Financiado por la Unión Europea
NextGenerationEU

GOBIERNO DE ESPAÑA

Plan de Recuperación, Transformación y Resiliencia

White Paper: A Closed Control Loop for Edge-Cloud Collaboration using Neuro-Symbolic AI

09 Dec. 2025

Webpage: https://i2cat.net/project/6genablers-sp1-ai-for-6g-ultra-automation-

Executive Summary

The proliferation of latency-sensitive, resource-intensive edge applications, such as holoportation services, necessitates a paradigm shift from reactive to proactive resource management. This white paper details the architecture, components, and validation of a robust, closed control loop system designed for Edge-Cloud collaboration. The system leverages the OPTARE telemetry platform to feed real-time performance metrics to an i2CAT-developed Neuro-Symbolic (NeSy) prediction model managed via a Kubeflow MLOps pipeline. The resulting CPU usage prediction is seamlessly integrated with the OpenNebula orchestrator's OneFlow elasticity policies, enabling anticipatory scaling of the holoportation application stack. Critically, the use of NeSy AI addresses the prevalent challenge of black-box prediction, injecting interpretability and transparency into the automated resource allocation decisions, thereby minimizing the critical trade-off between resource over-provisioning and under-provisioning.

1. Introduction and Motivation

In future-proof 6G virtualized networks, the strategic allocation of Central Processing Unit (CPU) resources is fundamentally linked to overall network performance. Experimental evaluations have consistently demonstrated a non-linear relationship between factors like virtual RAN (vRAN) bandwidth and CPU utilization, confirming that network performance severely degrades without adequate computational capacity. Thus, achieving effective 6G network management requires strategic and coordinated CPU allocation, moving beyond simplistic rule-based systems.

Artificial intelligence and machine learning (AI/ML) offer the necessary tools to optimize CPU resources. However, the lack of transparency and interpretability in many Deep Learning models poses a significant barrier to their adoption in critical infrastructure like telecommunications. Informed resource allocation requires striking a delicate balance: i) Over-provisioning, which leads to unnecessary operational costs, and ii) Under-provisioning, which risks Service Level Agreement (SLA) violations and poor user experience.

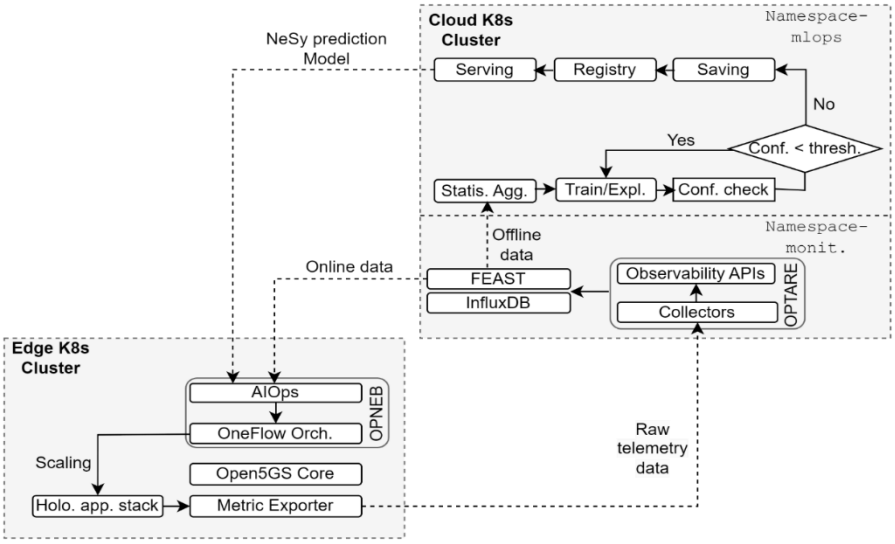
To address these limitations, the system adopts a Machine Reasoning (MR), or Neuro-Symbolic (NeSy), approach. This methodology is viewed as an advanced solution because it not only excels at prediction tasks but also ensures transparency through the embedded fuzzy logic, offering a demonstrable path toward proactive, cost-effective resource prediction that adapts fluidly to network dynamics.

2. System Architecture and Setup

The solution operates on a hybrid cloud infrastructure comprising two interconnected Kubernetes clusters, labeled Edge and Cloud. This setup is strategically designed to provide high-performance computation at the edge, while centralizing complex AI training and data processing functions in the cloud.

