow Lite for microcontrollers is the number one choice. It is made up of two parts: a Python library that can be used to turn a TensorFlow Lite optimised graph into a C byte array that can simply be copied into the source code of the embedded application, and a C++ library that is capable of reading the model architecture and weights from the byte array and use them to make inference directly on the microcontroller.

As of now, the set of supported operations is quite limited, but new ones are added continuously. Another important tool to consider is CMSIS-NN, a low-level C library specifically designed to enhance a microcontrollers performance in evaluating AI models by leveraging SIMD instructions that are available on common DSP microcontrollers. CMSIS-NN is used indirectly also by TensorFlow Lite for Microcontrollers.

In the realm of FPGAs, we can cite Xilinx's FINN, a framework that is capable of optimising binarised networks and automatically create the hardware representation of them, ready to be deployed on Xilinx's FPGAs products.

OUTLOOK: AI MODEL COMPRESSION FOR EDGE DEVICES

The compression of neural networks to fit low-performance devices is an active field of research in the AI panorama. The goal is to deploy even more powerful AI models on Edge devices while reducing the energy and computational requirements.

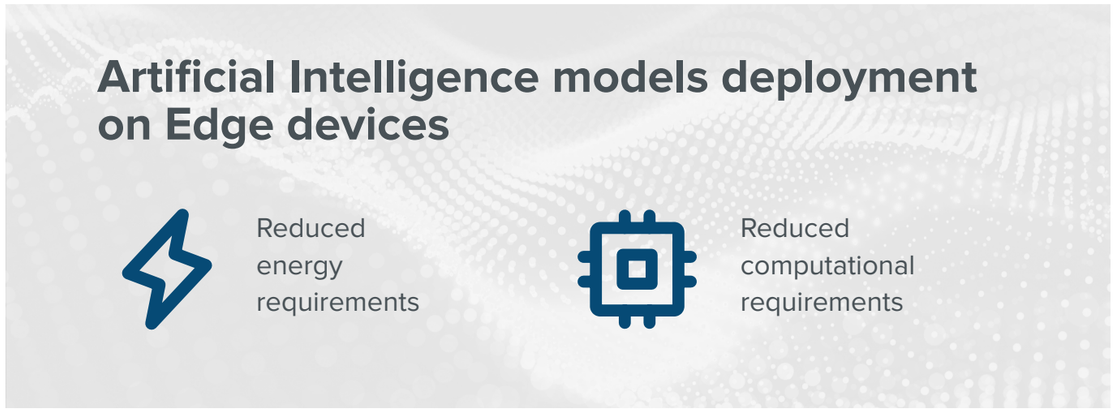


IMAGE CLASSIFICATION WITH TENSORFLOW AND TENSORRT

Image classification is one the most common tasks in the computer vision field: it can be performed on edge devices equipped with a GPU, so TensorRT is the best tool to optimise and serve your ML model. Model architectures such as MobileNet have been developed for extremely efficient edge-based computer vision and many frameworks can be used to design and develop a neural network, but there is a powerful integration between TensorFlow and TensorRT named TF-TRT that makes this process very convenient for expert users as well as beginners.
Typically, the first is to design and train a neural network using Tensorflow in the same way it would be done for a normal application that must be served on a local PC or in the cloud; then this network and its weights can be saved and passed to TensorRT. Now, optimisation takes place: Tensor RT can automatically optimise your model with a precision calibration and model quantisation as well as perform some operations at graph level that decrease the memory required to use the model and improve the efficiency and the speed at the inference time.
Finally your model can be easily and directly deployed on your edge device using again TensorRT that provides a runtime with some APIs and interfaces to connect that to C++ or Python applications.

When it comes to effectively reducing the resource requirements of machine learning models, two main approaches can be identified: **topology optimisation** and **parameters quantisation**.

TOPOLOGY OPTIMISATION

Topology optimisation aims to reduce the amount of computation required to run a model and its memory footprint by reducing the complexity of the computational graph structure.
The topology quantisation techniques can be further separated into three main categories. The first one is made up of all those approaches that reduce the complexity of the model by manual intervention. In these cases, high expertise is needed to spot redundancies in existing architectures and reduce them while keeping the accuracy acceptable.

An example is **MobileNets**, a class of neural networks designed for solving image-related problems that are specifically tailored for low-performance devices.
There are then techniques that can perform a search over a pre-defined space of possible network topologies during the training process. With these approaches, the network complexity becomes an objective to optimise at the same level as the accuracy of the model, so the training process turns into a joint optimisation problem in which a trade-off between accuracy and complexity can be set by properly weighting the two objectives.

