



# AI WORKLOADS AT SCALE

*Kubernetes Cluster with Supermicro's Systems with AMD EPYC™ 7002 Series processors*

## TABLE OF CONTENTS

- Executive Summary..... 1
- System Configuration..... 2
- Introduction to Kubernetes..... 4
- Kubernetes Cluster Deployment..... 4
- Scale-up With Kubernetes Cluster..... 8
- Conclusion ..... 9



*Supermicro AMD based WIO System*

## Executive Summary

The Deep Learning (DL) benchmark results in the previous white paper<sup>1</sup> clearly show that a DL workload in Docker containers performs the same as on a BareMetal system. Building an on-prem Kubernetes cluster with GPU workers and AI framework-specific Docker containers can help organizations run projects or productions in a highly reliable and scalable platform. This white paper introduces Supermicro AMD based WIO systems, AS-1114S-WTRT, as Kubernetes Admin and master nodes. Along with AS-2023US-TR4, we build an NVIDIA GPU capable Kubernetes cluster that uses Cloud-native CEPH storage as persistent volumes and demonstrates how a DL workload can scale the Kubernetes cluster.

## SUPERMICRO

Supermicro (Nasdaq: SMCI), the leading innovator in high-performance, high-efficiency server and storage technology is a premier provider of advanced server Building Block Solutions® for Enterprise Data Center, Cloud Computing, Artificial Intelligence, and Edge Computing Systems worldwide. Supermicro is committed to protecting the environment through its “We Keep IT Green®” initiative and provides customers with the most energy-efficient, environmentally-friendly solutions available on the market.

<sup>1</sup> White paper: [SUPERMICRO® SYSTEM COMBINES AMD EPYC™ PROCESSORS AND NVIDIA GPUS TO ACHIEVE CONSISTENT DEEP LEARNING PERFORMANCE WITH LINEAR SCALING](#)

## System Configuration

AS-1114S-WTRT is one of the Supermicro AMD EPYC™ 7002 series based WIO series servers, offering a wide range of I/O options to deliver truly optimized systems for specific requirements. For more detailed system information, please go [HERE](#). Customers can optimize the storage and networking alternatives to accelerate performance, find the perfect fit for their applications. In our case, it uses the VMWare host and Kubernetes master nodes. Figure 1 and Figure 2 provide an overview of the system:

Table 1 provides a sample configuration for AS-1114S-WTRT

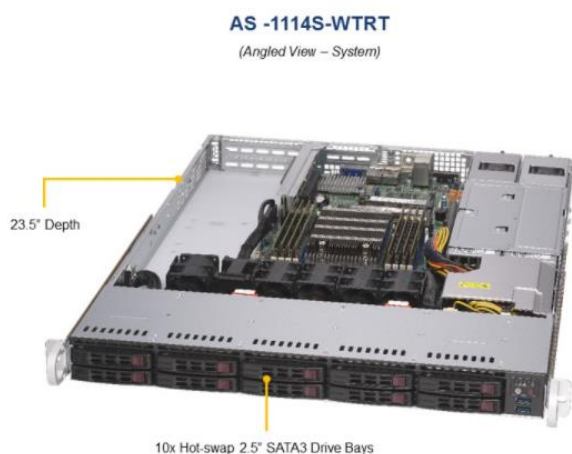


Figure 1

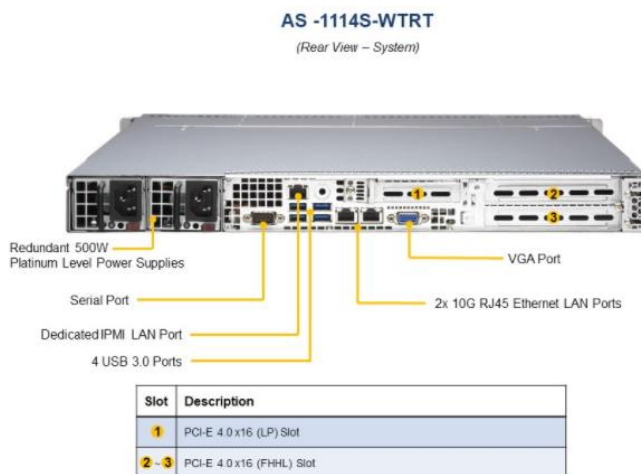


Figure 2

System	Part Number	QTY	Part Description
Infrastructure	AS-1114S-WTRT	1	H12 SSW-NT,CSV116TS-R504WBP
CPU	PSE-ROM7552	1	AMD EPYC™ 7552 DP/UP 48C/96T 2.2G 192M 200W 4094
Memory	MEM-DR416L-HL01-ER32	8	16GB DDR4-3200 2Rx8 ECC REG DIMM
HDD/SSD (Storage)	HDS-X2A-XS7680TE70004	2	[NR]Seagate Lange 7.68TB SAS 12Gb/s, 15mm, 2.5", 0.8DWPD SSD, HF
HDD/SSD (OS)	HDS-SMN1-MZ1LB3T8HMLA07	2	Samsung PM983 3.84TB NVMe PCIe3x4 V4 M.2 22x110mm (1.3 DWPD)
AOC	MCX4121A-ACAT	2	Standard Low-profile Mellanox 25GbE card with 2x SFP28 ports
Storage Controller	AOC-S3008L-L8i	1	AOC-S3008L-L8i Retail Pack
Cable	CBL-SAST-0593	1	Internal Mini-SAS HD to Mini-SAS HD 60cm,30AWG,12Gb/s
Software License	SFT-DCMS-SINGLE	1	Supermicro System Management Software Suite node license, HF, RoHS/REACH, PBF

Table 1

Table 2 provides cluster system roles and components. For more information on solution reference architectures, which consists of different level bundle configurations with various software vendors depending on the customers' workloads, refer to our team's solution page <https://verticalsolutions.supermicro.com> for more information.

System	Role	QTY	Software
AS-1114S-WTRT WIO server	Virtual host with Kubernetes admin node	1	VMWare 6.5
AS-1114S-WTRT WIO server	Kubernetes Master nodes	3	Kubernetes 1.19
AS-2023US-TR4 with 2xV100 32GB PCIe GPU Per Node	Kubernetes Worker nodes	3	Kubernetes 1.19 Nvidia-docker
SSE-X3648T	Network Switches TOR	2	SMCI OS

Table 2

Figure 3 shows the reference architecture.