Edge Cluster Deployment

The Edge cluster is provisioned via OpenNebula OneKE, a cloud management tool that automates the deployment of virtualized infrastructures. Its purpose is to host the live, real-time services and network functions.



The Edge cluster topology consists of: i) One master node; ii) Two worker nodes; iii) One Virtual Network Function (VNF) node; and iv) One dedicated storage node.

The Holoportation App—the target application for autoscaling—is deployed on one worker node. A second worker node hosts the OPTARE Prometheus Push gateway for metrics collection.

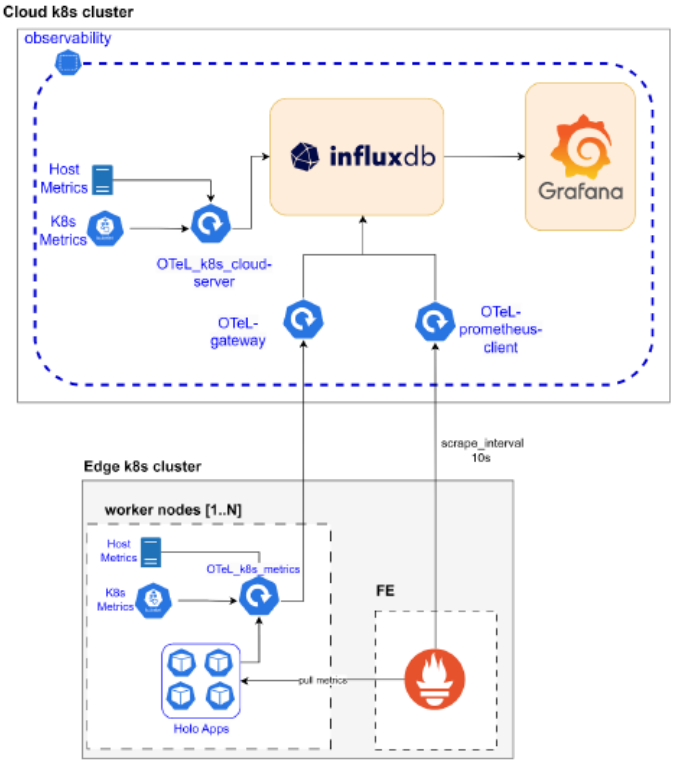
```
NAME                                    READY   STATUS    RESTARTS   AGE
continuous-image-puller-ct8sd           1/1     Running   0          146d
dummy-python-deployment-596b8856cc-cbzgn  1/1   Running   0          213d
feature-store-deployment-6d68687df9-8q78s  1/1  Running   0          20d
grafana-596db97574-g22kv                1/1     Running   0          91d
hub-64985cd7bf-jxl9p                    1/1     Running   0          146d
influxdb-65ddd88b9d-jcnzx               1/1     Running   0          21d
otel-gateway-5bfddb7595-wv2zb           1/1     Running   0          91d
otel-prometheus-58659ffc59-tbkct        1/1     Running   0          21d
postgres-8f47cdddb-ch6vd                1/1     Running   0          229d
proxy-5c8b7dc64b-kcvqc                  1/1     Running   0          146d
user-scheduler-6f9b5cb9b-7pc97          1/1     Running   0          146d
user-scheduler-6f9b5cb9b-7qrwd          1/1     Running   0          146d
```

Holoportation Stack Deployment

NAME	READY	STATUS	RESTARTS	AGE
api-server	1/1	Running	0	62d
cache-deployer-deployment-65fb47dd94-8hqgc	1/1	Running	0	197d
cache-server-657b5f8474-65tjk	1/1	Running	5 (196d ago)	197d
controller-manager-566cbcbc45-wkf7j	1/1	Running	0	197d
metadata-envoy-deployment-758c78ccb9-p752j	1/1	Running	0	197d
metadata-grpc-deployment-68d6f447cc-j7d42	1/1	Running	18 (196d ago)	197d
metadata-writer-6bf88bb8c4-4v4m8	1/1	Running	12 (196d ago)	197d
minio-59b68688b5-hnrbv	1/1	Running	0	197d
ml-pipeline-6fd7df65fd-v69l6	1/1	Running	22 (196d ago)	197d
ml-pipeline-persistenceagent-6f86458589-jtwkx	1/1	Running	14 (196d ago)	197d
ml-pipeline-scheduledworkflow-6d47f64655-jm7th	1/1	Running	0	197d
ml-pipeline-ui-59864db569-65c86	1/1	Running	0	197d
ml-pipeline-viewer-crd-c84f488f8-pt47k	1/1	Running	0	197d
ml-pipeline-visualizationserver-b688864fb-g84p8	1/1	Running	0	197d
mysql-5f8cbd6df7-zbmhm	1/1	Running	0	197d
proxy-agent-754f555d7c-66jb8	0/1	CrashLoopBackOff	55431 (103s ago)	197d
test-client	1/1	Running	1579 (5m3s ago)	65d
workflow-controller-7b46c9c84f-tg7sq	1/1	Running	0	197d

