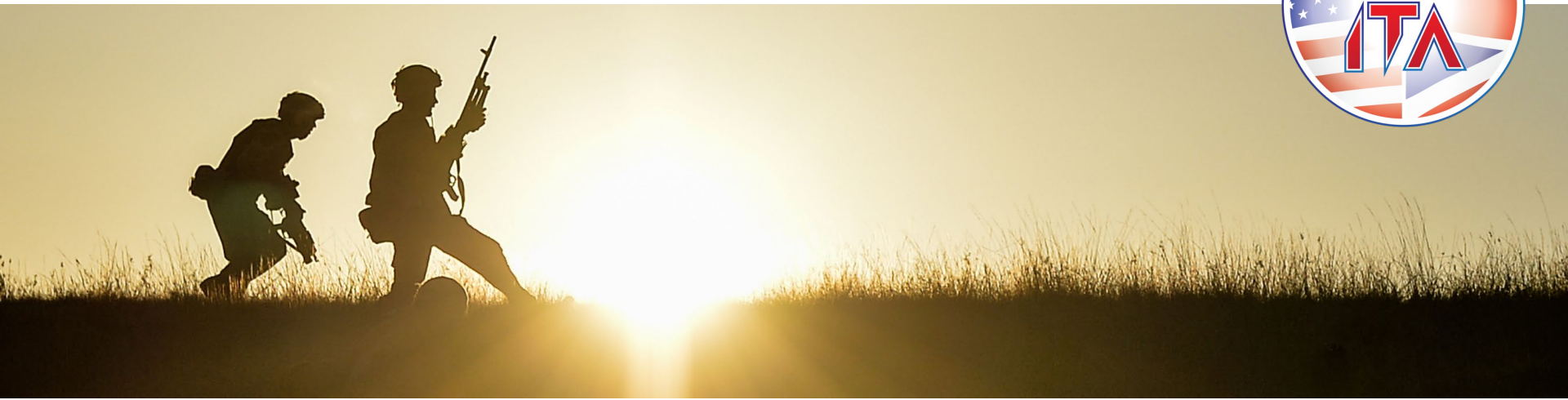


DAIS Edge Transition

IBM US March 2022

Keith Grueneberg, Xiping Wang, and Seraphin Calo

Overview

Under the DAIS Transition work, IBM deliver to U.S. Army Research Laboratories the following:

- An edge Docker container with a REST based API that implements the edge components required for core set/data summarization technologies and AI model quality assessment using model out of distribution characterization, supporting policy-based controls for core sets/data summarization and model assessment.

- A core Docker container with a REST based API that implements the core components required for core set/data summarization technologies, AI model quality assessment using model out of distribution characterization, and policy definition for core set and model quality assessment.

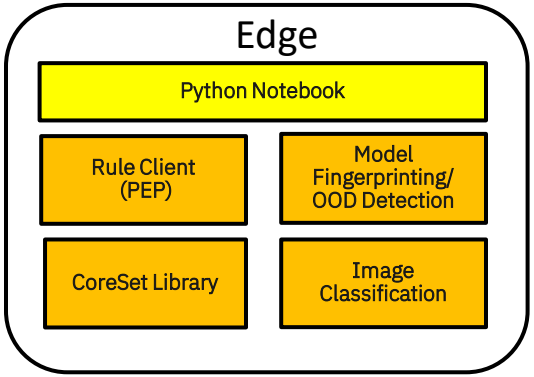
- Documentation and the source code for the software library implementing the core set/data summarization technologies, AI model quality assessment using model out of distribution characterization and policy-based control

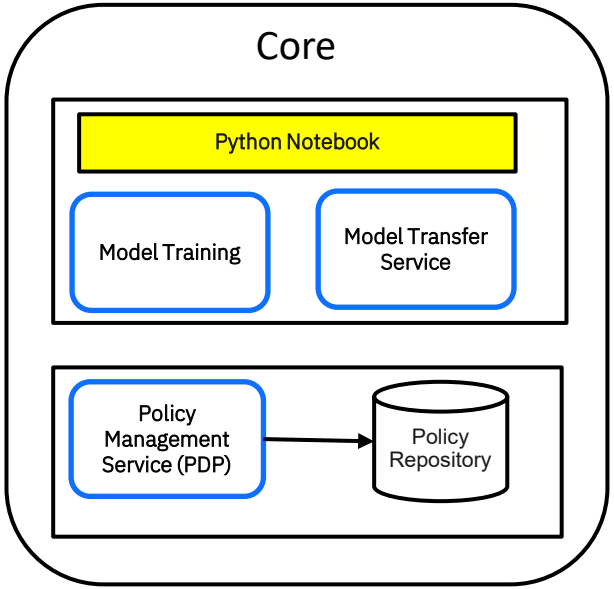
Deliverables

- Scripts:
  - Deploy Edge Docker Container
  - Deploy Core Docker Containers

- Edge:
  - GUI
    - Jupyter Notebook for Image Classification
  - Python Libraries:
    - CoreSet – Feature Selection
    - Model Management – Model. Fingerprinting/OOD Detection
    - Image Classifier
    - Rule client to evaluate OOD rules

