

ModelArts

Data Preparation and Analysis

Issue 01
Date 2024-08-15



Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2024. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Introduction to Data Preparation.....	1
2 Getting Started.....	3
3 Creating a Dataset.....	10
3.1 Dataset Overview.....	10
3.2 Creating a Dataset.....	12
3.3 Modifying a Dataset.....	19
4 Importing Data.....	20
4.1 Introduction to Data Importing.....	20
4.2 Importing Data from OBS.....	21
4.2.1 Introduction to Importing Data from OBS.....	22
4.2.2 Importing Data from an OBS Path.....	24
4.2.3 Specifications for Importing Data from an OBS Directory.....	26
4.2.4 Importing a Manifest File.....	32
4.2.5 Specifications for Importing a Manifest File.....	33
4.3 Importing Data from DLI.....	51
4.4 Importing Data from MRS.....	52
4.5 Importing Data from DWS.....	52
4.6 Importing Data from Local Files.....	53
5 Data Analysis and Preview.....	55
5.1 Auto Grouping.....	55
5.2 Data Filtering.....	57
5.3 Data Feature Analysis.....	58
6 Labeling Data.....	64
7 Publishing Data.....	65
7.1 Introduction to Data Publishing.....	65
7.2 Publishing a Data Version.....	65
7.3 Managing Data Versions.....	67
8 Exporting Data.....	69
8.1 Introduction to Exporting Data.....	69
8.2 Exporting Data to a New Dataset.....	69

8.3 Exporting Data to OBS..... 70

1 Introduction to Data Preparation

NOTE

Data management is being upgraded and is invisible to users who have not used data management. To use data management functions, you are advised to submit a service ticket to obtain the permissions.

The driving forces behind AI are computing power, algorithms, and data. Data quality affects model precision. Generally, a large amount of high-quality data is more likely to train a high-precision AI model. Models trained using normal data achieves 85% to 90% accuracy, while commercial applications have higher requirements. If you want to improve the model accuracy to 96% or even 99%, a large amount of high-quality data is required. In this case, the data must be more refined, scenario-based, and professional. The preparation of a large amount of high-quality data has become a challenging issue in AI development.

ModelArts is a one-stop AI development platform that supports AI lifecycle development, including data processing, algorithm development, model training, and model deployment. In addition, ModelArts provides AI Gallery that can be used to share data, algorithms, and models. ModelArts data management provides end-to-end data preparation, processing, and labeling.

ModelArts data management provides the following functions for you to obtain high-quality AI data:

- Data acquisition
 - Allows you to import data from OBS, MRS, DLI, and GaussDB(DWS).
 - Provides 18+ data augmentation operators to increase data volume for training.
- Improved data quality
 - Allows you to preview various formats of data including images, text, audios, and videos, helping you identify data quality.
 - Allows you to filter data by multiple search criteria, such as sample attributes and labeling information.
 - Provides 12+ labeling tools for refined, scenario-based, and professional data labeling.
 - Performs feature analysis based on samples and labeling results, helping you understand data quality.

- More efficient data preparation
 - Allows you to manage data by version for more efficient data management.
 - Provides capabilities such as interactive labeling and auto labeling for more efficient data labeling.
 - Enables team labeling and team labeling management for labeling a large amount of data.

2 Getting Started

This section uses preparing data for training an object detection model as an example to describe how to analyze and label sample data. During actual service development, you can select one or more data management functions to prepare data based on service requirements. The operation process is as follows:

- [Making Preparations](#)
- [Creating a Dataset](#)
- [Analyzing Data](#)
- [Labeling Data](#)
- [Publishing Data](#)
- [Exporting Data](#)

Preparations

Before using data management of ModelArts, complete the following preparations:

When using data management, ModelArts needs to access dependent services such as OBS. Therefore, grant permissions on the **Global Configuration** page. For details, see [Configuring Agency Authorization \(Recommended\)](#).

Creating a Dataset

In this example, an OBS path is used as the input path to create a dataset. Perform the following operations to create an object detection dataset and import the data to the dataset:

- Step 1** Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
- Step 2** Click **Create**. On the **Create Dataset** page, create a dataset based on the data type and data labeling requirements.
 1. Set the basic information, the name and description of the dataset. Set **Data Type** to **Images** and **Data Source** to **OBS**.
 2. Select an OBS path as **Input Dataset Path** Import Path, and select another OBS path as **Output Dataset Path**.

Figure 2-1 Data import and output paths

The screenshot shows a configuration interface for data import and output paths. It includes the following fields and options:

- Data Source:** Radio buttons for "OBS" (selected) and "Local file".
- Import Mode:** Radio button for "Path" (selected).
- Import Path:** A text input field with the placeholder "Select an OBS path." and a file selection icon. Below it, a note states: "You can import up to 1,000,000 samples and 1,000 labels."
- Labeling Status:** Radio buttons for "Unlabeled" (selected) and "Labeled".
- Output Dataset Path:** A text input field with the placeholder "Select an OBS path." and a file selection icon. Below it, a note states: "Path for storing output files such as labeled files. The path cannot be the same as the import path or subdirectory of the import path."

3. After setting the parameters, click **CreateSubmit** in the lower right corner to create a dataset.

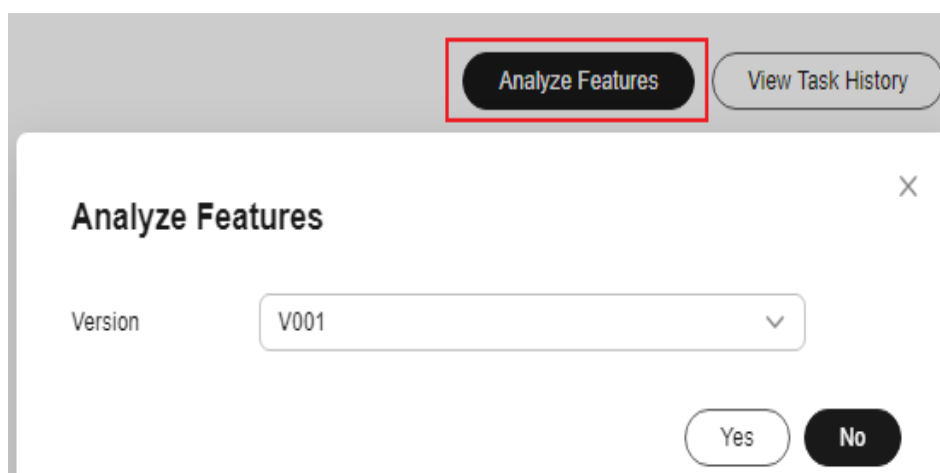
----End

Analyzing Data

After a dataset is created, you can perform data analysis based on image features, such as blurs and brightness, to better understand the data quality and determine whether the dataset meets your algorithm and model requirements.