The holoportation application stack requires specific component deployment: i) The Comm Server is installed on the K8s Edge server. ii) Holo Clients (User Equipment) connect to the testbed, typically requiring only Ethernet connectivity. iii) The Holo Orchestrator, responsible for session management and control information, is supplied as a Docker image for deployment on the Edge server. iv) Data plane traffic is secured via TCP WebSockets, with bandwidth requirements per uplink/downlink stream.

Cloud Cluster Segmentation



The Cloud cluster is segmented into two distinct namespaces to enhance organizational control, resource isolation, and fault tolerance: i) MLOps Pipeline Namespace: Dedicated solely to the machine learning workflow, allowing for independent scaling and resource allocation based on model training and inference requirements. ii) Telemetry and Data Aggregation Namespace: Dedicated to receiving and processing metrics data, centered around the InfluxDB time-series database. Inter-namespace communication is strictly managed through Kubernetes APIs and Service Discovery mechanisms, using Kubernetes Services for data exchange and network policies to ensure secure and efficient traffic flow while maintaining isolation.

3. Closed Control Loop Components

The closed control loop integrates three foundational pillars: the telemetry collection system, the AI prediction service, and the resource orchestration layer.

3.1 The Telemetry System (OPTARE & FEAST)

The OPTARE telemetry platform forms the data foundation of the loop. It collects performance metrics from the Edge-deployed holoportation stack via an exporter, forwarding this raw data to the Cloud environment.

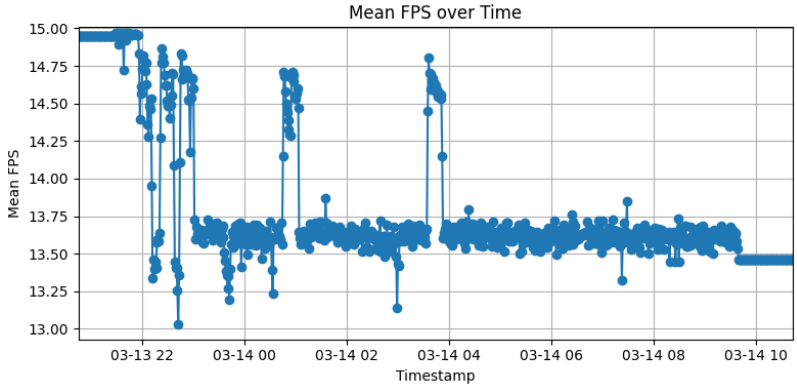
Edge Components for Collection: i) Cert Manager: Responsible for managing security certificates. ii) Otel Operator: Manages telemetry data collection using the OpenTelemetry SDK Client. iii) K8s Cluster Metrics Collector: Collects fundamental metrics (CPU, memory, network) from the Edge Kubernetes cluster. iv) App Metrics Collector: Collects custom metrics from the Holotransportation applications (e.g., frames per second, latency).