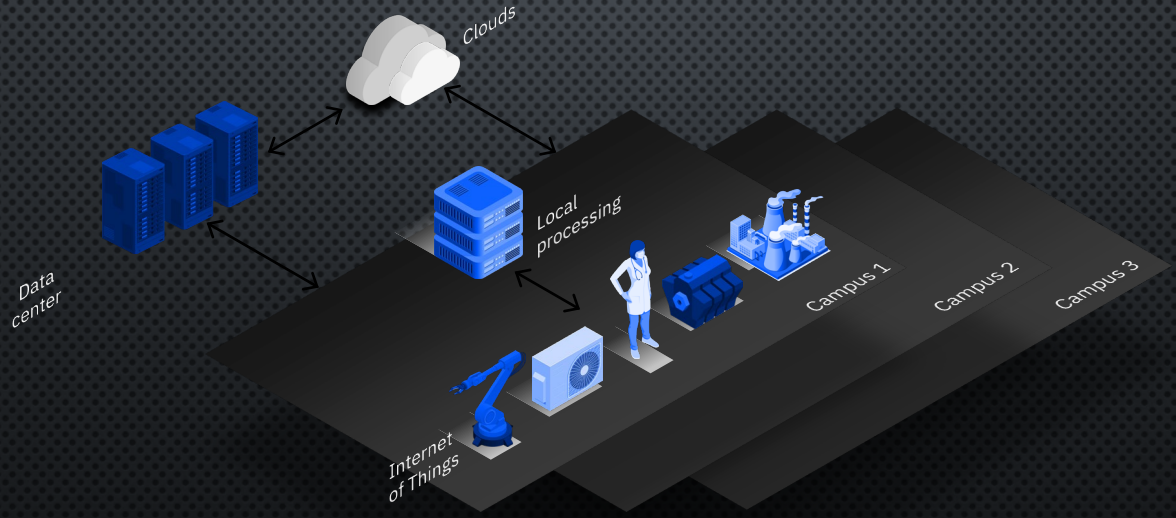
- Core:
  - GUI
    - Jupyter Notebook for Model Training
  - Core Services
    - Policy Management Web Application and DB
  - Python Libraries
    - REST Service to transfer Models Core to Edge
    - Image Model Training