1. Create a feature analysis task.
 - a. Before performing feature analysis, publish a dataset version. On the dataset details page, click **Publish** in the upper right corner to publish a new version of the dataset.
 - b. After the version is published, click the **View Data Feature** tab and click **Analyze Features**. In the dialog box that is displayed, select the published dataset version and click **Yes** to start the feature analysis task.

Figure 2-2 Starting feature analysis



- c. View the task progress.

You can click **View Task History** to view the task progress. When the task status changes to **Successful**, the task is complete.

Figure 2-3 Feature analysis progress

Dataset Version	Task ID	Created	Duration(hh...	Status
V002	ZjrO7W8735...	Mar 16, 2022 ...	00:01:23	Successful

2. View feature analysis results.

After the feature analysis is complete, you can select **Version**, **Type**, and **Data Feature Metric** on the **View Data Feature** tab page. Then, the selected versions and metrics are displayed on the page. The displayed chart helps you understand data distribution for a better understanding of your data.

- **Version:** Select one or more versions for comparison.
- **Type:** Select types to be analyzed. The values **all**, **train**, **eval**, and **inference** are available for you to select. They indicate all, training, evaluation, and inference, respectively.
- **Data Feature Metric:** Select the metrics to be displayed. For details about the metrics, see [Data feature metrics](#).

In feature analysis results, for example, image brightness distribution is uneven, which means images of a certain brightness are lacking. This greatly affects model training. In this case, increase images of that brightness to make data more even for subsequent model building.

Labeling Data

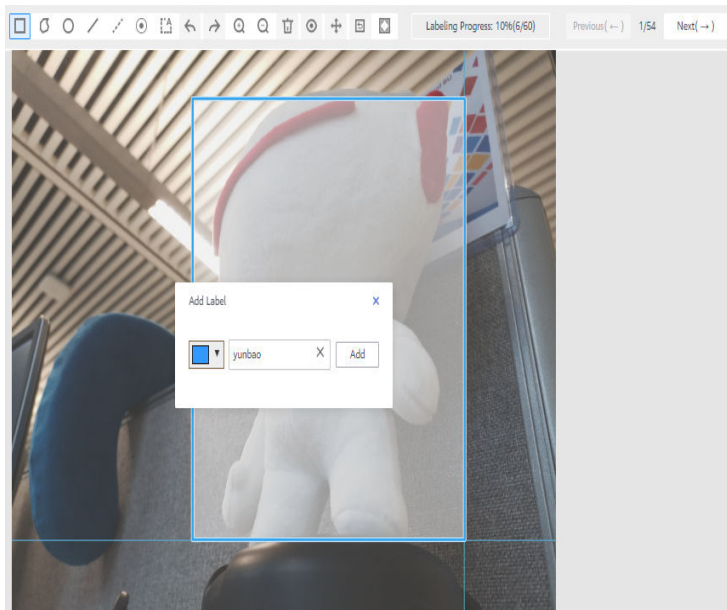
- Manual labeling
 - a. On the **Unlabeled** tab page, click an image. The system automatically directs you to the page for labeling the image.
 - b. On the toolbar of the labeling page, select a proper labeling tool. In this example, a rectangle is used for labeling.

Figure 2-4 Labeling tools



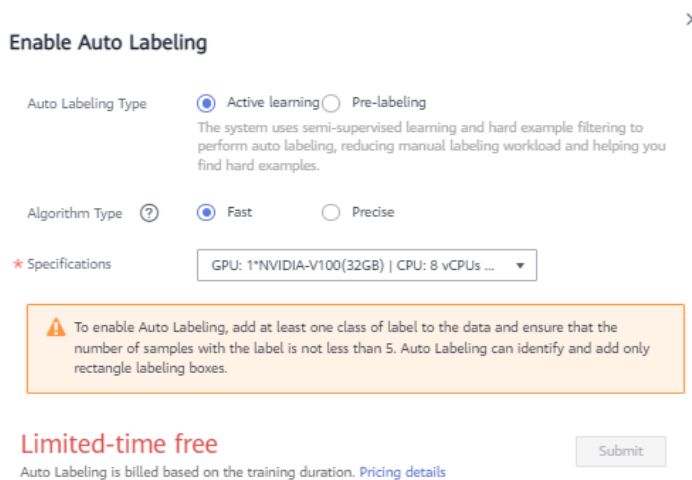
- c. Drag the mouse to select an object, enter a new label name in the displayed text box. If labels already exist, select one from the drop-down list box. Click **Add**.

Figure 2-5 Adding an object detection label



- d. Click **Back to Data Labeling Preview** in the upper left part of the page to view the labeling information. In the dialog box that is displayed, click **Yes** to save the labeling settings. The selected image is automatically moved to the **Labeled** tab page. On the **Unlabeled** and **All** tab pages, the labeling information is updated along with the labeling process, including the added label names and the number of images for each label.
- Auto labeling
Auto labeling allows you to automatically label remaining data after a small amount of data is manually labeled.
 - a. On the dataset details page, click **Auto Label** in the upper right corner.
 - b. In the **Enable Auto Labeling** dialog box, set the following parameters and click **Submit**.
 - **Auto Labeling Type: Active learning**
 - **Algorithm Type: Fast**Retain the default values of other parameters.

Figure 2-6 Starting auto labeling



- c. View auto labeling progress.

After auto labeling is started, you can view the task progress on the **To Be Confirmed** tab page. After a task is complete, you can view the automatically labeled data on the **To Be Confirmed** tab page.

Figure 2-7 Viewing auto labeling progress

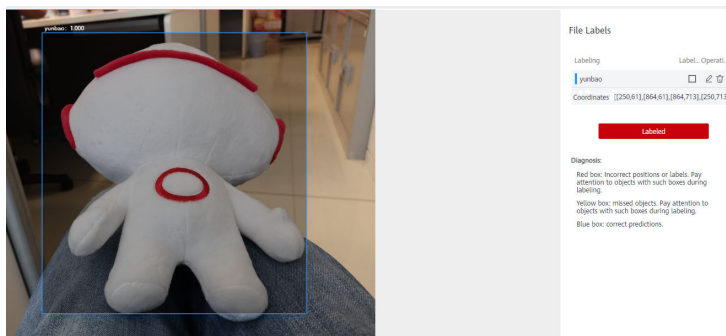


- d. Confirm auto labeling results.

After auto labeling is complete, click the image on the **To be confirmed** tab page. On the labeling details page, you can view or modify the auto labeling result.

For correct labeling, click **Labeled** on the right. For wrong labeling, correct wrong labels. For auto labeling of object detection datasets, confirm images one by one. Ensure that all images are confirmed and go to the next step.

