



# **Kubeflow**

## **NetApp Solutions**

NetApp  
July 31, 2024

This PDF was generated from [https://docs.netapp.com/us-en/netapp-solutions/ai/aicp\\_kubeflow\\_deployment\\_overview.html](https://docs.netapp.com/us-en/netapp-solutions/ai/aicp_kubeflow_deployment_overview.html) on July 31, 2024. Always check docs.netapp.com for the latest.

# Table of Contents

- Kubeflow ..... 1
- Kubeflow Deployment ..... 1
- Example Kubeflow Operations and Tasks ..... 2

# Kubeflow

## Kubeflow Deployment

This section describes the tasks that you must complete to deploy Kubeflow in your Kubernetes cluster.

### Prerequisites

Before you perform the deployment exercise that is outlined in this section, we assume that you have already performed the following tasks:

1. You already have a working Kubernetes cluster, and you are running a version of Kubernetes that is supported by the Kubeflow version that you intend to deploy. For a list of supported Kubernetes versions, refer to the dependencies for your Kubeflow version in the [official Kubeflow documentation](#).
2. You have already installed and configured NetApp Astra Trident in your Kubernetes cluster. For more details on Astra Trident, refer to the [Astra Trident documentation](#).

### Set Default Kubernetes StorageClass

Before you deploy Kubeflow, we recommend designating a default StorageClass within your Kubernetes cluster. The Kubeflow deployment process may attempt to provision new persistent volumes using the default StorageClass. If no StorageClass is designated as the default StorageClass, then the deployment may fail. To designate a default StorageClass within your cluster, perform the following task from the deployment jump host. If you have already designated a default StorageClass within your cluster, then you can skip this step.

1. Designate one of your existing StorageClasses as the default StorageClass. The example commands that follow show the designation of a StorageClass named `ontap-ai-flexvols-retain` as the default StorageClass.



The `ontap-nas-flexgroup` Trident Backend type has a minimum PVC size that is fairly large. By default, Kubeflow attempts to provision PVCs that are only a few GBs in size. Therefore, you should not designate a StorageClass that utilizes the `ontap-nas-flexgroup` Backend type as the default StorageClass for the purposes of Kubeflow deployment.

```

$ kubectl get sc
NAME                                PROVISIONER                        AGE
ontap-ai-flexgroups-retain          csi.trident.netapp.io             25h
ontap-ai-flexgroups-retain-iface1   csi.trident.netapp.io             25h
ontap-ai-flexgroups-retain-iface2   csi.trident.netapp.io             25h
ontap-ai-flexvols-retain             csi.trident.netapp.io             3s
$ kubectl patch storageclass ontap-ai-flexvols-retain -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
storageclass.storage.k8s.io/ontap-ai-flexvols-retain patched
$ kubectl get sc
NAME                                PROVISIONER                        AGE
ontap-ai-flexgroups-retain          csi.trident.netapp.io             25h
ontap-ai-flexgroups-retain-iface1   csi.trident.netapp.io             25h
ontap-ai-flexgroups-retain-iface2   csi.trident.netapp.io             25h
ontap-ai-flexvols-retain (default)   csi.trident.netapp.io             54s

```

## Kubeflow Deployment Options

There are many different options for deploying Kubeflow. Refer to the [official Kubeflow documentation](#) for a list of deployment options, and choose the option that is the best fit for your needs.