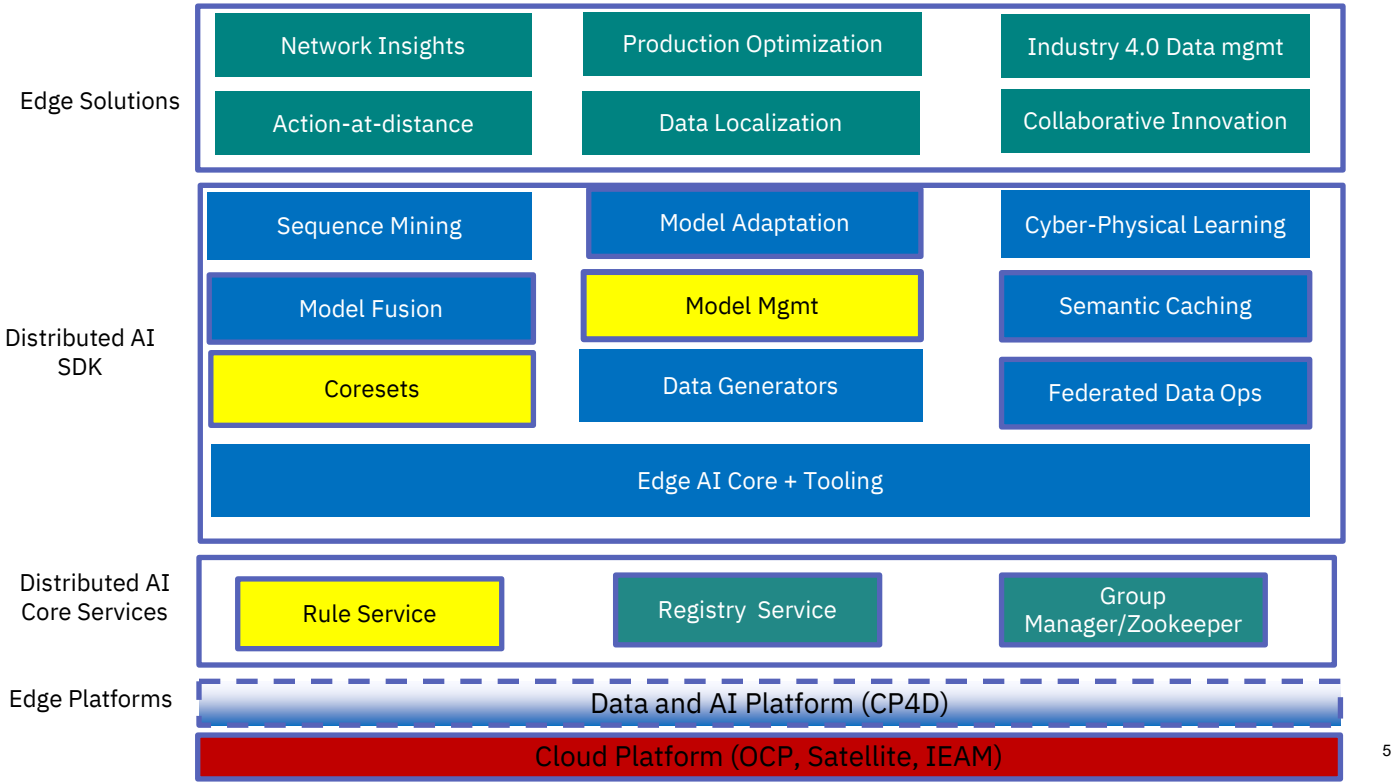
DISTRIBUTED AI SDK OVERVIEW

The **Distributed AI APIs** are early access offerings from IBM Research to enable AI in distributed environments. Distributed AI APIs are a set of **python libraries** with **data and AI algorithms** to support AI applications across **hybrid cloud and distributed and edge computing environments**. These APIs are general purpose and support many modalities of data, for example, visual, acoustic, sensors, network logs, time series, or natural language. Using these APIs, you can create AI-based applications in various industries.