Figure 2-8 Confirming auto labeling results



Publishing Data

ModelArts training management allows you to create training jobs using ModelArts datasets or files in an OBS directory. If a dataset is used as the data source of a training job, specify a dataset and version. Therefore, you must have published a dataset version. For details, see [Publishing a Data Version](#).

NOTE

Data that is from the same source and labeled in different batches are differentiated by version. This facilitates subsequent model building and development. You can select specified versions.

Exporting Data



ModelArts training management allows you to create training jobs using ModelArts datasets or files in an OBS directory. If you create a training job using an OBS directory, export the prepared data to OBS.

1. Export data to OBS.
 - a. On the dataset details page, select or filter the data to be exported, and click **Export** in the upper right corner.
 - b. Set Data Source to **OBS**, enter related information, and click **OK**.
Storage Path: path where the data to be exported is stored. You are advised not to save data to the input or output path of the current dataset.

Figure 2-9 Exporting data to OBS

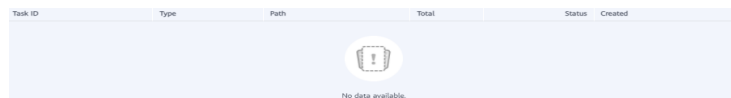
Export To

Data Source

Storage Path  

- c. After the data is exported, view it in the specified path.
2. View task history.
After exporting data, you can view the export task details in **Export History**.
 - a. On the dataset details page, click **Export History** in the upper right corner.
 - b. In the **View Task History** dialog box, view the export task history of the current dataset. You can view the task ID, creation time, export type, export path, total number of exported samples, and export status.

Figure 2-10 Export history



3 Creating a Dataset

3.1 Dataset Overview

Dataset Types

ModelArts supports the following types of datasets:

- Images: in .jpg, .png, .jpeg, or .bmp format for image classification, image segmentation, and object detection
- Audio: in .wav format for sound classification, speech labeling, and speech paragraph labeling
- Text: in .txt or .csv format for text classification, named entity recognition, and text triplet labeling
- Video: in .mp4 format for video labeling
- Free format: allows data in any format. Labeling is not available for free format data. The free format applies if labeling is not required or needs to be customized. Select this format if your data is in multiple formats or your data is not in any of the preceding formats.
- Tables
Table: applies to structured data processing such as tables. The file format can be CSV. Tables cannot be labeled but you can preview up to 100 data records in a table.

Dataset Functions

Different types of datasets support different functions, such as auto labeling and team labeling. For details, see [Table 3-1](#).

Table 3-1 Functions supported by different types of datasets

Data set Type	Labeling Type	Creating a Dataset	Importing Data	Exporting Data	Publishing a Dataset	Modifying a Dataset	Managing Dataset Versions	Auto Grouping	Data Features
Image	Image classification	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
	Object detection	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
	Image segmentation	Supported	Supported	Supported	Supported	Supported	Supported	Supported	N/A
Audio	Sound classification	Supported	Supported	N/A	Supported	Supported	Supported	N/A	N/A
	Speech labeling	Supported	Supported	N/A	Supported	Supported	Supported	N/A	N/A
	Speech paragraph labeling	Supported	Supported	N/A	Supported	Supported	Supported	N/A	N/A
Text	Text classification	Supported	Supported	N/A	Supported	Supported	Supported	N/A	N/A
	Name entity recognition	Supported	Supported	N/A	Supported	Supported	Supported	N/A	N/A

Data set Type	Labeling Type	Creating a Dataset	Importing Data	Exporting Data	Publishing a Dataset	Modifying a Dataset	Managing Dataset Versions	Auto Grouping	Data Features
	Text triplet	Supported	Supported	N/A	Supported	Supported	Supported	N/A	N/A
Video	Video labeling	Supported	Supported	N/A	Supported	Supported	Supported	N/A	N/A
Free format	Free format	Supported	N/A	-	Supported	Supported	Supported	N/A	N/A
Table	Table	Supported	Supported	N/A	Supported	Supported	Supported	N/A	N/A

Specifications Restrictions

- The maximum numbers of samples and labels in a single text, video, or audio database other than a table dataset are 1,000,000 and 10,000, respectively.
- The maximum size of a sample in a single text, video, or audio database other than an image dataset is 5 GB.
- The maximum size of an image for object detection, image segmentation, or image classification is 25 MB.
- The maximum size of a manifest file is 5 GB.
- The maximum size of a text file in a line is 100 KB.
- The maximum size of a labeling result file is 100 MB.

3.2 Creating a Dataset

Before using ModelArts to manage data, create a dataset. Then, you can perform operations on the dataset, such as labeling data, importing data, and publishing the dataset. This section describes how to create a dataset of the non-table type (image, audio, text, video, and free format) and table type.

Prerequisites

- You have been authorized to access OBS. To do so, click the **Settings** page in the navigation pane of the ModelArts management console and add access authorization using an agency.
- OBS buckets and folders for storing data are available. In addition, the OBS buckets and ModelArts are in the same region. OBS parallel file systems are not supported. Select object storage.
- OBS buckets are not encrypted. ModelArts does not support encrypted OBS buckets. When creating an OBS bucket, do not enable bucket encryption.

Image, Audio, Text, Video, and Free Format

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. Click **Create**. On the **Create Dataset** page, create a dataset based on the data type and data labeling requirements.

Figure 3-1 Parameter settings

The screenshot displays the 'Create Dataset' configuration interface. It includes the following elements:

- Data Type:** A row of buttons for 'Images', 'Audio', 'Text', 'Video', 'Free format', and 'Table'. Below this row, it lists supported formats: .jpg, .png, .jpeg, .bmp.
- Data Source:** Three buttons: 'OBS' (selected), 'AI Gallery', and 'Local file'.
- Import Mode:** A single button labeled 'Path'.
- Import Path:** A text input field with the placeholder 'Select an OBS path.' and a file selection icon. A note below states: 'You can import up to 200,000 samples and 100,000 labels.'
- Labeling Status:** Two buttons: 'Unlabeled' (selected) and 'Labeled'.
- Output Dataset Path:** A text input field with the placeholder 'Select an OBS path.' and a file selection icon. A note below states: 'Path for storing output files such as labeled files. The path cannot be the same as the import path or subdirectory of the import path.'

- **Name:** name of the dataset, which is customizable
- **Description:** details about the dataset
- **Data Type:** Select a data type based on your needs.
- **Data Source**
 - i. Importing data from OBS

If data is available in OBS, select **OBS** for **Data Source**, and set **Import Mode**, **Import Path**, **Labeling Status**, and **Labeling Format** (mandatory when **Labeling Status** is set to **Labeled**). The labeling formats of the input data vary depending on the dataset type. For details about the labeling formats supported by ModelArts, see [Importing Data](#).
 - ii. Importing data from a local path

If data is not stored in OBS and the required data cannot be downloaded from AI Gallery, ModelArts enables you to upload the data from a local path. Before uploading data, configure **Storage Path** and **Labeling Status**. Click **Upload data** to select the local file for uploading. Select a labeling format when the labeling status is **Labeled**. The labeling formats of the input data vary depending on the dataset type. For details about the labeling formats supported by ModelArts, see [Importing Data](#).