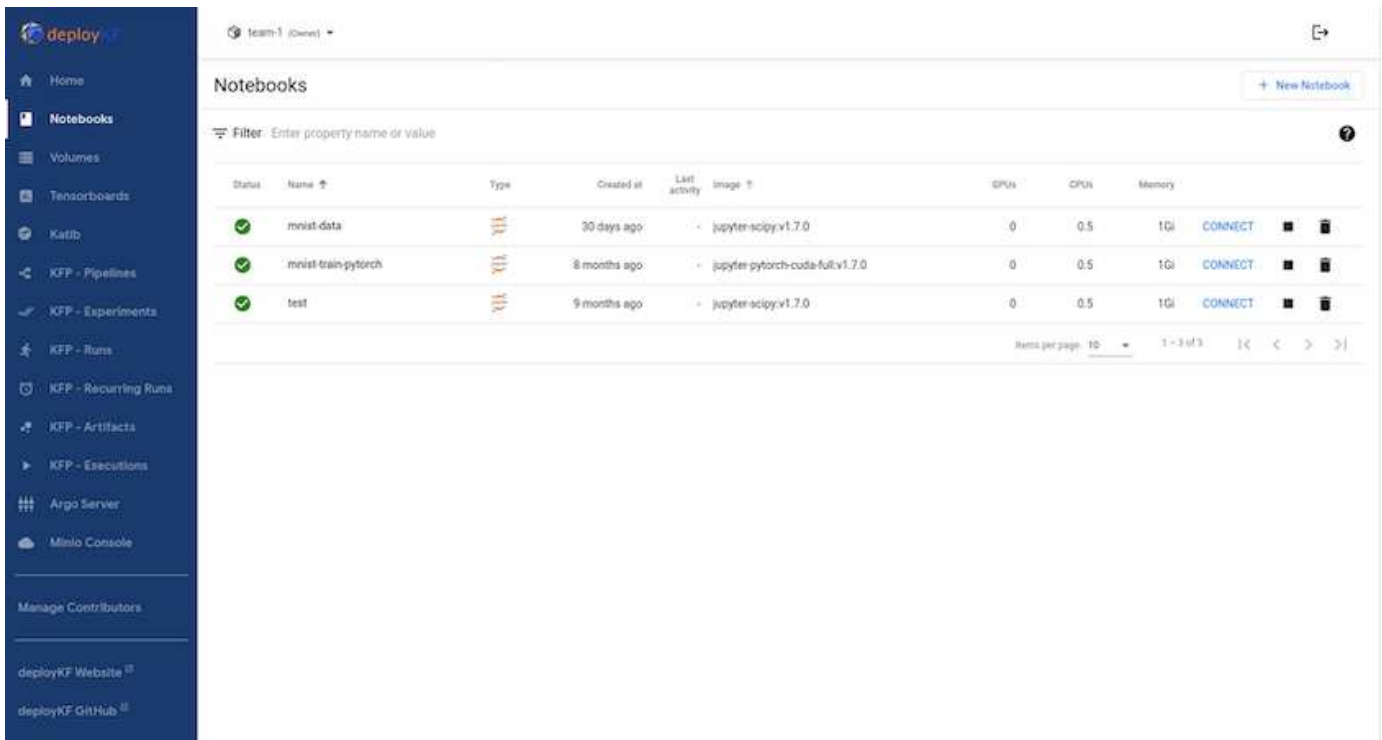


For validation purposes, we deployed Kubeflow 1.7 using [deployKF 0.1.1](#).

## Example Kubeflow Operations and Tasks

### Provision a Jupyter Notebook Workspace for Data Scientist or Developer Use

Kubeflow is capable of rapidly provisioning new Jupyter Notebook servers to act as data scientist workspaces. For more information about Jupyter Notebooks within the Kubeflow context, see the [official Kubeflow documentation](#).



## Use the NetApp DataOps Toolkit with Kubeflow

The [NetApp Data Science Toolkit for Kubernetes](#) can be used in conjunction with Kubeflow. Using the NetApp Data Science Toolkit with Kubeflow provides the following benefits:

- Data scientists can perform advanced NetApp data management operations, such as creating snapshots and clones, directly from within a Jupyter Notebook.
- Advanced NetApp data management operations, such as creating snapshots and clones, can be incorporated into automated workflows using the Kubeflow Pipelines framework.

Refer to the [Kubeflow Examples](#) section within the NetApp Data Science Toolkit GitHub repository for details on using the toolkit with Kubeflow.

## Example Workflow - Train an Image Recognition Model Using Kubeflow and the NetApp DataOps Toolkit

This section describes the steps involved in training and deploying a Neural Network for Image Recognition using Kubeflow and the NetApp DataOps Toolkit. This is intended to serve as an example to show a training job that incorporates NetApp storage.