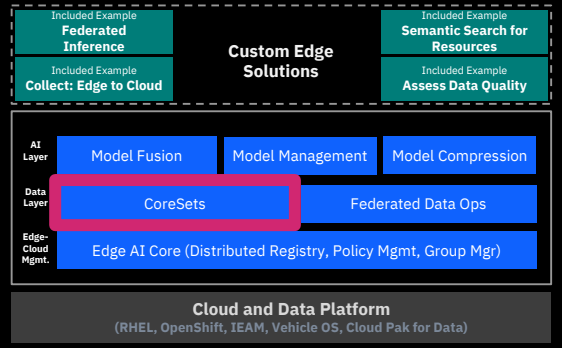


Distributed AI SDK - Libraries

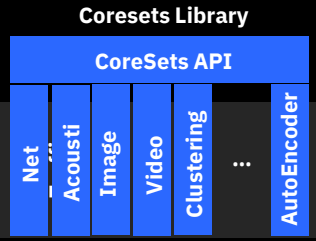
Included in Deliverables



The Distributed AI SDK – Core Set Module

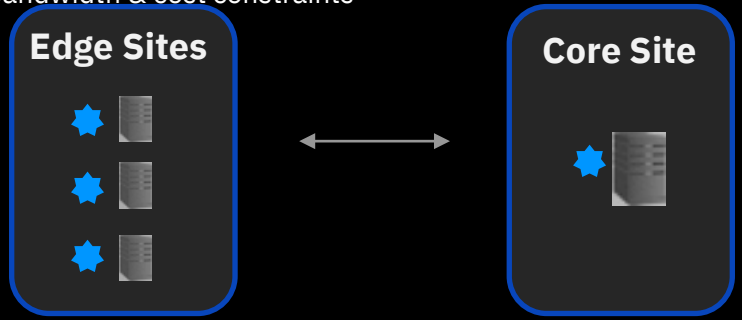


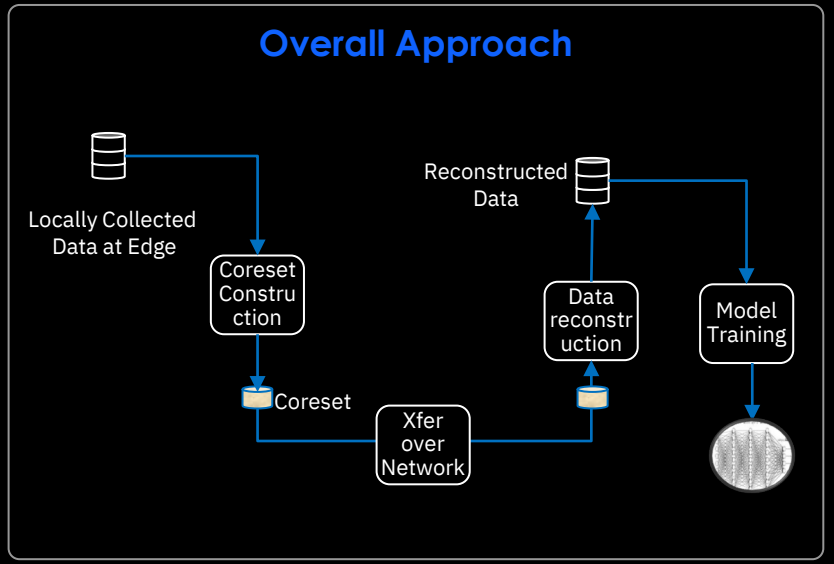
Pipeline-aware semantic data compression for data transfer between edge and core



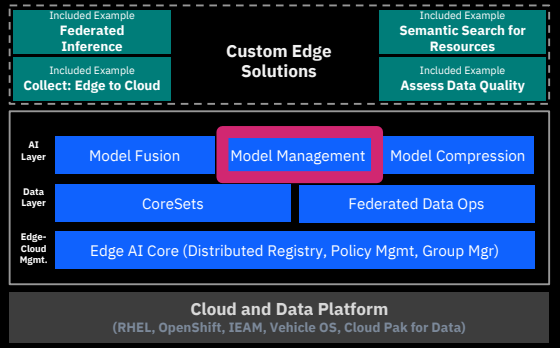
Q: How can we transfer data more efficiently between edges and the central location without degradation of model fidelity?

A: Data would need to be sub-sampled / compressed due to bandwidth & cost constraints





The Distributed AI SDK – Model Management



Manage the deployment of AI models specific the edge applications

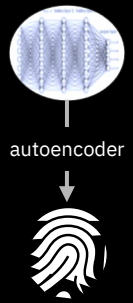
- Model fingerprints
- Model evaluation from a model collection
- Device-appropriate model adaptation
- Model performance monitoring

Model Fingerprinting

A *fingerprint* is a semantic representation of how a model's neurons respond to an input

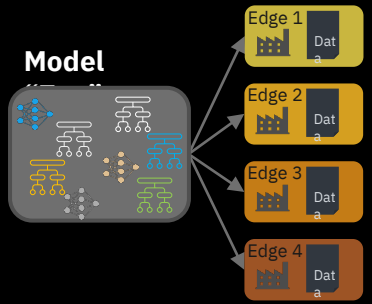
Train an autoencoder on layer activations

Reconstruction errors derived from test data become the fingerprint representation vector

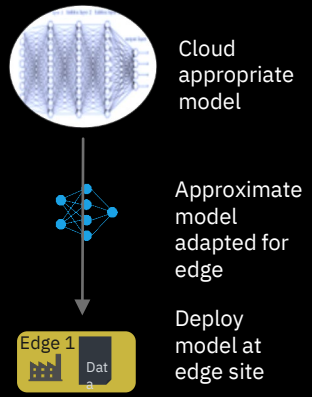


Model Selection

Select the best model to suit device & data using fingerprinting



Model Adaptation



Cloud appropriate model

Approximate model adapted for edge

Deploy model at edge site

Model Drift

detect, measure, & retrain if needed

- Data drift – caused by inconsistency between training and production data
- Concept drift – caused by changes to fundamentals linking predictors to targets
- Generalized – changes in quality metric not due to the above factors
- Monitor continuously to detect drift

Policy Management Library (PML)

Enable and accelerate the development of policy-based systems

Approach

A generic software library for authoring, management, administration, analysis and execution of policies that can be applied in real-time, leading to automated responses to system events