Figure 3-2 Selecting **Local file**

★ Data Source: OBS, AI Gallery, **Local file**

★ Storage Path: Select an OBS path.

You can import up to 200,000 samples and 100,000 labels.

Uploading Data: Upload data

★ Labeling Status: **Unlabeled**, Labeled

- For more details about parameters, see [Table 3-2](#).

Table 3-2 Dataset parameters

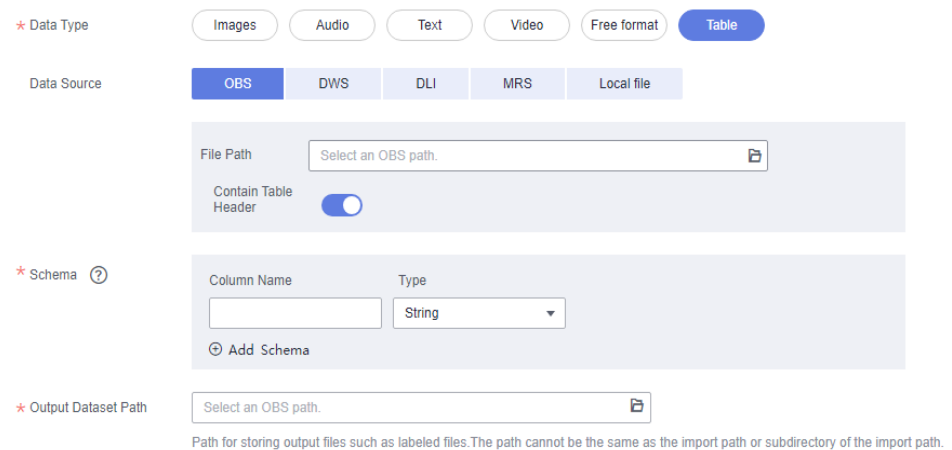
Parameter	Description
Import Path	<p>OBS path from which your data is to be imported. This path is used as the data storage path of the dataset.</p> <p>NOTE OBS parallel file systems are not supported. Select an OBS bucket.</p> <p>When you create a dataset, data in the OBS path will be imported to the dataset. If you modify data in OBS, the data in the dataset will be inconsistent with that in OBS. As a result, certain data may be unavailable. To modify data in a dataset, follow the operations provided in Import Mode or Importing Data from an OBS Path.</p> <p>If the numbers of samples and labels of the dataset exceed quotas, importing the samples and labels will fail.</p>
Labeling Status	<p>Labeling status of the selected data, which can be Unlabeled or Labeled.</p> <p>If you select Labeled, specify a labeling format and ensure the data file complies with format specifications. Otherwise, the import may fail.</p> <p>Only image (object detection, image classification, and image segmentation), audio (sound classification), and text (text classification) labeling tasks support the import of labeled data.</p>
Output Dataset Path	<p>OBS path where your labeled data is stored.</p> <p>NOTE</p> <ul style="list-style-type: none"> • Ensure that your OBS path name contains letters, digits, and underscores (_) and does not contain special characters, such as ~'@#\$\$%^&*[] ; : ; + = < > / and spaces. • The dataset output path cannot be the same as the data input path or subdirectory of the data input path. • It is a good practice to select an empty directory as the dataset output path. • OBS parallel file systems are not supported. Select an OBS bucket.

3. After setting the parameters, click **Submit**.

Table


1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. Click **Create**. On the **Create Dataset** page, create a table dataset based on the data type and data labeling requirements.

Figure 3-3 Parameters of a table dataset




* Data Type: Images, Audio, Text, Video, Free format, **Table**


Data Source: **OBS**, DWS, DLI, MRS, Local file


File Path: Select an OBS path. 

Contain Table Header:

* Schema 

Column Name	Type
<input type="text"/>	String

 Add Schema

* Output Dataset Path: Select an OBS path. 

Path for storing output files such as labeled files. The path cannot be the same as the import path or subdirectory of the import path.

- **Name:** name of the dataset, which is customizable
- **Description:** details about the dataset
- **Data Type:** Select a data type based on your needs.
- For more details about parameters, see [Table 3-3](#).

Table 3-3 Dataset parameters

Parameter	Description
Data Source (OBS)	<ul style="list-style-type: none"> ● File Path: Browse all OBS buckets of the account and select the directory where the data file to be imported is located. ● Contain Table Header: This setting is enabled by default, indicating that the imported file contains table headers. <ul style="list-style-type: none"> - If the original table contains table headers and this setting is enabled, first rows (table header) of the imported file are used as column names. You do not need to modify the schema information. - If the original table does not contain table headers, you need to disable this setting and change column names in Schema to attr_1, attr_2, ..., and attr_n. attr_n is the last column, indicating the prediction column. <p>For details about OBS functions, see Object Storage Service Console Operation Guide.</p>
Data Source (DWS)	<ul style="list-style-type: none"> ● Cluster Name: All DWS clusters of the current account are automatically displayed. Select the required DWS cluster from the drop-down list. ● Database Name: Enter the name of the database where the data is located based on the selected DWS cluster. ● Table Name: Enter the name of the table where the data is located based on the selected database. ● User Name: Enter the username of the DWS cluster administrator. ● Password: Enter the password of the DWS cluster administrator. <p>For details about DWS functions, see Data Warehouse Service User Guide.</p> <p>NOTE To import data from DWS, use DLI functions. If you do not have the permission to access DLI, create a DLI agency as prompted.</p>

Parameter	Description
Data Source (DLI)	<ul style="list-style-type: none"> ● Queue Name: All DLI queues of the current account are automatically displayed. Select the required queue from the drop-down list. ● Database Name: All databases are displayed based on the selected queue. Select the required database from the drop-down list. ● Table Name: All tables in the selected database are displayed. Select the required table from the drop-down list. <p>For details about DLI functions, see Data Lake Insight User Guide.</p>
Data Source (MRS)	<ul style="list-style-type: none"> ● Cluster Name: All MRS clusters of the current account are automatically displayed. However, streaming clusters do not support data import. Select the required cluster from the drop-down list. ● File Path: Enter the HDFS file path based on the selected cluster. ● Contain Table Header: If this setting is enabled, the imported file contains table headers. <p>For details about MRS functions, see MapReduce Service User Guide.</p>
Local file	<p>Storage Path: Select an OBS path.</p>
Schema	<p>Names and types of table columns, which must be the same as those of the imported data. Set the column name based on the imported data and select the column type. For details about the supported types, see Table 3-4.</p> <p>Click Add Schema to add a new record. When creating a dataset, you must specify a schema. Once created, the schema cannot be modified.</p> <p>When data is imported from OBS, the schema of the CSV file in the file path is automatically obtained. If the schemas of multiple CSV files are inconsistent, an error will be reported.</p> <p>NOTE After you select data from OBS, column names in Schema are automatically displayed, which is the first-row data of the table by default. To ensure the correct prediction code, you need to change column names in Schema to attr_1, attr_2, ..., and attr_n. attr_n is the last column, indicating the prediction column.</p>