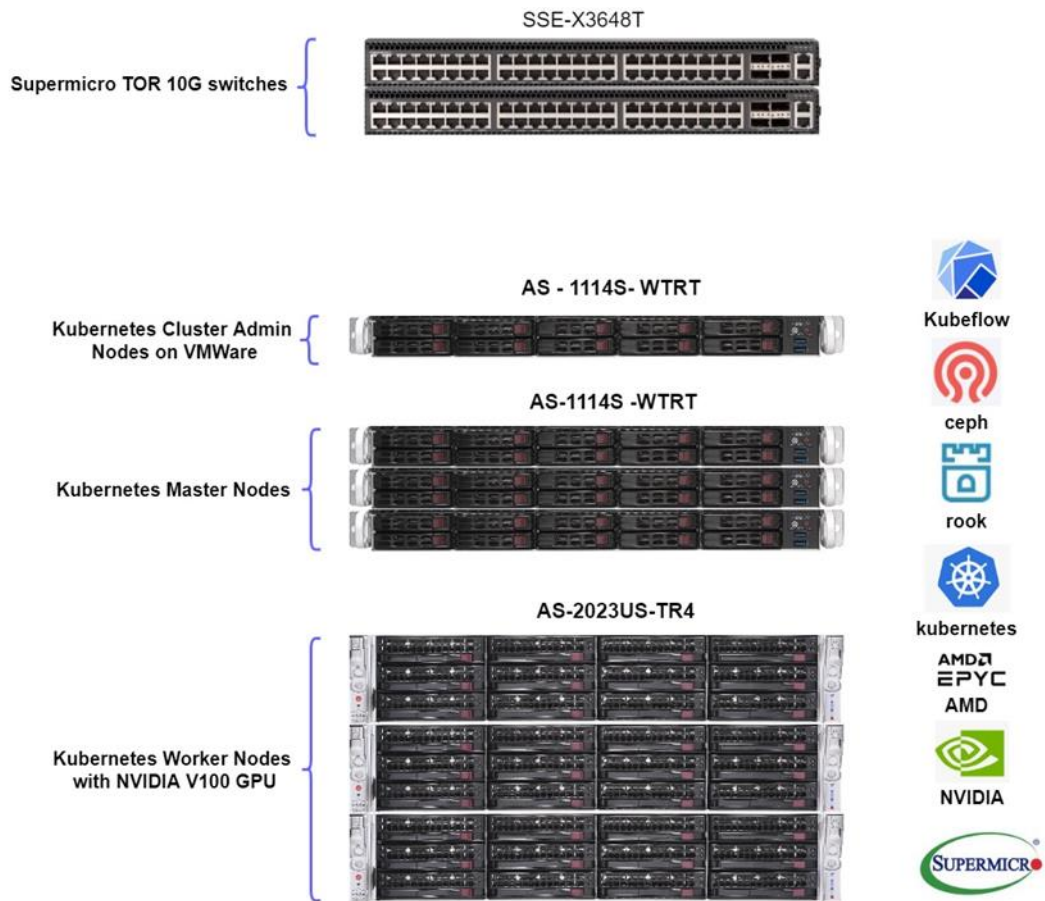


Figure 3

## Introduction to Kubernetes

Kubeflow is a machine learning toolkit that runs on Kubernetes by abstracting machine learning solutions and best practices utilizing the features of Kubernetes. It makes deployments of machine learning workflows portable and scalable and runs in distributed environments. Kubeflow's core and ecosystem critical user journeys (CUJs) provide software solutions for end-to-end workflows to build, train, deploy, and/or develop a model and create and run a workflow pipeline. Details regarding Kubernetes are at: <https://www.kubeflow.org/docs/started/kubeflow-overview/>

The AI/ML applications can be deployed in either a single node cluster or multi-node clusters, with each node loaded with multiple GPUs. Single node clusters can be used to build proof of concept solutions and smaller-scale training tasks utilizing multiple GPUs. The single-node clusters are usually used in development environments and for non-production use. Multi-node clusters are used in production deployments as they provide high availability, significant improvement in speed of execution, and efficient scalability.

## Kubernetes Cluster Deployment

Making GPU work with the Kubernetes cluster on-prem can be a complicated task. Therefore, NVIDIA has developed a tool named DeepOps, to automate Kubernetes cluster provisioning, configuring, and scaling. Further information regarding DeepOps can be found [here](#).

Figure 4 outlines the process to prepare and deploy the NVIDIA GPU cluster on-prem.



Figure 4

The tool is based on two open-source projects: Ansible and Kubespray, which are very flexible and scalable. It can deploy the Kubernetes cluster onto a single GPU node or hundreds of systems except for the cluster-admin node. Once DeepOps is installed on the admin node, modifying the inventory file according to the design. Figure 6 is the inventory file located under the deepops/config directory. In addition, the k8s-cluster.yml in the deepops/playbooks provides cluster options.

```
# Also add mgmt/master nodes here if they will run non-control plane jobs
[kube-node]
gpu2
gpu3
#gpu01
#gpu02

[k8s-cluster:children]
kube-master
kube-node

#####
# SLURM
#####
[slurm-master]
#login01

[slurm-node]
#gpu01
#gpu02

[slurm-cluster:children]
slurm-master
slurm-node

#####
# SSH connection configuration
#####
[all:vars]
# SSH User
#ansible_user=ubuntu
#ansible_ssh_private_key_file=~/.ssh/id_rsa'
# SSH bastion/jumpbox
#ansible_ssh_common_args='-o ProxyCommand="ssh -W %h:%p -q ubuntu@10.0.0.1"'
ubuntu@deepops:~/deepops/config$
```

Figure 5

To set up the Kubernetes cluster with GPU installed worker nodes, run the following command:

```
ansible-playbook --limit k8s-cluster playbooks/k8s-cluster.yml -k -K3
```

<sup>2</sup> Link to the k8s-cluster.yml file <https://drive.google.com/file/d/1EZLH4lfp1IVtaJWhPOvcjvNyut7L5ady/view?usp=sharing>

<sup>3</sup> More instructions can be found in <https://verticalsolutions.supermicro.com> and [HERE](#)

After the installation process is completed, the Kubernetes cluster should be up and running. To validate the cluster, type the following command on the admin node: `kubectl get node`. Figure 6 shows the output.

```
ubuntu@deepops:~$ kubectl get node
NAME      STATUS   ROLES    AGE   VERSION
gpu2     Ready   master   56d   v1.18.9
gpu3     Ready   <none>   56d   v1.18.9
ubuntu@deepops:~$ kubectl get namespace
NAME                STATUS   AGE
cert-manager        Active   56d
cluster-gpu-verify  Active   13m
default             Active   56d
gpu-operator-resources  Active   56d
istio-system        Active   56d
knative-serving     Active   56d
kube-node-lease     Active   56d
kube-public         Active   56d
```

Figure 6

Figure 7 shows the overall architecture discussed in this paper.