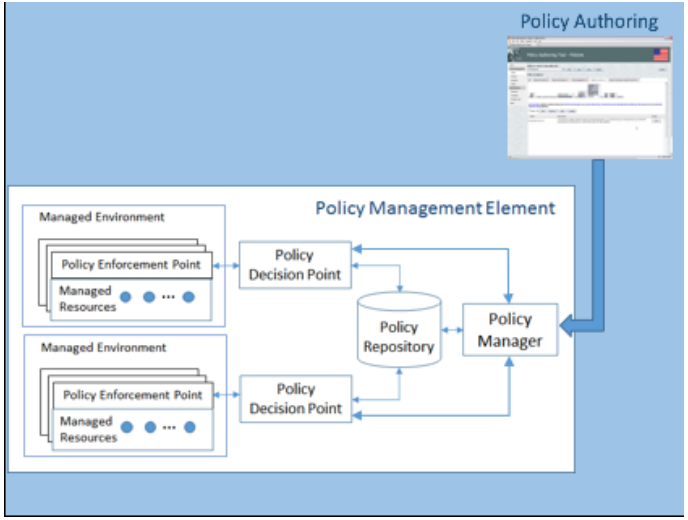
- Light weight, flexible set of tools for operational management
- Structured process for adaptation of policy management techniques to different domains through “resource models”
- Easily configured to support different architectures, e.g., Hierarchical or Federated policy managers

Advantages

- A Light Weight and generic software library & operational tool for the administration of policies, supporting the authoring, management, administration, analysis and execution of policies that can be applied in real-time
- Tailored for complex business processes with multiple/nested “if then” decision points; and for both Structured and Unstructured, dynamic data inputs
- Pre-integrated with our existing Analytic and future Cognitive libraries to drive the selection of policies to apply, and feedback insights from execution of policies to enable machine learning

Opportunities

- PML is a key differentiator in closing CPT and CAI deals for Robotic Process Automation and Cognitive Automation



Policy Manager - Administers policies in the managed runtime, Deploys policies from administrative repository, Remove/activate/deactivate policies in runtime

Policy Enforcement Point (PEP)- Decides when to acquire policy decisions, Provides input data to policy

Policy Decision Point (PDP) - Evaluates policies against context provided by PEP, Maintains registry of PEPs, Accesses deployed policies

Policy Repository - Generalized storage model for policies

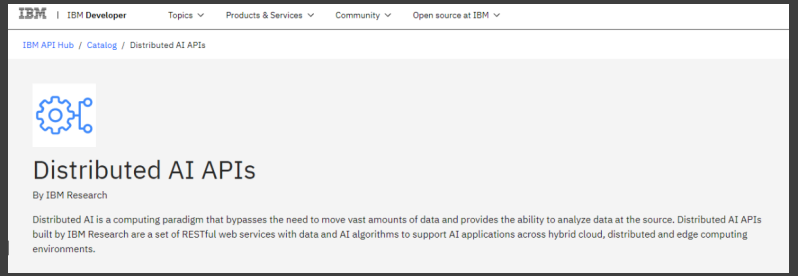
Distributed AI APIs on IBM API Hub

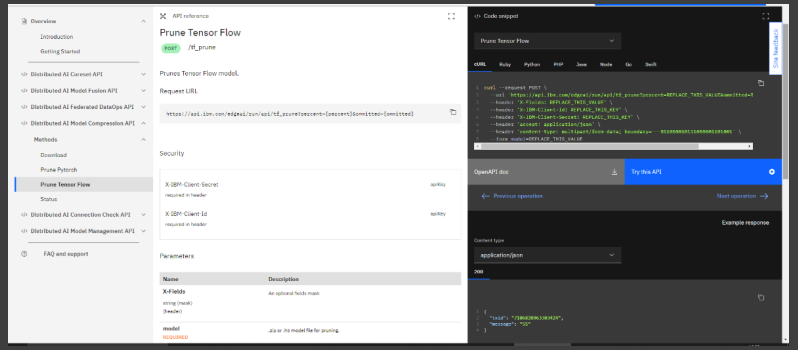
Free trial of key capabilities

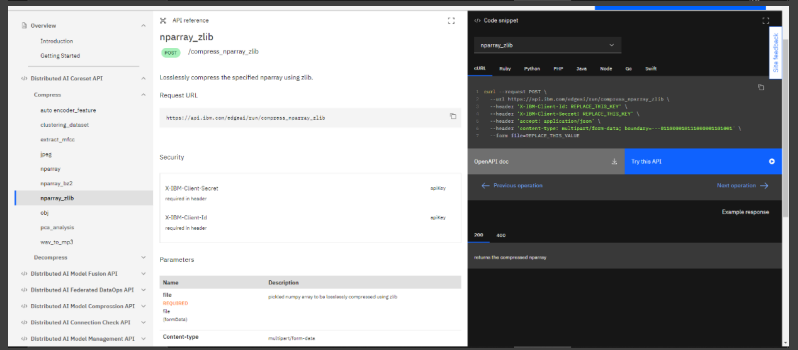
- Early access offerings from IBM Research to enable AI in **distributed environments**
- Support **many modalities of data**, for example, visual, acoustic, sensors, network logs, time series, or natural language
- Help **optimize data and model management** across cloud and distributed frameworks