Parameter	Description
Output Dataset Path	<p>OBS path for storing table data. The data imported from the data source is stored in this path. The path cannot be the same as the file path in the OBS data source or subdirectories of the file path.</p> <p>After a table dataset is created, the following four directories are automatically generated in the storage path:</p> <ul style="list-style-type: none"> • annotation: version publishing directory. Each time a version is published, a subdirectory with the same name as the version is generated in this directory. • data: data storage directory. Imported data is stored in this directory. • logs: directory for storing logs. • temp: temporary working directory.

Table 3-4 Schema data types

Type	Description	Storage Space	Range
String	String type	N/A	N/A
Short	Signed integer	2 bytes	-32768 to 32767
Int	Signed integer	4 bytes	-2147483648 to 2147483647
Long	Signed integer	8 bytes	-9223372036854775808 to 9223372036854775807
Double	Double-precision floating point	8 bytes	N/A
Float	Single-precision floating point	4 bytes	N/A
Byte	Signed integer	1 byte	-128 to 127
Date	Date type in the format of "yyyy-MM-dd", for example, 2014-05-29	N/A	N/A
Timestamp	Timestamp that represents date and time in the format of "yyyy-MM-dd HH:mm:ss"	N/A	N/A

Type	Description	Storage Space	Range
Boolean	Boolean type	1 byte	TRUE/FALSE

 **NOTE**

When using a CSV file, pay attention to the following:

- When the data type is set to **String**, the data in the double quotation marks is regarded as one record by default. Ensure the double quotation marks in the same row are closed. Otherwise, the data will be too large to display.
- If the number of columns in a row of the CSV file is different from that defined in the schema, the row will be ignored.

3. After setting the parameters, click **Submit**.

3.3 Modifying a Dataset

The basic information of a created dataset can be modified to keep pace with service changes.

Prerequisites

A created dataset is available.

Modifying the Basic Information of a Dataset

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. In the dataset list, choose **More > Modify** in the **Operation** column of the target dataset. Modify the basic information and click **OK**.

Table 3-5 Parameters

Parameter	Description
Name	Name of a dataset, which must be 1 to 64 characters long and start with a letter. Only letters, digits, underscores (_), and hyphens (-) are allowed. The name must start with a letter.
Description	Brief description of the dataset.

4 Importing Data

4.1 Introduction to Data Importing

After a dataset is created, you can import more data. ModelArts allows you to import data from different data sources.

- [Importing Data from OBS](#)
- [Importing Data from DLI](#)
- [Importing Data from MRS](#)
- [Importing Data from DWS](#)
- [Importing Data from Local Files](#)

ModelArts AI Gallery provides a large number of built-in datasets. You can download and use the built-in datasets from AI Gallery. You can also import your data to ModelArts.

File Data Sources

You can import data by downloading built-in datasets from AI Gallery, or from OBS or a local file. After the import, the data from the import path is automatically synchronized to the data source path of the dataset.

- **OBS:** Import data from an OBS path or a manifest file.
- **Local file:** Import local data that has been uploaded to an OBS path.

Table Data Sources

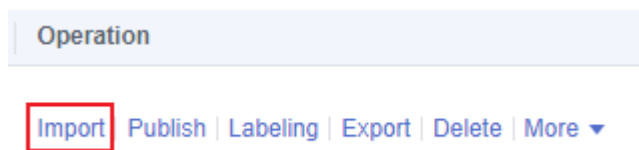
You can import data by downloading built-in datasets from AI Gallery, or from OBS, DWS, DLI, MRS, and local files.

Import Mode

There are five modes for importing data to a dataset.

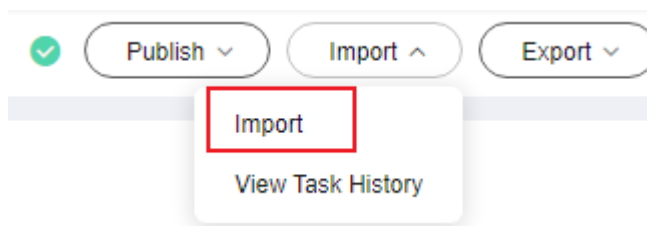
- When you create a dataset, select an import path. The data is automatically synchronized from the import path.
- After a dataset is created, click **Import** in the **Operation** column on the dataset list page.

Figure 4-1 Importing data on the dataset list page



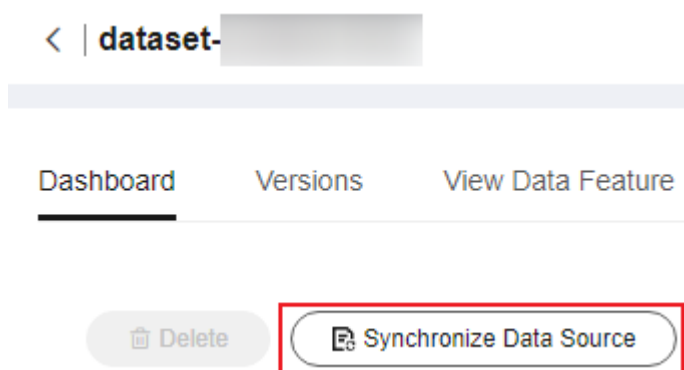
- On the dataset list page, click a dataset. On the dataset details page, choose **Import > Import**.

Figure 4-2 Importing data on the dataset details page



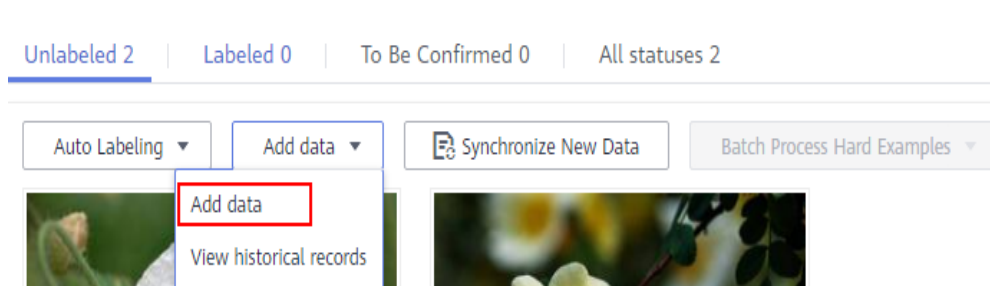
- On the dataset list page, click a dataset. On the dataset details page, click **Synchronize Data Source** to synchronize data from OBS.