A well-known meta-algorithm that falls into this category is **Neural Architecture Search**, which is also employed by Google in its AutoML products, enabling the users to decide if the model should be optimised for the cloud or edge devices.
These techniques have also been shown to produce more accurate networks than human-crafted ones in many different tasks. The last category contains all those approaches that apply a post-processing step to the computational graph to reduce its complexity after training. These processes are usually referred to as **network pruning algorithms**.

PARAMETERS QUANTISATION

Machine learning models themselves can be optimised, to provide maximum performance and runtime speed, whilst retaining accuracy and precision. TensorFlow, an incredibly powerful open-source library for building neural networks, offers a framework known as TFLite - a lightweight conversion of TensorFlow models, that reduces the size and computational complexity of the model, for deployment onto edge, mobile, or IoT devices.

Further optimisation can be carried out by ‘quantising’ the model - converting the 32-bit floats in the model’s structure to more efficient 8-bit integers. These optimisations come with an expected loss of model accuracy of less than 5%, ensuring that robust, precise inference is maintained.

The parameters quantisation techniques aim to reduce the complexity by decreasing the number of bits which each parameter (weights and biases) is made of and the through the kind of representation associated to those bits. The weakest form of quantisation is obtained by keeping floating point numbers but reducing their precision. This is obtained by approximating doubles to the nearest float. There are then the techniques that aim to convert the 32-bit floats in the model’s structure to more efficient 8-bit integers. These optimisations come with an expected loss of model accuracy of less than 5%, ensuring that robust, precise inference is maintained.

One example is the **quantisation algorithm implemented in TensorFlow Lite**, which is capable of producing computational graphs composed by integer operations only. This is a great advantage for low power devices lacking a floating-point unit and can speed up computation by a factor of 3, by reducing at the same time the model size by 4 times. Furthermore, there are **Binary Neural Networks**, which aim to reduce the size of every parameter to a single bit. This is also accompanied by a redesign of common neural network operations to adapt them to work on binary parameters.

This technique has a huge potential since, with a single operation, a CPU can work on many bits at the same time, leading to strong performance improvements. Quantisation can be applied both after training, which usually drastically reduces the accuracy, or during training, in which case the quantisation operation becomes part of the computational graph allowing the training process to compensate for the error introduced by approximation, leading to accuracies that are close to full-sized models.

OPENING UP NEW HORIZONS WITH EDGE AI

Reply has gained substantial experience and technical expertise in the Edge AI landscape through numerous projects, using autonomous mobile robots and drones in various scenarios.

In combination with computer vision technology and advanced Machine Learning models, the autonomy enabled or supported by AI on edge can further promote high precision automation in defect detection, increasingly accurate predictive maintenance scenarios, warehouse and facility management, including revolutionary approaches to Building Information Modeling (BIM) with LIDAR 3D generated digital twins. In addition, AI can help to improve the user experience by safeguarding personal data, processing them on the device instead of transferring them to a cloud-centric service.

High precision automation in defect detection

Accurate predictive maintenance scenarios

Warehouse and facility management

BIM with LIDAR 3D generated digital twins

INTELLIGENT QUALITY ASSURANCE FOR GREENER ENERGY

For an international industrial services provider, Machine Learning Reply has developed a solution using advanced autonomous drones for the inspection of wind turbines. The prevention of equipment breakdowns before they happen includes inspections, adjustments, regular service and planned shutdowns. Large structures such as tall buildings, power lines, wind farms and solar parks need regular visual inspection. Currently, this is often done using pictures or drone video. Both the data collection and the analysis can be managed more efficiently with the help of AI. The goal is to move towards more sustainable industrial operations by increasing the efficiency of green energy production, minimising waste and saving resources through highly efficient and intelligent processes.

AUTONOMOUS UNMANNED AERIAL VEHICLES FOR WIND PARK INSPECTION

For a leading industrial service providers, Machine Learning Reply has developed an approach utilising autonomous drones to ingest images for damage detection.

The solution relies on machine learning technology sorting the image in the pipeline in two basic categories of “defect” and “no defect” allowing for the detection of potential material damages also in other Industry 4.0 scenarios.

The drones equipped with cameras serve as visual sensors, as they are ideal for inspection of high-rise buildings and wind and solar farms. The images are stored on SD cards and can save minimum of 200 pictures. Engineers benefit from the diligence of these drones in autonomously scanning the structures for defects.

The drone does not depend on a stable network connection, that cannot be guaranteed in remote sites, but it is able to process location as well as visual data for navigation purposes without external assistance thanks to computer vision, empowered by on device AI capabilities.
The drone is able to identify an object as a wind turbine, climb to the appropriate height and circle around the object to take 360° pictures.