https://developer.ibm.com/apis/catalog/edgeai--distributed-ai-apis/Introduction

Click in full screen or while holding the ctrl key







Demo Scenario

- Surveillance photos are captured at the Edge by a camera attached to a UAV.
- Since there is limited bandwidth from the Edge to the Cloud, the images are compressed before sending to the cloud.
- A model is trained in the Cloud and sent down to the Edge.
- After the image model is trained, the captured images are classified at the Edge.
- New images are determined to be Out of Distribution from original model
- Based on the classification results and outlier score, Policy will determine if the image model needs to be updated.
- When policy states that model is no longer current, compressed images will be sent to the cloud in order to update the model.

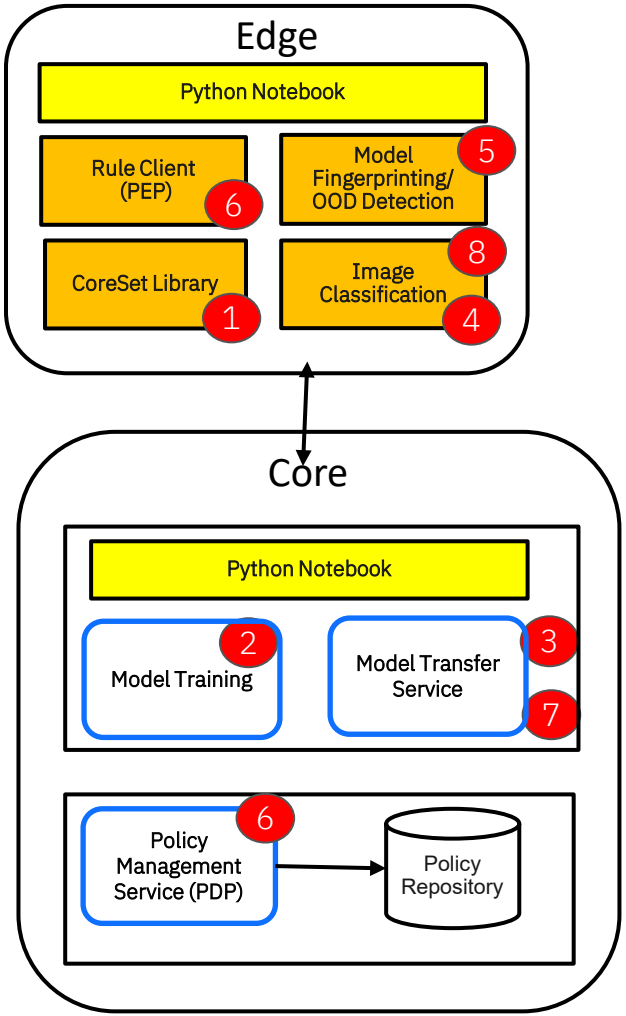
Demonstration Flow

Steps completed before demo:

1. Compressed images and features are sent to core
2. Train Image Model In Cloud Container
3. Model 1 sent to Edge Nodes

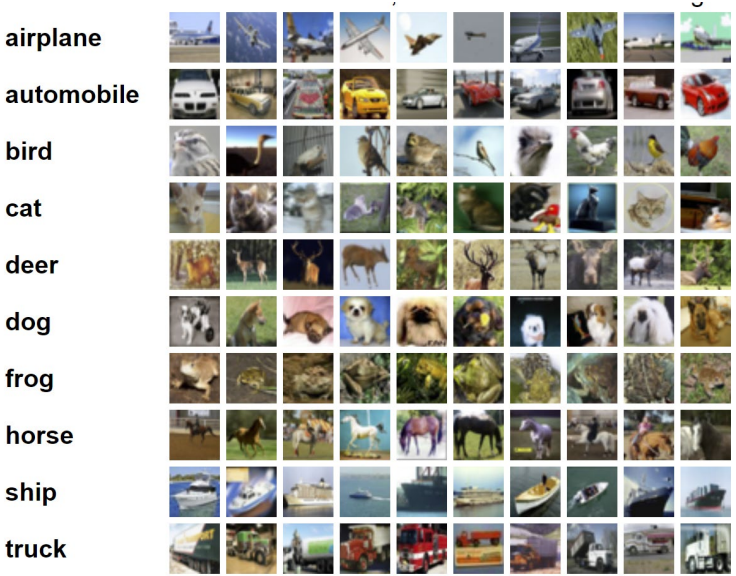
Steps completed live for demo:

4. Newly collected images are classified at Edge
5. Determine if images are Out of Distribution (OOD)
6. Use Policies to Determine if Model should be updated.
7. Push pre-trained Model 2 to Edge
8. New images are now classified correctly

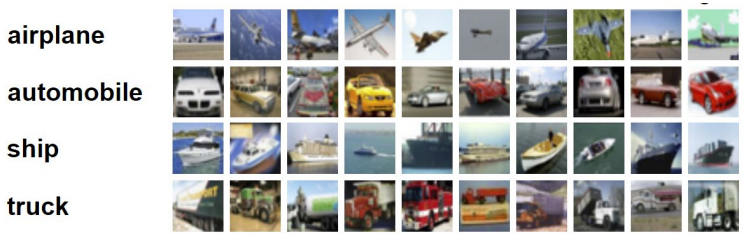


Demo Scenario: Cifar-10 Dataset

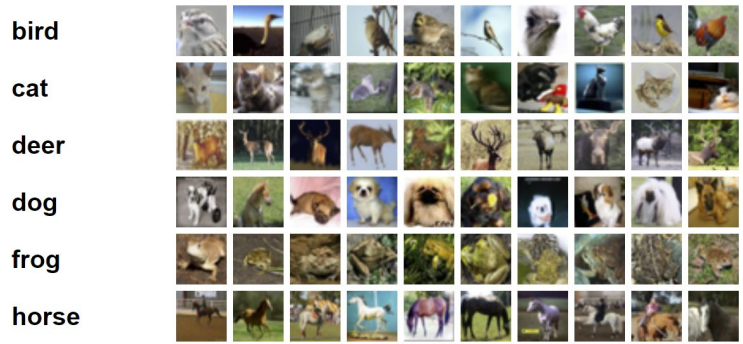
class_zero_model



class_one_model

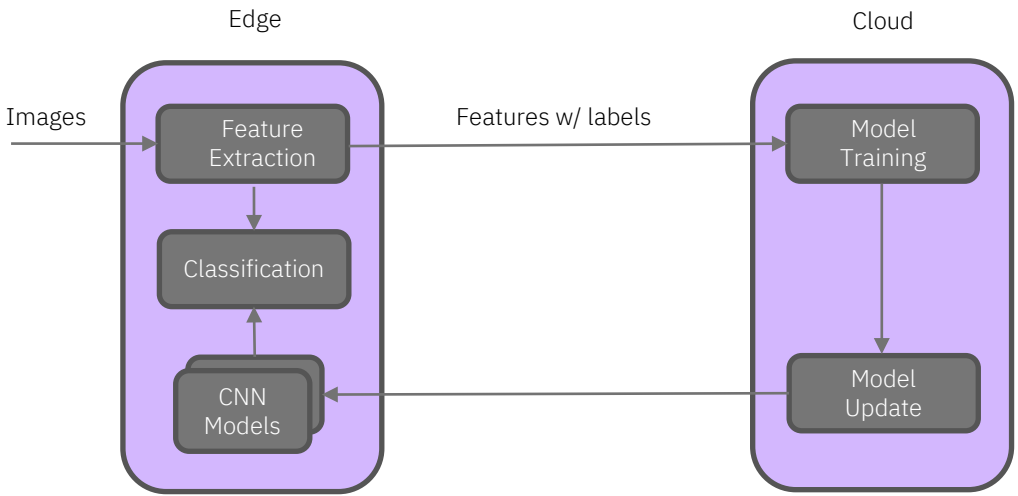


class_two_model



Demo Scenario: CNN Classifier

Image Training/Feature Extraction Components



Demonstration

Steps to Deploy and Run Containers

Edge

From Edge Folder <project root>/edge

1. Edit .env file to set ip address of Core Container
2. ./docker-build.sh
3. ./docker-run.sh
4. Access Notebook page: http://<host>:8888/
5. Enter notebook token “easy

Core

From Core Folder <project root>/core

1. ./docker-build.sh
2. ./docker-run.sh
3. Access Notebook page: http://<host>:8888/. (token = “easy”)

Rule Service

From Rule Service Folder <project root>/rule-service

1. ./docker-build.sh
2. ./docker-init.sh (Loads policies)
3. ./docker-run.sh
4. Access Policy Management Page: https://<host>:8443/WPML. (User name/password ibm/ibm)

Acknowledgment

”This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.”