Figure 4-3 Synchronizing data sources on the dataset details page



- Add data on the labeling job details page.

Figure 4-4 Adding data on the labeling job details page



4.2 Importing Data from OBS

4.2.1 Introduction to Importing Data from OBS

Import Modes

You can import data from OBS through an OBS path or a manifest file.

- **OBS path:** indicates that the dataset to be imported has been stored in an OBS path. In this case, select an OBS path that you can access. In addition, the directory structure in the OBS path must comply with the specifications. For details, see [Specifications for Importing Data from an OBS Directory](#). This import mode is available only for the following types of datasets: **Image classification**, **Object detection**, **Text classification**, **Table**, and **Sound classification**. For other types of datasets, data can be imported only through a manifest file.
- **Manifest file:** indicates that the dataset file is in the manifest format and the manifest file has been uploaded to OBS. The manifest file defines the mapping between labeling objects and content. For details about the specifications of manifest files, see [Specifications for Importing a Manifest File](#).

 **NOTE**

Before importing an object detection dataset, ensure that the labeling range of the labeling file does not exceed the size of the original image. Otherwise, the import may fail.

Table 4-1 Import modes supported by datasets

Dataset Type	Labeling Type	From an OBS Path	From a Manifest File
Images	Image classification	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Image Classification	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Image Classification
	Object detection	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Object Detection	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Object Detection
	Image segmentation	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Image Segmentation	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Image Segmentation

Dataset Type	Labeling Type	From an OBS Path	From a Manifest File
Audio	Sound classification	Supported You can import unlabeled or labeled data. Follow the format specifications described in Sound Classification .	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Sound Classification
	Speech labeling	Supported You can import unlabeled data.	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Speech Labeling
	Speech paragraph labeling	Supported You can import unlabeled data.	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Speech Paragraph Labeling
Text	Text classification	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Text Classification	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Text Classification
	Named entity recognition	Supported You can import unlabeled data.	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Named Entity Recognition
	Text triplet	Supported You can import unlabeled data.	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Text Triplet

Dataset Type	Labeling Type	From an OBS Path	From a Manifest File
Video	Video labeling	Supported You can import unlabeled data.	Supported You can import unlabeled or labeled data. Format specifications of labeled data: Video Labeling
Other	Free format	Supported You can import unlabeled data.	N/A
Table	Table	Supported You can also import data from DWS, DLI, or MRS. Follow the format specifications described in Tables .	N/A

4.2.2 Importing Data from an OBS Path

Prerequisites

- A dataset is available.
- The data to be imported is stored in OBS. The manifest file is stored in OBS.
- The OBS bucket and ModelArts are in the same region and you can operate the bucket.

Importing File Data from an OBS Path

The parameters on the GUI for data import vary according to the dataset type. The following uses a dataset of the image classification type as an example.

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. Locate the row that contains the desired dataset and click **Import** in the **Operation** column. Alternatively, click the dataset name to go to the **Dashboard** tab page of the dataset, and click **Import** in the upper right corner.
3. In the **Import** dialog box, configure parameters as follows and click **OK**.
 - **Data Source: OBS**
 - **Import Mode: Path**
 - **Import Path:** OBS path for storing data
 - **Labeling Status: Labeled**

- **Advanced Feature Settings:** disabled by default
Import by Tag enables the system to automatically obtain the labels of the current dataset. Click **Add Label** to add a label. This parameter is optional. If **Import by Tag** is disabled, you can add or delete labels for imported data when labeling data.

Figure 4-5 Importing data from an OBS path

Import

* Data Source: OBS (selected), Local file

* Import Mode: Path (selected), manifest

You can save the dataset file to be imported to the OBS path that you have permission to access. [Labeling file format](#)

* Import Path: Select an OBS path.

You can import up to 200,000 samples and 100,000 labels.

* Labeling Status: Unlabeled, Labeled (selected)

* Labeling Format: Image classification (selected), ModelArts imageNet 1.0 (selected)

Images with the same label must be stored in the same directory, with the label name as the directory name.

File storage structure

```
dataset-import-example
├── Cat
│   ├── image1.jpg
│   └── image2.jpg
└── Dog
    ├── image11.jpg
    └── image12.jpg
```

OK Cancel

After the data is imported, it will be automatically synchronized to the dataset. On the **Datasets** page, click the dataset name to view its details and create a labeling job to label the data.

Labeling Status of File Data

The labeling status can be **Unlabeled** or **Labeled**.

- **Unlabeled:** Only the labeling object (such as unlabeled images or texts) is imported.
- **Labeled:** Both the labeling object and content are imported. Labeling content importing is not supported for datasets in free format.

To ensure that the labeling content can be correctly read, you must store data in strict accordance with the specifications.

If **Import Mode** is set to **Path**, store the data to be imported according to the labeling file specifications. For details, see [Specifications for Importing Data from an OBS Directory](#).

If **Import Mode** is set to **manifest**, the manifest file specifications must be met.

 **NOTE**

- If the labeling status is set to **Labeled**, ensure that the folder or manifest file complies with the format specifications. Otherwise, the import may fail.
- After the import of labeled data, check whether the imported data is in the labeled state.

Importing a Table Dataset from OBS

ModelArts allows you to import table data (CSV files) from OBS.

Import description:

- The prerequisite for successful import is that the schema of the data source must be the same as that specified during dataset creation. The schema indicates column names and types of a table. Once specified during dataset creation, the values cannot be changed.
- When a CSV file is imported from OBS, the data type is not validated, but the number of columns must be the same as that in the schema of the dataset. If the data format is invalid, the data is set to null. For details, see [Table 3-4](#).
- You must select the directory where the CSV file is stored. The number of columns in the CSV file must be the same as that in the dataset schema. The schema of the CSV file can be automatically obtained.

```
dataset-import-example
table_import_1.csv
table_import_2.csv
table_import_3.csv
table_import_4.csv
```

4.2.3 Specifications for Importing Data from an OBS Directory

When importing data from OBS, the data storage directory and file name must comply with the ModelArts specifications.

Only the following labeling types of data can be imported by **Labeling Format**: image classification, object detection, image segmentation, text classification, and sound classification.

A table dataset can import data from sources such as OBS, GaussDB(DWS), DLI, and MRS.

 **NOTE**

- To import data from an OBS directory, you must have the read permission on the OBS directory.
- The OBS buckets and ModelArts must be in the same region.

Image Classification

Data for image classification can be stored in two formats:

- **Format 1: ModelArts imageNet 1.0 (supporting a single label)**
Images with the same label must be stored in the same directory, with the label name as the directory name. If there are multiple levels of directories, the last level is used as the label name.