### Prerequisites

Create a Dockerfile with the required configurations to use for the train and test steps within the Kubeflow pipeline.

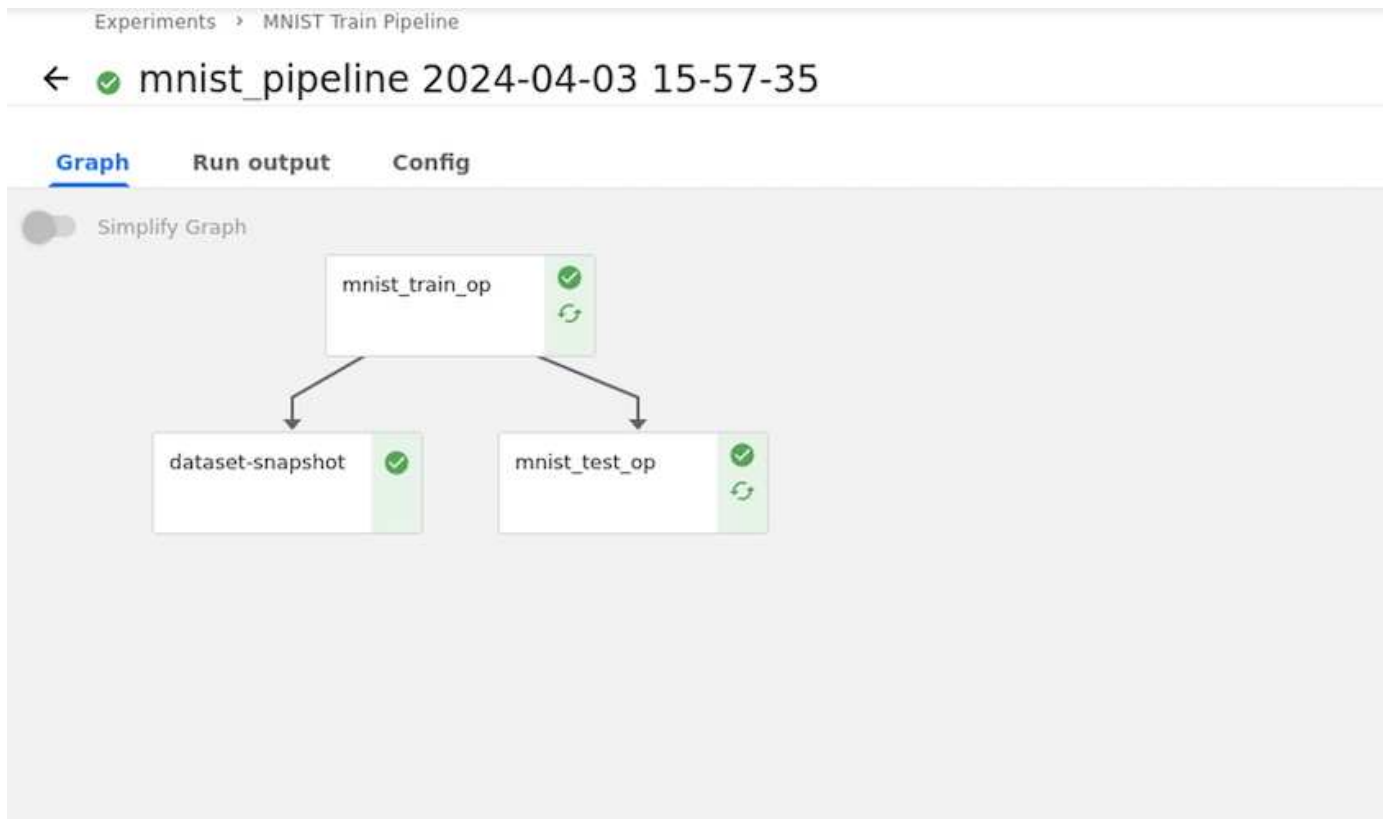
Here is an example of a Dockerfile -

```
FROM pytorch/pytorch:latest
RUN pip install torchvision numpy scikit-learn matplotlib tensorboard
WORKDIR /app
COPY . /app
COPY train_mnist.py /app/train_mnist.py
CMD ["python", "train_mnist.py"]
```

Depending on your requirements, install all required libraries and packages needed to run the program. Before you train the Machine Learning model, it is assumed that you already have a working Kubeflow deployment.