Backup

Coreset Library

- Coresets Library consists of a set of compression algorithms enabling edge AI by efficient data transfer between edge and core. The primary purpose is creating an AI model when collecting training data from edge locations.
- API
  - Core sets configuration:

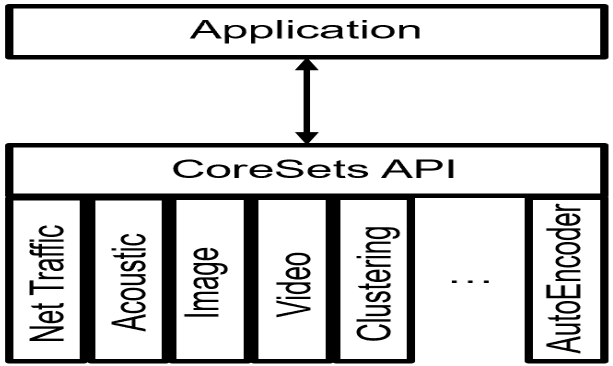
    xformation_description configure_coreset(dataset_description)
  - Data reduction

    transformed_data_set reduce_data_set(xformation_description, original_dataset)
  - Data expansion

    reconstructed_data_set expand_data_set(xformation_description, transformed_data_set)

- For streaming mode
  - API called on a batch repeatedly
- For network transfer
  - ***Xformation_description*** can be serialized and sent over the network and deserialized

- Coreset Library has many implementations of this API
  - Which can implement a variety of algorithms/approaches.

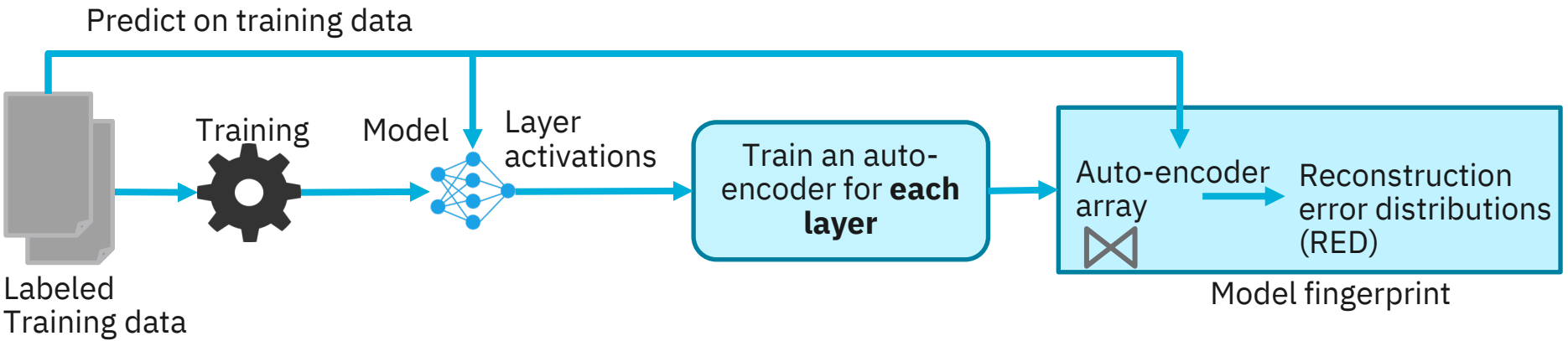


CoreSet - Approaches for Data Reduction

- **Lossless compression (Zip, Gzip, PNG, GIF, TIF, MNG, LZW)**
  - Remove redundant information in dataset
  - No change in model fidelity, but data reduction is not significant
- **Feature Extraction**
  - Use domain specific feature extractors (Can further Zip the extracted features)
  - High compression ratio
  - No change in model fidelity
- **Lossy compression**
  - Standard compression algorithms (e.g., MP3, JPEG, MP4)
  - Extract features from frequency domain/temporal domain and then encode them
- **Advanced algorithm**
  - **Principle component analysis**
    - Computing the principal components and using them to perform a change of basis on the data
  - **Clustering-based approach**
    - Represent each cluster by its centroid, a statistical distribution along the features
    - Recreate statistically similar data at the core location
  - **Variational Autoencoder**
    - An artificial neural network that learns the distribution inherent in the data and recreates statistically similar data at the core location
  - **Generative Adversarial Network (GAN)**
    - Two neural networks contest with each other in a game to learn to generate new data with the same statistics as the training set.
  - **Probabilistic Context Free Grammar**
    - Learn rules to recreate the data set present locally

Depending on data type and applications, an up-to-hundreds-to-one compression ratio can be achieved by coreset library

Step 1: Generate model fingerprints during training



Step 2: Measure likelihood of a model input being OOD given the RED during training