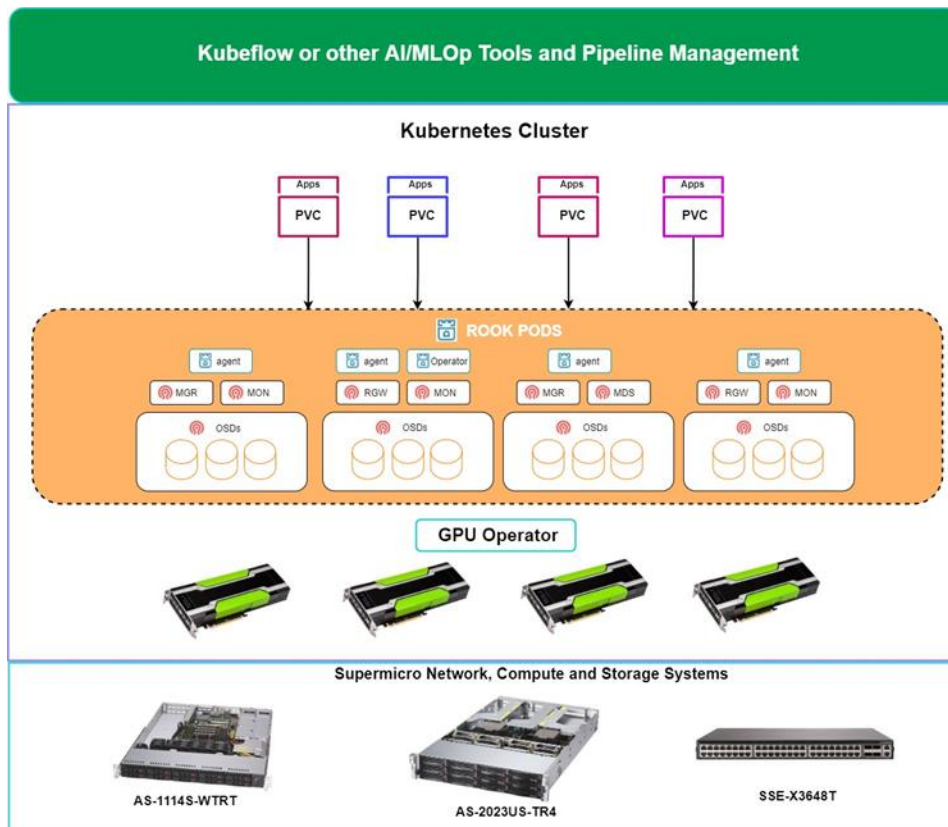


Figure 7

NVIDIA DeepOps is a handy open-source tool to deploy GPU Cloud-Native clusters. The tested environment consists of one VM as a deployment host and two AS-2023US-TR4 as GPU nodes. You can find the detailed features and configurations [here](#).

The tool also installs and configures a Cloud-native certified rook-ceph CEPH storage system as the cluster's persistent volumes, which is a must for all Deep Learning workloads since the datasets and intermediate training results have to be shared among the GPU nodes. Figure 8 and Figure 9 show the dynamic CEPH StorageClass information for the Kubernetes cluster.

```
ubuntu@deepops:~$ kubectl get pods -n rook-ceph
NAME                                READY   STATUS    RESTARTS   AGE
csi-cephfsplugin-4xghp              3/3    Running   0           56d
csi-cephfsplugin-b7hcr              3/3    Running   0           56d
csi-cephfsplugin-provisioner-68b5fb6499-m2tsf  4/4    Running   0           56d
csi-cephfsplugin-provisioner-68b5fb6499-x5z5k  4/4    Running   1           56d
csi-rbdplugin-btsp4                 3/3    Running   0           56d
csi-rbdplugin-dhgq9                 3/3    Running   0           56d
csi-rbdplugin-provisioner-675d74c797-pn4sw    5/5    Running   0           56d
csi-rbdplugin-provisioner-675d74c797-szn8z    5/5    Running   3           56d
rook-ceph-agent-qhbpc               1/1    Running   0           56d
rook-ceph-agent-xmtkl               1/1    Running   0           56d
rook-ceph-mds-cephfs-a-56dcd84ddc-scqd7      1/1    Running   0           56d
rook-ceph-mds-cephfs-b-7d666c9755-l8kp2     1/1    Running   0           56d
rook-ceph-mgr-a-7ff587789b-k5mtx          1/1    Running   0           56d
rook-ceph-mon-a-754458fd4-l9czm           1/1    Running   0           56d
rook-ceph-mon-b-66d96d95d-wm9z1           1/1    Running   0           56d
rook-ceph-mon-c-89c7569fd-gg2mm           1/1    Running   0           56d
rook-ceph-operator-768dfb98c4-9hzvx        1/1    Running   2574        56d
rook-ceph-osd-0-75b46fcfb6-l2w8           1/1    Running   0           56d
rook-ceph-osd-1-7747c8df4f-skqjs          1/1    Running   0           56d
rook-ceph-osd-prepare-gpu2-sp4qk          0/1    Completed 0           35m
rook-ceph-osd-prepare-gpu3-fbjxq          0/1    Completed 0           35m
rook-ceph-tools-8d9d7c8f4-nnmxm           1/1    Running   0           56d
rook-discover-2f6w4                   1/1    Running   0           56d
rook-discover-tb2lw                    1/1    Running   0           56d
ubuntu@deepops:~$
```