Once the drone has finished its tasks, it returns to the operator who downloads the images for further processing. Utilising a deep learning model developed by Machine Learning Reply, the algorithm is able to detect and categorise damage on the photos. The damage detection is performed directly on site, from the mobile PC in the vehicle of the operator.

There the model is trains itself in an unsupervised was to understand the patterns that have to be qualified as defect and categorise them according to their properties. This system is able to detect potential material defects with a high accuracy and helps to optimise availability and performance in industrial contexts.

AUTONOMOUS SUPPORT FOR AN ECOLOGICAL TRANSFORMATION

With their expertise in edge AI development, Reply has supported the customer to successfully take the next step in their journey from automation to autonomy in their pursuit of creating efficient and holistic solutions that meet the challenges of ecological change.

FEDERATED LEARNING: ML AT THE EDGE COMBINES ENHANCED UX WITH HIGH DATA PROTECTION STANDARDS

The advances of machine learning and AI have led to a breakthrough in User Experience (UX) and personalisation of digital services. Oftentimes, however, the user friendliness, ease of use comes with a trade-off with legitimate or legally established privacy concerns. The reason for these concerns stems from the fact, that behavioural or other user data is processed with machine learning models in central cloud instance, where the data of all users is stored. This creates risks and nurtures trust issues.
On the other hand, it is not in all scenarios efficient or feasible to transmit data from an edge device, be it a mobile phone, a wearable, a robotic application, an industrial machine, a vehicle or a computer to a cloud service to feed a machine learning model, and to receive feedback in acceptable time spans.

THE EDGE IS SMART

With rapid evolution of computing power in edge devices such as mobile phones, a new approach looks at overcoming privacy issues by deploying the machine learning algorithms on the devices itself. AI is moving to the edge.

Examples for this approach include the autocomplete function in smart phones. The autocomplete function learns from previous entries of the user, e.g. in a social media or messaging app, and can also take the context into consideration.

When starting to type, the user is given the most likely options based on the data available and the learnings from the machine. Naturally, these suggestions become very personal over time, and many users would feel uncomfortable to have this information become public.

The federated learning approach leverages on the AI capacity of the edge device itself to process the user's typing data, keeping it invisible for the operator of the service and for other users of the app on their device.

A SUBSIDIARITY PRINCIPLE FOR PERSONAL DATA USAGE

The machine learning model running on the individual device is referred to as 'local model'. The local model processes the personal or identifiable data, which is not transmitted to any other device. It is also possible, and quite common, to have a global model, that is centrally running on a cloud server. This global model is fed with meta-data from all the local models, that does not allow any deduction of personal data. The meta data allows to detect any issues or to improve the overall functionality of the global model and based on these learnings, the local model can be improved and the updates can be deployed with the next revision of the application, again without the need to access the personal data from the central service.

Even tech giant Google is embracing this approach to defuse persistent accusation of gathering and processing too much user data to optimise their mainly advertising-based business model. Using a Federated Learning of Cohorts (FLoC) algorithm that runs inside Google's chrome browser, the use of third party cookies will become obsolete. This ensures that Google can improve the individual user experience, uphold quasi-personalised advertising while respecting strict privacy regulations that might conflict with the usage of cookies.
Other well-known fields of application for federated learning include voice assistants, autonomous vehicles, autonomous mobile robots or drones and image recognition e.g. in photo apps.