In the following example, **Cat** and **Dog** are label names.

```
dataset-import-example
├── Cat
│   ├── 10.jpg
│   ├── 11.jpg
│   └── 12.jpg
└── Dog
    ├── 1.jpg
    ├── 2.jpg
    └── 3.jpg
```

- **Format 2: ModelArts image classification 1.0 (supporting multiple labels)**
The image and label files must be stored in the same directory, with the content in the label file used as the label of the image.

In the following example, **import-dir-1** and **import-dir-2** are the imported subdirectories:

```
dataset-import-example
├── import-dir-1
│   ├── 10.jpg
│   ├── 10.txt
│   ├── 11.jpg
│   ├── 11.txt
│   ├── 12.jpg
│   └── 12.txt
└── import-dir-2
    ├── 1.jpg
    ├── 1.txt
    ├── 2.jpg
    └── 2.txt
```

- The following shows a label file for a single label, for example, the **1.txt** file:
Cat
- The following shows a label file for multiple labels, for example, the **2.txt** file:
Cat
Dog

Only images in JPG, JPEG, PNG, and BMP formats are supported. The size of a single image cannot exceed 5 MB, and the total size of all images uploaded at a time cannot exceed 8 MB.

Object Detection

Data for object detection can be stored in two formats:

- **ModelArts PASCAL VOC 1.0**
The simple mode of object detection requires you to store labeled objects and your label files (in one-to-one relationship with the labeled objects) in the same directory. For example, if the name of the labeled object file is **IMG_20180919_114745.jpg**, the name of the label file must be **IMG_20180919_114745.xml**.
The label files must be in PASCAL VOC format. For details about the format, see [Table 4-9](#).

Example:

```
dataset-import-example
  IMG_20180919_114732.jpg
  IMG_20180919_114732.xml
  IMG_20180919_114745.jpg
  IMG_20180919_114745.xml
  IMG_20180919_114945.jpg
  IMG_20180919_114945.xml
```

A label file example is as follows:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<annotation>
  <folder>NA</folder>
  <filename>bike_1_1593531469339.png</filename>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>554</width>
    <height>606</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>Dog</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <occluded>0</occluded>
    <bndbox>
      <xmin>279</xmin>
      <ymin>52</ymin>
      <xmax>474</xmax>
      <ymin>278</ymin>
    </bndbox>
  </object>
  <object>
    <name>Cat</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <occluded>0</occluded>
    <bndbox>
      <xmin>279</xmin>
      <ymin>198</ymin>
      <xmax>456</xmax>
      <ymin>421</ymin>
    </bndbox>
  </object>
</annotation>
```

Only images in JPG, JPEG, PNG, and BMP formats are supported. A single image cannot exceed 5 MB, and the total size of all images uploaded at a time cannot exceed 8 MB.

- YOLO:

A YOLO dataset must comply with the following structure:

For a YOLO dataset, all file names in the dataset cannot contain special characters such as underscores (_). Otherwise, the training task may fail.

```
yolo_dataset/
├── obj.names # Label set file
├── obj.data # Files and relative paths for recording dataset information
├── train.txt # Relative path of images in the training set
├── valid.txt # Relative path of images in the validation set
└── obj_train_data/ # Directory where the images in the training set and the corresponding label
```



```
files are stored
├── image1.txt # BBox label list for image 1
├── image1.jpg
├── image2.txt
├── image2.jpg
├── ...
└── obj_valid_data/ # Directory where the images in the validation set and the corresponding
    label files are stored
        ├── image101.txt
        ├── image101.jpg
        ├── image102.txt
        ├── image102.jpg
        └── ...
```

A YOLO dataset supports only training sets and validation sets. If other sets are imported, they will be invalid in the YOLO dataset.

- **obj.data** contains the following content and at least one of the **train** and **valid** subsets must be contained. The file paths are relative paths.

```
classes = 5 # Optional
names = <path/to/obj.names># For example, obj.names
train = <path/to/train.txt># For example, train.txt
valid = <path/to/valid.txt># Optional, for example, valid.txt
backup = backup/ # Optional
```

- The **obj.names** file records the label list. Each row label is used as the file index.

```
label1 # index of label 1: 0
label2 # index of label 2: 1
label3
...
```

- The file paths in **train.txt** and **valid.txt** are relative paths, and the file list must be in one-to-one relationship with the files in the directories. The file structures of the two files are as follows:

```
<path/to/image1.jpg># For example, obj_train_data/image.jpg
<path/to/image2.jpg># For example, obj_train_data/image.jpg
...
```

- The .txt files in the **obj_train_data/** and **obj_valid_data/** directories contain the BBox label information of the corresponding images. Each line indicates a BBox label.

```
# image1.txt:
# <label_index> <x_center> <y_center> <width> <height>
0 0.250000 0.400000 0.300000 0.400000
3 0.600000 0.400000 0.400000 0.266667
```

x_center, **y_center**, **width**, and **height** indicate the normalized parameters for the target bounding box: the x-coordinate and y-coordinate of the center point, width, and height.

Only images in JPG, JPEG, PNG, and BMP formats are supported. A single image cannot exceed 5 MB, and the total size of all images uploaded at a time cannot exceed 8 MB.

Image Segmentation

ModelArts image segmentation 1.0:

- Labeled objects and their label files (in one-to-one relationship with the labeled objects) must be in the same directory. For example, if the name of the labeled object file is **IMG_20180919_114746.jpg**, the name of the label file must be **IMG_20180919_114746.xml**.

Fields **mask_source** and **mask_color** are added to the label file in PASCAL VOC format. For details about the format, see [Table 4-5](#).

Example:

```
dataset-import-example
IMG_20180919_114732.jpg
IMG_20180919_114732.xml
IMG_20180919_114745.jpg
IMG_20180919_114745.xml
IMG_20180919_114945.jpg
IMG_20180919_114945.xml
```

A label file example is as follows:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<annotation>
  <folder>NA</folder>
  <filename>image_0006.jpg</filename>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>230</width>
    <height>300</height>
    <depth>3</depth>
  </size>
  <segmented>1</segmented>
  <mask_source>obs://xianao/out/dataset-8153-Jmf5yLjRmSacj9KevS/annotation/V001/segmentationClassRaw/image_0006.png</mask_source>
  <object>
    <name>bike</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <mask_color>193,243,53</mask_color>
    <occluded>0</occluded>
    <polygon>
      <x1>71</x1>
      <y1>48</y1>
      <x2>75</x2>
      <y2>73</y2>
      <x3>49</x3>
      <y3>69</y3>
      <x4>68</x4>
      <y4>92</y4>
      <x5>90</x5>
      <y5>101</y5>
      <x6>45</x6>
      <y6>110</y6>
      <x7>71</x7>
      <y7>48</y7>
    </polygon>
  </object>
</annotation>
```

Text Classification

txt and csv files can be imported for text classification, with the text encoding format of UTF-8 or GBK.