Figure 8

```
ubuntu@deepops:~/kubetest$ kubectl get pv
NAME                                CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS  CLAIM                                STORAGECLASS  AGE
ASS  REASON  AGE
benchmark-pv                        2Gi       RWX           Retain          Bound   kubeflow/benchmark-pv-claim         rook-ceph    54d
-block
mnist-pv                             1Gi       RWO           Retain          Bound   kubeflow/mnist-pv-claim            rook-ceph    56d
-block
pvc-03346976-32fa-4ffd-a1ab-903e8a0f04a9  20Gi      RWO           Retain          Released  kubeflow/minio-pv-claim           rook-ceph    56d
-block
pvc-244a88dd-9eaa-4d1f-b6b2-483d8ac88b5c  1Gi       RWO           Retain          Bound   default/busybox-1                  rook-ceph    55d
-block
pvc-538c9319-c92c-4546-b69f-b06bef2f7d9  1Gi       RWO           Retain          Bound   default/busybox-2                  rook-ceph    54d
-block
pvc-6c14fa1f-cfb6-4d1f-bf1d-24f45808d5c3  10Gi      RWO           Retain          Released  kubeflow/katib-mysql               rook-ceph    56d
-block
pvc-7c2d3c1f-9013-4874-a9b4-6325577caac5  10Gi      RWO           Retain          Released  kubeflow/metadata-mysql            rook-ceph    56d
-block
pvc-7fb9c834-9111-46a2-bdc4-9a96e19e4b49  2Gi       RWX           Retain          Released  kubeflow/benchmark-pv-claim         rook-ceph    55d
-block
pvc-8b0f205c-de3b-4106-b5ef-82528fc56526  1Gi       RWO           Retain          Bound   kubeflow/mnist1-pvc                rook-ceph    54d
-block
pvc-d0c982dd-e06c-40d6-9d78-aba84de69f40  20Gi      RWO           Retain          Released  kubeflow/mysql-pv-claim             rook-ceph    56d
-block
pvc-d38d7173-741b-49ac-b76f-f0db2abd02ce  10Gi      RWO           Retain          Bound   kubeflow/metadata-mysql            rook-ceph    56d
-block
pvc-e85d194a-2a72-4b35-be27-ba0077446369  10Gi      RWO           Retain          Bound   kubeflow/katib-mysql               rook-ceph    56d
-block
pvc-eaff52d2-a24b-4c1d-83ed-866e70072bdf  20Gi      RWO           Retain          Bound   kubeflow/mysql-pv-claim             rook-ceph    56d
-block
pvc-f7a4732a-4f62-4a55-b918-1963fbc127ed  20Gi      RWO           Retain          Bound   kubeflow/minio-pv-claim            rook-ceph    56d
-block
```