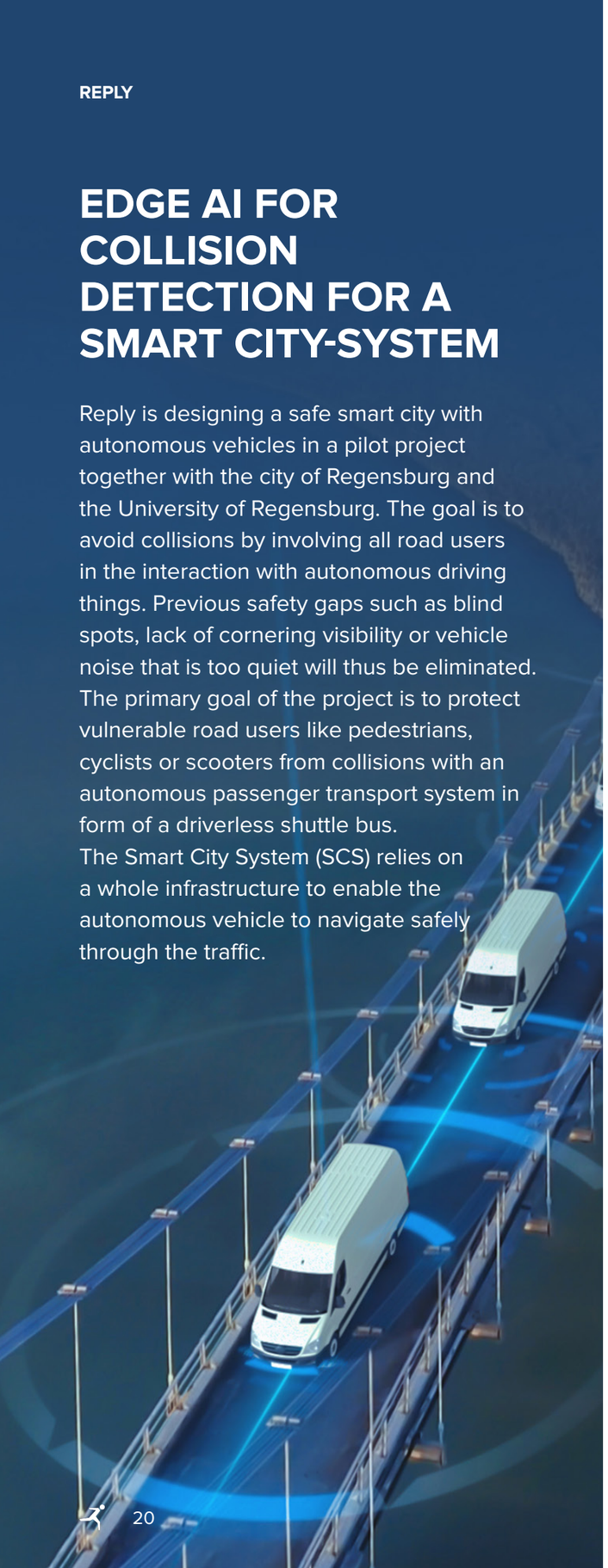
BRINGING AI TO THE EDGE WITH REPLY

Reply supports companies to develop federated learning applications to be integrated in their services and brings in in-depth expertise with state-of-the-art technologies that are supported by the globally leading tech companies like PyTorch by Facebook and TensorFlow by Google.

The case of federated learning once more demonstrates Reply's approach of constantly analysing technological trends and developments to translate them in value scaling innovation for companies across a wide range of industries.

EDGE AI FOR COLLISION DETECTION FOR A SMART CITY-SYSTEM

Reply is designing a safe smart city with autonomous vehicles in a pilot project together with the city of Regensburg and the University of Regensburg. The goal is to avoid collisions by involving all road users in the interaction with autonomous driving things. Previous safety gaps such as blind spots, lack of cornering visibility or vehicle noise that is too quiet will thus be eliminated. The primary goal of the project is to protect vulnerable road users like pedestrians, cyclists or scooters from collisions with an autonomous passenger transport system in form of a driverless shuttle bus.
The Smart City System (SCS) relies on a whole infrastructure to enable the autonomous vehicle to navigate safely through the traffic.



SMART CITY SYSTEM KEY ELEMENTS

Stationary sensors collecting traffic data
In the closed test area, Regensburg's industrial park, LIDAR sensors and cameras on street lamps, for example, continuously collect data on the traffic situation. Using several points, moving objects at an intersection are classified, for example as pedestrians or cyclists.

Dynamic sensor data determining the position of the vehicle
Computer vision and LIDAR sensors on the autonomous vehicle are used to monitor the traffic situation directly at the autonomous vehicles, including position and speed.

Neural networks in the cloud calculating the paths
The data from both sensor types are sent to the cloud via an edge device. There, they are used by neural networks that use prediction models to plan the path of the autonomous vehicle, among other things. The neural networks calculate all dependencies in the entire system in real time.

Information is fed back to autonomous vehicle and road users
The information is then delivered from the cloud to two types of recipients: Firstly, back to the AuT to control it. Secondly, via app to the road users in the test area. In the event of possible collision courses, the app issues a warning.

INTELLIGENCE FROM EDGE TO CLOUD

The solution leverages a wide array of cutting-edge technology to ensure the intelligence of the entire system, well calibrated between cloud and edge. Computer vision and LIDAR are core technologies for autonomous scenarios. Edge-AI algorithms running on high end GPUs inside the autonomous vehicle compile and classify pixels from camera or LIDAR data. This allows the identification of objects such as bicycles or pedestrians also based on specific movement profiles assigned to the objects.