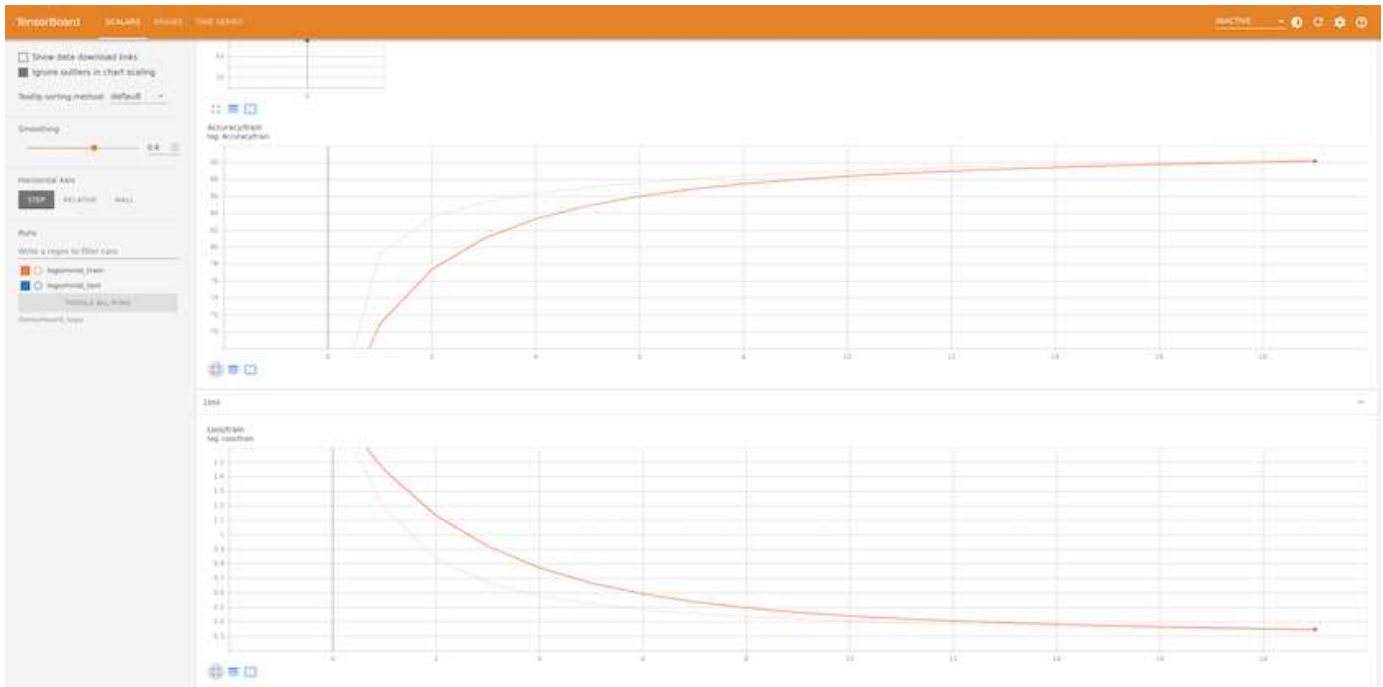
### Train a Small NN on MNIST Data Using PyTorch and Kubeflow Pipelines

We use the example of a small Neural Network trained on MNIST data. The MNIST dataset consists of handwritten images of digits from 0-9. The images are 28x28 pixels in size. The dataset is divided into 60,000 train images and 10,000 validation images. The Neural Network used for this experiment is a 2-layer feedforward network. Training is executed using Kubeflow Pipelines. Refer to the documentation [here](#) for more information. Our Kubeflow pipeline incorporates the docker image from the Prerequisites section.