Cloud Components for Processing and Storage: The telemetry data is transmitted from the Edge to the Cloud (via) where it is managed by the following components: i) otel-gateway: Receives and routes the data to observability components. ii) InfluxDB2 and Grafana: Store and visualize the time-series data. iii) ETL system (Apache Airflow): Processes the data, performing extraction, transformation, and loading into the data warehouse. iv) Data Warehouse (PostgreSQL): Provides structured storage for processed data. v) Feature Store (FEAST): Aggregates and organizes the cleaned metrics, facilitating efficient management of data features crucial for both offline training and online inference. This ensures consistency in model behaviour across different operational stages, exposing a REST API for consumption.

```
query = f'''
from(bucket: "{config['bucket']}")
|> range(start: {timeRangeStart}, stop: {timeRangeStop})
|> filter(fn: (r) =>
       r["_measurement"] == "RGBD_latency_ms" or
       r["_measurement"] == "fps")
|> filter(fn: (r) => r["_field"] == "gauge")
|> filter(fn: (r) => r["exported_job"] =~ {config['holo_app_regex']})
|> aggregateWindow(every: {config['window_period']}, fn: mean, createEmpty: false)
|> yield(name: "mean_rgbd_fps")

from(bucket: "{config['bucket']}")
|> range(start: {timeRangeStart}, stop: {timeRangeStop})
|> filter(fn: (r) => r["_measurement"] == "sessions_count")
|> filter(fn: (r) => r["_field"] == "gauge")
|> aggregateWindow(every: {config['window_period']}, fn: mean, createEmpty: false)
|> yield(name: "mean_sessions_count")
'''
```

	result	table	_time	_value	_field	_measurement	app
0	mean_sessions_count	0	16:17:00+00:00	2.000000	gauge	sessions_count	orchestrator-node
1	mean_sessions_count	0	16:17:00+00:00	2.000000	gauge	sessions_count	orchestrator-node
2	mean_rgbd_fps	0	16:17:00+00:00	3.000000	gauge	RGBD_latency_ms	NaN
3	mean_rgbd_fps	0	16:17:00+00:00	3.000000	gauge	RGBD_latency_ms	NaN
4	mean_rgbd_fps	1	16:17:00+00:00	14.811232	gauge	fps	NaN
5	mean_rgbd_fps	1	16:17:00+00:00	14.811232	gauge	fps	NaN
6	mean_rgbd_fps	2	16:17:00+00:00	15.017758	gauge	fps	NaN
7	mean_rgbd_fps	2	16:17:00+00:00	15.017758	gauge	fps	NaN



3.2 The NeSy Prediction Service (i2CAT MLOps)

The i2CAT solution provides a Neuro-Symbolic-empowered CPU prediction service that allows the orchestrator to make anticipatory scaling decisions. This service is managed and continuously delivered via a Kubeflow MLOps pipeline.

NeSy Concept and Transparency

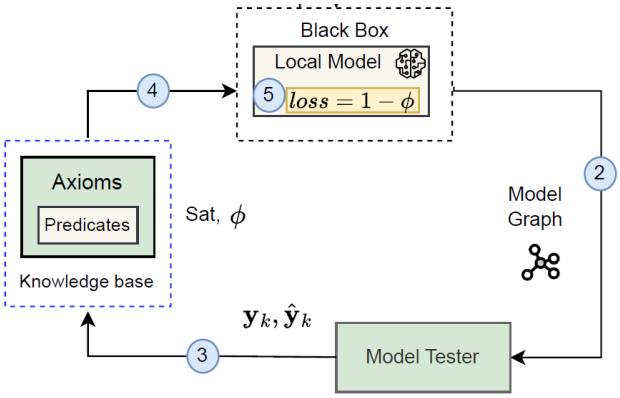
NeSy learning systems represent a unified approach, fusing the structured data proficiency of symbolic systems with the ability of neural systems to learn from unstructured data. This system utilizes Logical Tensor Networks (LTN), a framework that integrates neural and symbolic methodologies for learning and reasoning using first-order fuzzy logic semantics.

The core innovation is the integration of symbolic knowledge (predicates and axioms) into the training process, enhancing model interpretability. Transparency is achieved because

the LTN model's predictions are constrained by logical rules encoded in the Knowledge Base, aligning the AI's decision-making process with human cognitive reasoning, which is essential in dynamic telecom environments.