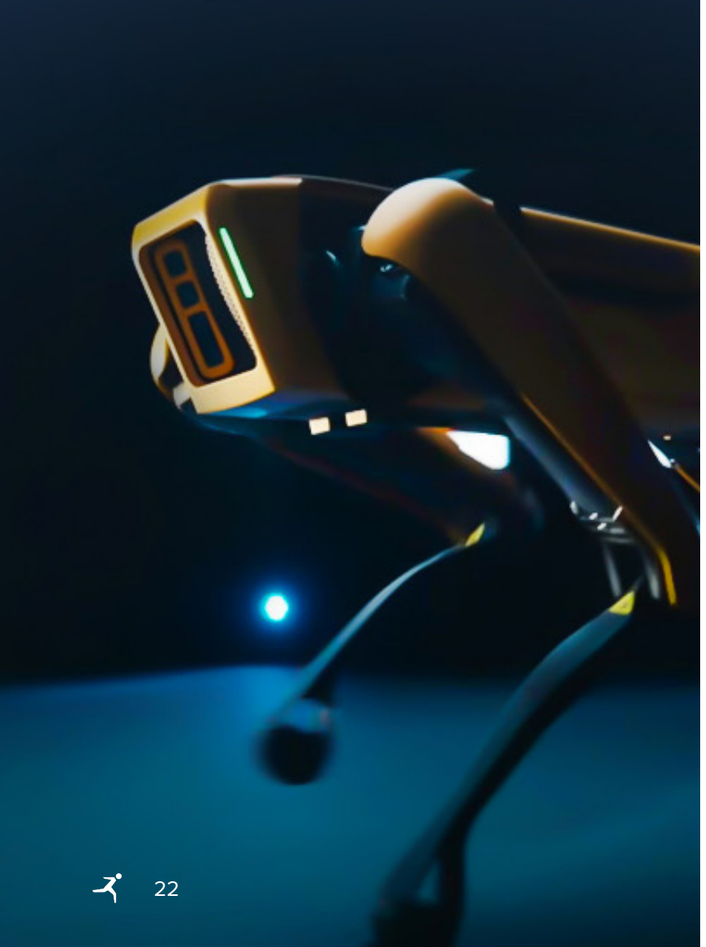
Per intersection of the route, 8 TB of data are to be sent time-synchronously via 5G to 3 edge points. Autonomous Reply's NVIDIA-based edge system processes the data into object lists. These are then sent on to the cloud. For data protection reasons, the city of Regensburg retains data sovereignty and has to ensure GDPR compliance.

Therefore, all information collected is first anonymised before transmission to a 'Smart City Cloud'. From there the data is transferred to the so-called 'People Mover Cloud', which controls the autonomous vehicles.

Autonomous Reply employs deep learning techniques to train the neural networks. In a first phase, this is done using synthetic data to create simulations using MatLab and CarMaker. In a second phase, the simulation data is expanded adding real world data.

COMPUTER VISION AND IMAGE RECOGNITION FOR AUTONOMOUS MOBILE ROBOTS

In cooperation with Microsoft, Reply has developed an end-to-end architecture for autonomous mobile robots on Microsoft Azure. Building on this architecture, Reply has developed a use case that leverages the agility of the Boston Dynamics' autonomous mobile robot SPOT for car damage recognition.



AUTONOMOUS MOBILITY AND DAMAGE DETECTION

The solution developed by Reply integrates Azure Cognitive Services, machine learning and DevOps as well as power apps and power BI. Thanks to Azure's intelligent service foundation, agile workflows and machine learning, this process can be fully automated.

Using computer vision as Edge-AI application, SPOT moves freely through the parking area and scans the license plates to find the right vehicle. Once detected, it walks around the vehicle to record its condition by continuously collecting visual data with its camera and sensors.

This information is processed "on the edge" or transmitted to the cloud, where advanced image recognition and machine leaning algorithms perform the damage detection. All detected damages are saved in the return protocol, and they can be presented to the customer and the fleet manager for approval.

A MODULAR UPGRADABILITY FOR DIFFERENT USE CASES

With their agility, autonomous mobile robots (AMR) like SPOT are able to move independently from a central infrastructure on terrain that is not traditionally designed for robots.

They can be used in hazardous environments, that are harmful to people. The autonomy of the robots is ensured through data processing and AI capabilities on the edge.

For even more computationally intensive machine learning operations, SPOT can be equipped with the Spot CORE AI. This complete development environment is available as a payload port for power and networking and comes with an Ubuntu 18.04 operating system.

Spot CORE AI is built with some of the highest performing components ever integrated in an environmentally hardened system: an Intel Xeon E3-1515M and Nvidia Quadro P5000 GPU. The modular upgradability of the robot ensures the versatility needed for the development of further, even more demanding use cases in the future.