Figure 9

## Scale-up with Kubernetes Cluster

Not only does a Cloud-native platform can provide reliable, resilient, and distributed microservices, but it also can scale up on demand. In the test environment, we have tested Deep Learning benchmark throughput with the number of POD replicas. Figure 10 is the snippet of the Kubernetes YAML configuration of the Deep Learning benchmark application. Figure 11 depicts the benchmark under different Deep learning frameworks.

```
apiVersion: "kubeflow.org/v1"
kind: "TFJob"
metadata:
  name: "benchmark8-gpu"
  namespace: kubeflow
spec:
  tfReplicaSpecs:
    Worker:
      replicas: 1
      restartPolicy: Never
      template:
        spec:
          containers:
            - name: tensorflow
              image: "gcr.io/kubeflow-images-public/kubebench/kubebench-example-tf-cnn-runner-gpu:v0.5.0-11-gea53ad5"
              #image: "cnn-gpu-tensor13"
              #image: "cnn-gpu-tf-2.3.1"
              #image: "cnn-gpu-tf-1.14.0"
              #imagePullPolicy: IfNotPresent
              command:
              args:
                - --batch_size=32
                - --model=alexnet
                - --variable_update=parameter_server
                - --local_parameter_device=gpu
                - --device=gpu
                - --num_gpus=2
                - --data_format=NHWC
                - --forward_only=true
              env:
                - name: KUBEBENCH_EXP_OUTPUT_PATH
                  value: "/opt/output"
                - name: LD_LIBRARY_PATH
                  value: "/usr/lib/x86_64-linux-gnu:/usr/local/cuda/extras/CUPTI/lib64"
              workingDir: "/opt/tf-benchmarks/scripts/tf_cnn_benchmarks"
              #imagePullPolicy: Never
              volumeMounts:
                - name: training
                  mountPath: "/output/data"
```

Figure 10

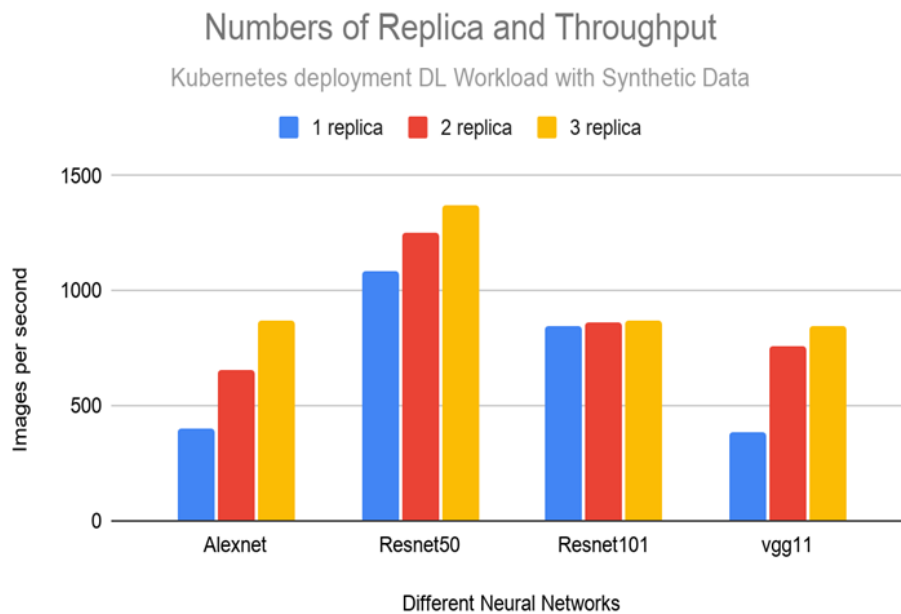


Figure 11



## Conclusion

The foundation of Cloud-native technologies is Kubernetes can be complicated to deploy, maintain and upgrade. Adding GPU into the mix can introduce another layer of complexity. However, with Supermicro AS-1114S-WTRT WIO systems, NVIDIA GPU appliances, and NVIDIA DeepOps tool, it becomes more straightforward for an organization to build a private Cloud-native platform along with MLOps applications. From a designing perspective, AS-1114S-WTRT as a flexible cloud node can meet compute, storage, and virtualization requirements for the hardware infrastructure; From an operational aspect, DeepOps makes the on-prem Cloud-native platform easier to deploy, maintain, and scale. This paper also demonstrated how the Kubernetes PODs could speed up Deep Learning processes that can reduce time-to-market.

## Reference

<https://docs.nvidia.com/datacenter/cloud-native/index.html>  
<https://www.kubeflow.org/docs/started/kubeflow-overview/>

[AMD, the AMD Arrow logo, EPYC, and combinations thereof are trademarks of Advanced Micro Devices, Inc.](#)