Formulation using Logical Tensor Networks (LTN)

The knowledge base employs a specialized predicate and a logical axiom to guide the model training:

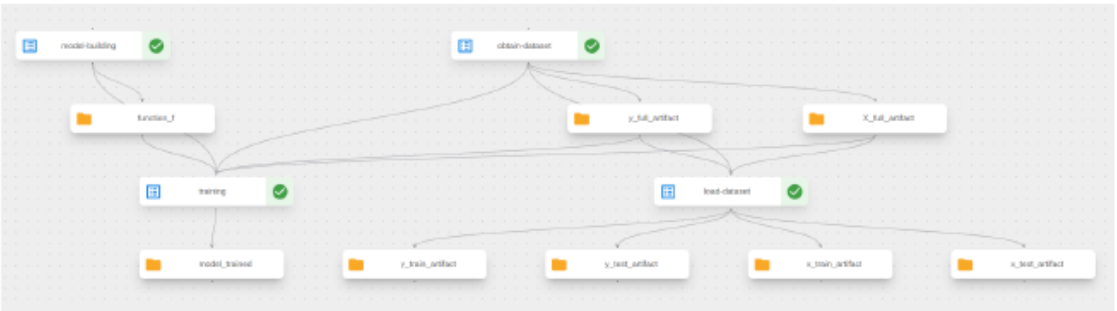


Predicate: Defined as a "smooth" variant of the equality symbol, the predicate is given by . Here, is the predicted value and is the observed value, and is a positive constant governing the smoothness. This formulation guarantees outputs within the range, allowing flexible matching between predicted and observed CPU usage.

Axiom: The logical statement is expressed using universal quantification () and a diagonal operator (): This axiom asserts that for all input features, the equality predicate should hold between the ground truth and the regressor function's prediction. The training process iteratively minimizes the loss function, which is a transformation of the Satisfaction Level of this axiom. Minimizing loss maximizes the model's adherence to the logical rule, ensuring robustness and explainability.

Distributed MLOps Kubeflow Pipeline Deployment

The MLOps pipeline is containerized using Kubeflow and deployed within its dedicated namespace in the Cloud cluster. The pipeline consists of: i) Statistical Data Aggregation, ii) Model Training, Confidence Evaluation and Saving, and iii) Model Serving and Deployment.

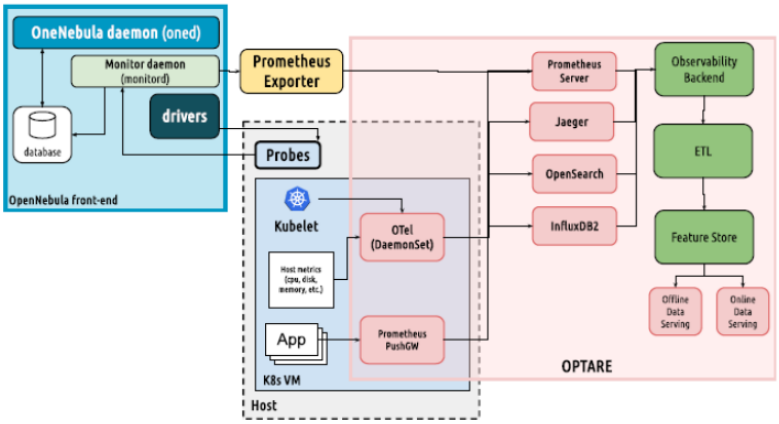


Crucially, access to the telemetry data is secured: i) A Service Account (feast-access) is created in the mlops namespace. ii) A Role Binding is established to grant this service account permissions to access data in the monitoring namespace, where FEAST is

deployed. iii) The pipeline scripts are updated to use the appropriate service account and access the FEAST data via its internal service address (`feast.svc.cluster.local:30101`).

3.3 The OpenNebula Orchestrator Integration

The final stage involves retrieving the trained and validated NeSy model and encapsulating it as a prediction service for the OpenNebula orchestrator.



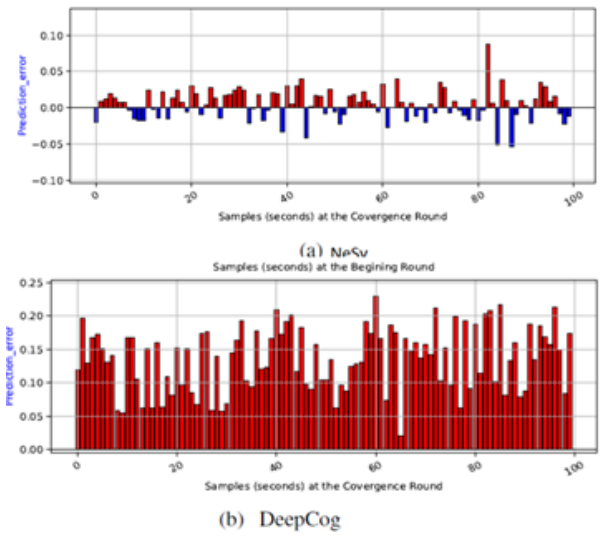
Orchestrator Components: The orchestrator relies on the OpenNebula front-ends running on Ubuntu 22.04, providing core services including the management daemon, Scheduler, Sunstone, FireEdge, MariaDB, and the crucial elasticity engine, OneFlow and OneGate.

