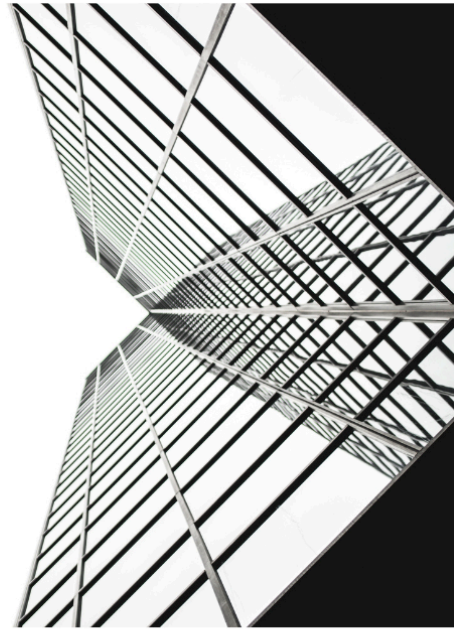2024

AI ON THE EDGE

## The Game-Changing Architecture for Every IoT Engineer

ReductStore

Introduction

The era of the Internet of Things (IoT) is opening doors to new technological advancements. With the number of connected devices expected to cross tens of billions soon, the demand for smart systems becomes crucial. This is where Artificial Intelligence (AI) plays a critical role, turning raw data into actionable insights.

Before diving into the challenges, let's have a look at the sheer volume of data generated by IoT devices. The graph below paints a compelling picture:
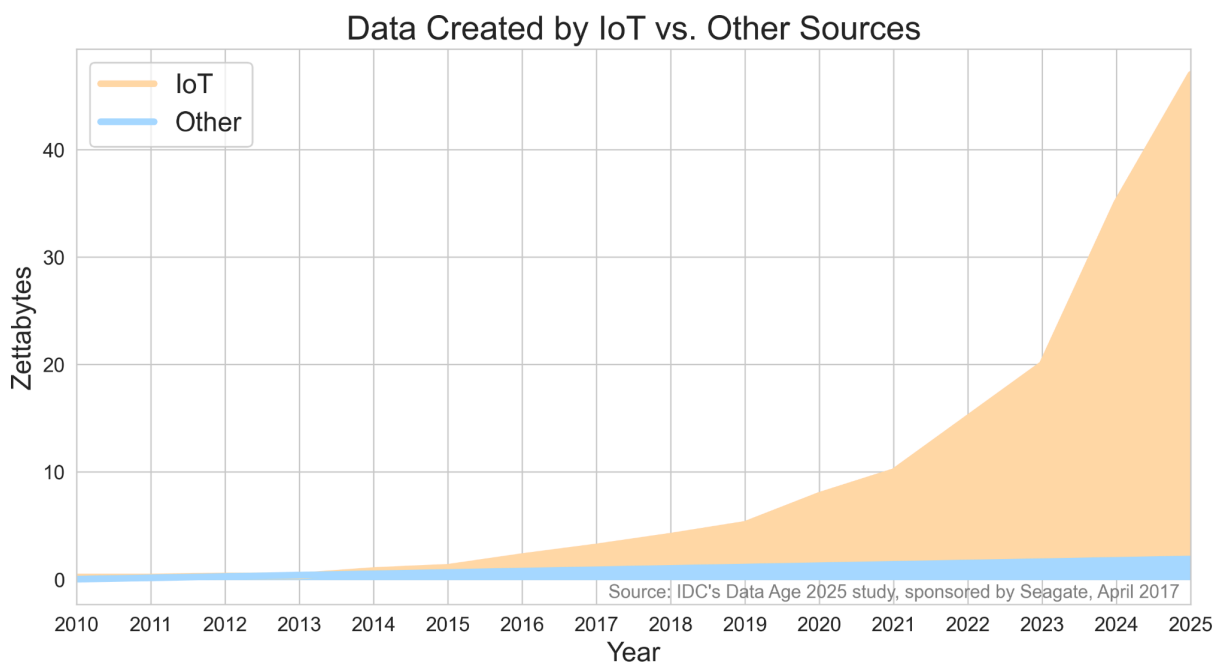
Comparing Data Generation: IoT vs. Other Sources

As depicted, the data created by IoT devices is experiencing a staggering exponential growth compared to other sources. The volume of data produced solely by IoT is eclipsing that of all other sources combined.

Such voluminous data generation brings with it a unique set of challenges.

Current Challenges in AI Development for IoT

High Volumes of Real-time Data

IoT devices produce a staggering amount of data. The challenge isn't just collecting this data, but also storing, analyzing, and deriving value from it in real-time.

Limitations of Central Processing

Reliance on centralized data processing poses challenges like latency, excessive bandwidth consumption, and security risks.

Data Organization and Monitoring

With vast amounts of data being continuously generated, organizing, and monitoring this data becomes challenging.