### Visualize Results Using Tensorboard

Once the model is trained, we can visualize the results using Tensorboard. [Tensorboard](#) is available as a feature on the Kubeflow Dashboard. You can create a custom tensorboard for your job. An example below shows the plot of training accuracy vs. number of epochs and training loss vs. number of epochs.



## Experiment with Hyperparameters Using Katib

[Katib](#) is a tool within Kubeflow that can be used to experiment with the model hyperparameters. To create an experiment, define a desired metric/goal first. This is usually the test accuracy. Once the metric is defined, choose hyperparameters that you would like to play around with (optimizer/learning\_rate/number of layers). Katib does a hyperparameter sweep with the user-defined values to find the best combination of parameters that satisfy the desired metric. You can define these parameters in each section in the UI. Alternatively, you could define a **YAML** file with the necessary specifications. Below is an illustration of a Katib experiment -

The screenshot shows the 'Experiment details' page in the Kubeflow UI. The page is organized into several sections:

- Objective:**
  - Name: Validation-accuracy
  - Type: maximize
  - Goal: 0.9
  - Additional metrics: Train-accuracy
- Trials:**
  - Max failed trials: 3
  - Max trials: 12
  - Parallel trials: 3
- Parameters:**
  - lr: Parameter type: double, Min: 0.01, Max: 0.03
  - num-layers: Parameter type: int, Min: 1, Max: 64
  - optimizer: Parameter type: categorical, sgd, adam, ftrl
- Algorithm:**
  - Name: grid
- Metrics collector:**
  - Collector type: File

The screenshot shows the KubeFlow web interface. On the left is a navigation sidebar with options like Home, Notebooks, Volumes, Tensorboards, Katib, KFP - Pipelines, KFP - Experiments, KFP - Runs, KFP - Recurring Runs, KFP - Artifacts, KFP - Executions, Argo Server, and Minio Console. The main area is titled 'Experiment details' for 'team-1 (Owner)'. A message at the top says 'Couldn't find any successful Trial'. Below is a table with columns: OVERVIEW, TRIALS, DETAILS, and YAML. The table lists various metrics for the 'mnist-pytorch' experiment, such as Name, Status (Experiment is running), Best trial, Best trial's params, Best trial performance, User defined goal (Validation-accuracy > 0.9), Running trials (3), Failed trials (0), and Succeeded trials (0). There is also a section for 'Experiment Conditions' and a filter input.

### Use NetApp Snapshots to Save Data for Traceability

During the model training, we may want to save a snapshot of the training dataset for traceability. To do this, we can add a snapshot step to the pipeline as shown below. To create the snapshot, we can use the [NetApp DataOps Toolkit for Kubernetes](#).

```
@dsl.pipeline(
    name = 'MNIST Classification Pipeline',
    description = 'Train a simple NN for classification'
)
def mnist_pipeline():
    mnist_train_task = mnist_train_op()
    mnist_train_task.apply(
        kfp.onprem.mount_pvc('mnist-data', 'mnist-data-vol', '/mnt/data/')
    )

    mnist_test_task = mnist_test_op()
    mnist_test_task.apply(
        kfp.onprem.mount_pvc('mnist-data', 'mnist-data-vol', '/mnt/data/')
    )

    volume_snapshot_name = "mnist-pytorch-snapshot"
    dataset_snapshot = dsl.ContainerOp(
        name="dataset-snapshot",
        image="python:3.9",
        command=["/bin/bash", "-c"],
        arguments=["\
            python3 -m pip install netapp-dataops-k8s && \
            echo "" + volume_snapshot_name + "" > /volume_snapshot_name.txt && \
            netapp_dataops_k8s_cli.py create volume-snapshot --pvc-name="" + "mnist-data" + " --snapshot-name="" + str(volume_snapshot_name) + " --namespace={{workflow.namespace}}".
        "]
        file_outputs={"volume_snapshot_name": "/volume_snapshot_name.txt"}
    )
    mnist_test_task.after(mnist_train_task)
    dataset_snapshot.after(mnist_train_task)
```

Refer to the [NetApp DataOps Toolkit example for Kubeflow](#) for more information.



## Copyright information

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

## Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.