Closed-Loop Execution via PyONE: The entire autoscaling solution is implemented as a Python program that acts as the control loop. It leverages OpenNebula's Python bindings (PyONE) to interact with the orchestration layer. The workflow is executed periodically: i) Data Retrieval: The Python program retrieves real-time performance metrics from the OPTARE's FEAST feature store via its HTTP endpoint. ii) Prediction: The i2CAT NeSy prediction service is called to forecast the optimal CPU usage (). iii) Scaling Injection: The predicted value is injected into the OneKE OneFlow service templates to dynamically trigger the configured elasticity policy. A complementary Model Updater service ensures the prediction service always uses the latest, most confident version of the ML model. This integration allows for proactive, cost-efficient scaling of OneFlow roles. For example, the opposite elasticity policy dictates a scaling action based on the predicted CPU value. This demonstrates that the system is configured to expand the cluster by two worker nodes whenever the predicted CPU usage exceeds a threshold for three consecutive 10-second intervals, moving scaling decisions from reactive to predictive.

```
"elasticity_policies": [
{
    "expression": "ESTIMATED_CPU_USAGE > 191900",
    "type": "CHANGE",
    "adjust": 2,
    "period_number": 3,
    "period": 10
  }
]
```

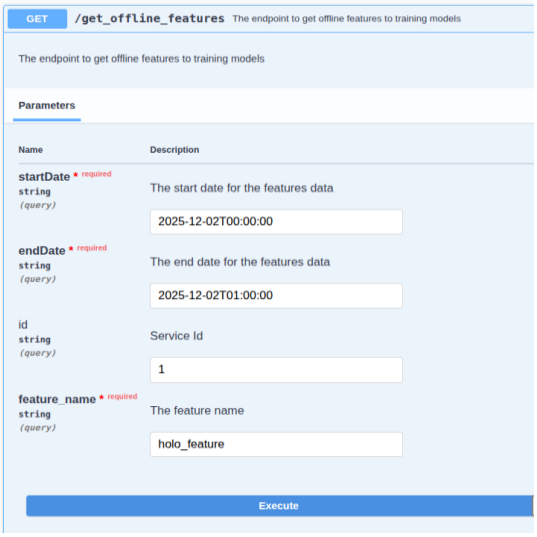
4. Validation and Results

The validation testbed demonstrated the dynamic resource optimization capabilities of the hybrid cloud infrastructure. The Edge cluster hosts the Holoportation App and metrics collector, while the Cloud cluster centralizes data storage (InfluxDB), processing (Airflow, PostgreSQL), feature engineering (FEAST), and the AI module (Kubeflow).

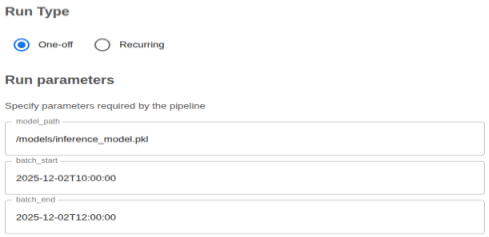


Over-provisioning and Under-provisioning Trade-off Analysis

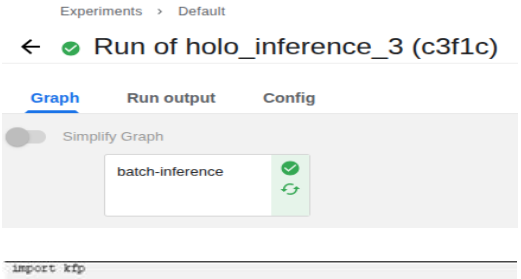
The NeSy model, implemented as a Multi-Layer Perceptron (MLP) within the Logical Tensor Networks (LTN) framework, was trained on a time-series dataset from the offline FEAST. The proposed NeSy model exhibits superior performance, closely tracking the actual CPU load. This superior tracking ability is attributed to the model's distinctive features: i) its neuro-symbolic design for interpretability, and ii) its ability to adaptively manage CPU demand fluctuations. The most critical validation point is the model's ability to manage the over/under-provisioning trade-off. As illustrated in the opposite figure, the red segment signifies over-provisioning (predicted resources exceeding actual requirements). ii) The blue segment signifies under-provisioning (insufficiently allocated resources, risking SLA violation). The results demonstrate that the NeSy model effectively balances this trade-off, minimizing both red and blue segments. This performance dramatically outperforms the DeepCog baseline model by a factor of 6, as the baseline consistently exhibits high over-provisioning. The NeSy model's constraint-guided training (via the logical axiom) ensures a more optimal and efficient resource prediction profile.