Model Adjustments

Different applications have varied requirements. This necessitates frequent AI model optimizations to ensure accurate and efficient operations.

Versioning and Model Retention

AI is ever-evolving. Keeping track of model changes and ensuring the right version is deployed can quickly become a challenging task.

Storage Constraints on Edge Devices

Edge devices, being limited in storage capacity, often face the risk of running out of space. This necessitates the implementation of efficient data management strategies and retention policies to ensure uninterrupted operations.

Our Solution: A Time-Series Database for Blob Data

While the vast amounts of data generated by IoT devices present a challenge, it also provides a wealth of information waiting to be tapped into. As you can imagine, it can offer insights into many areas such as user behavior, quality assurance, predictive maintenance, supply chain optimization, health outcome improvement, and much more.

However, to make the most of this data, we need the right tools—a strong storage system, speedy processing methods, and a flexible architecture that can grow with our needs.

Curious about how this works? Let's dive into an example focused on anomaly detection.

The diagram below shows the architecture for the entire operation from image processing, inference, training, model versioning, and validation.
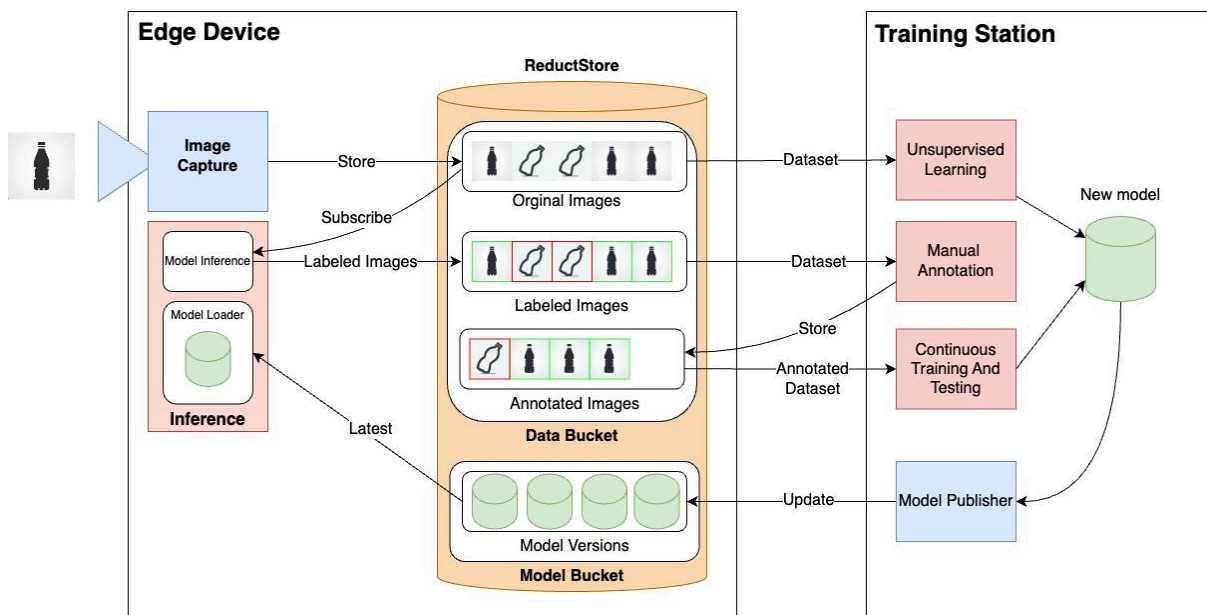

Diagram illustrating the flow of data capture, storage, inference, and training with ReductStore.

Edge Device Integration

- Image Capture: Edge devices kick off the procedure by taking real-time images.
- Model Loader: Load an instance of the latest model for inference.
- Model Inference: A runtime that uses the loaded model to process images in real-time.

ReductStore: The Data Bridge

- Data Storage: ReductStore plays a crucial role in storing original images, labeled pictures, and annotated information.
- Model Versioning: The Model Bucket stores different versions of AI models, ensuring that the right model can be easily accessed and utilized when needed.

Training Station: Where Magic Happens

- Unsupervised Learning: In our example, we use unsupervised learning to train an initial anomaly detection model out of the original images. Nevertheless, the unsupervised method assumes that the vast majority of pictures don't have any abnormalities.
- Manual Annotation: To enhance the model's accuracy, we can train it exclusively on images without anomalies. We can then validate its pattern recognition by testing it with annotated images that pinpoint the anomalies. That involves going through some pictures to annotate them: validating 'good' images and creating a mask that shows abnormalities in the 'bad' once.
- Continuous Training and Testing: Through ongoing training and evaluation, the model can adjust to evolving conditions, ensuring consistently high performance over time.
- Model Publisher: Once the model is trained and ready for deployment, the Model Publisher ensures that the latest and most efficient model is delivered to the edge devices.

How to Install ReductStore

There are multiple ways to install ReductStore from the download page. For a quick start, let's focus on using Docker.