Closed-Loop Execution Evidence



During the training stage, the NeSy model learns to associate system conditions with optimal scaling behaviours for the holoportation workload. The model is trained on historical telemetry that reflects a wide range of operating states, preparing it to handle sudden changes in XR traffic patterns, fluctuations in delay or throughput, and variable compute availability. Once training concludes, the confidence checking initiates, during which the newly trained model is evaluated to ensure that its predictions meet the performance targets established in earlier work packages. Confidence verification may involve comparing the model's accuracy against validation datasets, monitoring loss trends, or detecting potential data drift. Only when the model satisfies these criteria, ensuring that it can operate reliably under current telemetry patterns, does it proceed to deployment. This evaluation step is essential to prevent sub-optimal or unstable models from influencing Edge-side resource management. In the final stage, the validated NeSy model becomes an exposed model, where it is registered and deployed as a live prediction service. Furthermore, this inference server can be customized to use different models, depending on the type of data that will be used. The final step validated the end-to-end integration and the automated actuation of the scaling policy. The Python interface, using PyONE, successfully connected to the OpenNebula XML-RPC endpoint and updated the worker node's VM template based on the predicted CPU allocation. The use of the PyONE binding to call the API function was confirmed to modify the CPU attribute in the VM's template. As shown below, the predicted CPU change was successfully applied at the node level, validating the completion of the closed control loop from edge data collection to cloud-based predictive decision-making and edge infrastructure reconfiguration.




```
import kfp

client = kfp.Client(host="http://ml-
pipeline.kubeflow.svc.cluster.local:8888")

experiment = client.create_experiment(name="holo-batch-inference")

run = client.create_run_from_pipeline_func(
    holo_inference_pipeline,
    arguments={
        "model_path": "/models/dummy_model.pkl",
        "batch_start": "2025-12-03T10:00:00",
        "batch_end": "2025-12-03T11:00:00",
    },
    experiment_name=experiment.name,
)

print("Started run:", run.run_id)
```

```
Every 2.0s: onevm list                                  sna11: Wed Nov 26 14:19:03 2025

  ID USER     GROUP    NAME              STAT  CPU     MEM HOST             TIME
  37 oneadmin oneadmin worker_3_(servic runn  2.5      3G 172.27.13.2    7d 01h53
  36 oneadmin oneadmin worker_2_(servic runn    2      3G 172.27.13.2    7d 02h03
  35 oneadmin oneadmin cloned_master_vm poff    2      3G 172.27.13.2   22d 23h09
```

```
ID USER     GROUP    NAME              STAT  CPU     MEM HOST             TIME
37 oneadmin oneadmin worker_3_(servic runn  1.5      3G 172.27.13.2    7d 01h57
36 oneadmin oneadmin worker_2_(servic runn    2      3G 172.27.13.2    7d 02h08
35 oneadmin oneadmin cloned_master_vm poff    2      3G 172.27.13.2   22d 23h14
```

Team

Hatim Chergui, *i2CAT Foundation, Spain*
Juan Sebastián Camargo, *i2CAT Foundation, Spain*
Shuaib Siddiqui, *i2CAT Foundation, Spain*
Guillermo Candela Belmonte, *OPTARE Solutions, Spain*
Alberto González Barneo, *OPTARE Solutions, Spain*
Daniel Garcia Fernandez, *OpenNebula Systems, Spain*

This White Paper received support from MINECO (Spain) and the EU NextGeneration EU/PRTR through the UNICO I+D 5G 2021 Call, with reference TSI-063000-2021-10 (6GENABLERS-AI).