Prerequisites

- Make sure to have Docker installed on your machine. If you don't have it yet, you can download and install it from the official Docker website.
- ReductStore currently runs on amd64, arm64, and arm32 platforms.

Run ReductStore's Docker Image

1. Open Your Terminal: Navigate to the terminal on your computer, whether it's Command Prompt, PowerShell, or a Terminal in Linux/MacOS.
2. Run the Docker Command: Execute the following command to download and run the ReductStore's Docker image:

   `docker run -p 8383:8383 -v ${PWD}/data:/data reduct/store:latest`

   - -p 8383:8383 to map port 8383 on your local machine to port 8383 on the Docker container, which is where ReductStore will be accessible.
   - -v ${PWD}/data:/data to map the ./data directory on your local machine to the /data directory in the Docker container, where ReductStore will store its data.

Validate the Installation

After running the Docker container, you may want to ensure that everything is set up correctly. Perform a simple HTTP request to check if ReductStore is functioning as expected with `curl http://127.0.0.1:8383/api/v1/info`.

If everything is set up correctly, you should see some information about the ReductStore instance you just started.
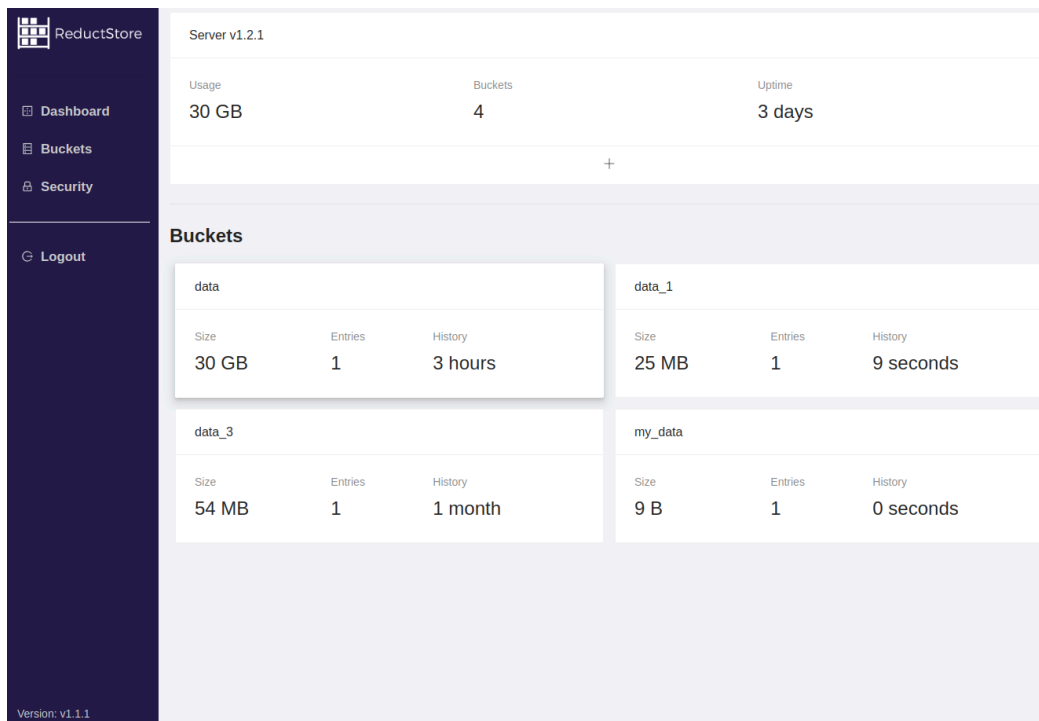
Configuration

ReductStore employs environment variables for its setup and provisioning, allowing the creation and configuration of resources like buckets and access tokens.

You can view the details in the ReductStore Configuration Documentation.

Web Console

ReductStore comes with a built-in web console for easy data management and access, which you can find at http://127.0.0.1:8383:

ReductStore Web Console: Simplified Data Management

Further Development and Resources

With the initial setup completed, the journey is just beginning. Your next steps will involve diving deeper, enhancing, and tailoring the database to suit your specific needs.

For those looking to further refine their setup, ReductStore's GitHub repository is a treasure full of resources. It's home to a range of SDKs tailored for different programming languages and use-cases.

Additionally, the official ReductStore documentation is your go-to guide for in-depth information. Whether you're troubleshooting an issue or looking to understand a feature in detail.

Moreover, we believe in collaboration and transparent development. To that end, we warmly welcome contributors and have made the entire source code available to the public. Whether you're an experienced developer or just starting out, your insights, improvements, and innovations can make a tangible difference to the ReductStore community–connect with us on Discord and be part of the journey.

Happy developing!

ReductStore

🔗 LinkedIn - ReductStore   ✉ info@reduct.store

💬 Join us on Discord   🌐 www.reduct.store