Labeled objects and labels for text classification can be stored in two formats:

- ModelArts text classification combine 1.0: The labeled objects and labels for text classification are in the same text file. You can specify a separator to separate the labeled objects and labels, as well as multiple labels.

For example, the following shows an example text file. The **Tab** key is used to separate the labeled objects from the labels.

```
It touches good and responds quickly. I don't know how it performs in the future. positive
Three months ago, I bought a very good phone and replaced my old one with it. It can operate longer
between charges. positive
```

```
Why does my phone heat up if I charge it for a while? The volume button stuck after being pressed down. negative  
It's a gift for Father's Day. The delivery is fast and I received it in 24 hours. I like the earphones because the bass sounds feel good and they would not fall off. positive
```

- **ModelArts text classification 1.0:** The labeled objects and labels for text classification are text files, and correspond to each other based on the rows. For example, the first row in a label file indicates the label of the first row in the file of the labeled object.

For example, the content of the labeled object **COMMENTS_20180919_114745.txt** is as follows:

```
It touches good and responds quickly. I don't know how it performs in the future.  
Three months ago, I bought a very good phone and replaced my old one with it. It can operate longer between charges.  
Why does my phone heat up if I charge it for a while? The volume button stuck after being pressed down.  
It's a gift for Father's Day. The delivery is fast and I received it in 24 hours. I like the earphones because the bass sounds feel good and they would not fall off.
```

The content of the label file **COMMENTS_20180919_114745_result.txt** is as follows:

```
positive  
negative  
negative  
positive
```

This data format requires you to store labeled objects and your label files (in one-to-one relationship with the labeled objects) in the same directory. For example, if the name of the labeled object file is

COMMENTS_20180919_114745.txt, the name of the label file must be **COMMENTS_20180919_114745_result.txt**.

Example of data files:

```
dataset-import-example  
├── COMMENTS_20180919_114732.txt  
├── COMMENTS_20180919_114732_result.txt  
├── COMMENTS_20180919_114745.txt  
├── COMMENTS_20180919_114745_result.txt  
├── COMMENTS_20180919_114945.txt  
└── COMMENTS_20180919_114945_result.txt
```

Sound Classification

ModelArts audio classification dir 1.0: Sound files with the same label must be stored in the same directory, and the label name is the directory name.

Example:

```
dataset-import-example  
├── Cat  
│   ├── 10.wav  
│   ├── 11.wav  
│   └── 12.wav  
└── Dog  
    ├── 1.wav  
    ├── 2.wav  
    └── 3.wav
```

Tables

CSV files can be imported from OBS. Select the directory where the files are stored. The number of columns in the CSV file must be the same as that in the dataset schema. The schema of the CSV file can be automatically obtained.

```
dataset-import-example
table_import_1.csv
table_import_2.csv
table_import_3.csv
table_import_4.csv
```

4.2.4 Importing a Manifest File

Prerequisites

- You have created a dataset.
- You have stored the data to be imported in OBS. You have stored the manifest file in OBS.
- The OBS bucket and ModelArts are in the same region and you can operate the bucket.

Importing File Data from a Manifest File

The parameters for data import vary according to the dataset type. The following uses an image dataset as an example.

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. Locate the row that contains the desired dataset and click **Import** in the **Operation** column. Alternatively, you can click the dataset name to go to the **Dashboard** tab page of the dataset, and click **Import** in the upper right corner.
3. In the **Import** dialog box, set the parameters as follows and click **OK**.
 - **Data Source:** OBS
 - **Import Mode:** manifest
 - **Manifest File:** OBS path for storing the manifest file
 - **Labeling Status:** Labeled
 - **Advanced Feature Settings:** disabled by default

Import by Tag The system automatically obtains the labels of the dataset. You can click **Add Label** to add a label. This parameter is optional. If **Import by Tag** is disabled, you can add or delete labels for imported data when labeling data.

Import Only Hard Examples: If this parameter is selected, only the **hard** attribute data of the manifest file is imported.

Figure 4-6 Importing a manifest file

Import

* Data Source: **OBS** Local file

* Import Mode: Path **manifest**

The manifest file needs to define the mapping between labeling objects and content. [Manifest file specifications and examples](#)

* Manifest File: Select the directory where the manifest file resides.

You can import up to 199,999 samples and 100,000 labels.

* Labeling Status: Unlabeled **Labeled**

Advanced Feature Settings

Import Only Hard Examples

Import by Tag

OK Cancel

After the data is imported, it will be automatically synchronized to the dataset. On the **Datasets** page, click the dataset name to view its details and create a labeling job to label the data.

Labeling Status of File Data

The labeling status can be **Unlabeled** or **Labeled**.

- **Unlabeled:** Only the labeling object (such as unlabeled images or texts) is imported.
- **Labeled:** Both the labeling object and content are imported. Labeling content importing is not supported for datasets in free format.

To ensure that the labeling content can be correctly read, you must store data in strict accordance with the specifications.

- If **Import Mode** is set to **Path**, store the data to be imported according to the labeling file specifications.
- If **Import Mode** is set to **manifest**, the manifest file specifications must be met. For details, see [Specifications for Importing a Manifest File](#).

NOTE

If the labeling status is set to **Labeled**, ensure that the folder or manifest file complies with the format specifications. Otherwise, the import may fail.

4.2.5 Specifications for Importing a Manifest File

The manifest file defines the mapping between labeled objects and content. The manifest file import mode means that the manifest file is used for dataset import. The manifest file can be imported from OBS. When importing a manifest file from

OBS, ensure that you have the permissions to access the directory where the manifest file is stored.

NOTE

There are many requirements on the manifest file compilation. Import new data from OBS. Generally, manifest file import is used for data migration of ModelArts in different regions or using different accounts. If you have labeled data in a region using ModelArts, you can obtain the manifest file of the published dataset from the output path. Then you can import the dataset using the manifest file to ModelArts of other regions or accounts. The imported data carries the labeling information and does not need to be labeled again, improving development efficiency.

The manifest file that contains information about the original file and labeling can be used in labeling, training, and inference scenarios. The manifest file that contains only information about the original file can be used in inference scenarios or used to generate an unlabeled dataset. The manifest file must meet the following requirements:

- The manifest file uses the UTF-8 encoding format.
- The manifest file uses the JSON Lines format (jsonlines.org). A line contains one JSON object.

```
{"source": "/path/to/image1.jpg", "annotation": ... }  
{"source": "/path/to/image2.jpg", "annotation": ... }  
{"source": "/path/to/image3.jpg", "annotation": ... }
```

In the preceding example, the manifest file contains multiple lines of JSON object.

- The manifest file can be generated by you, third-party tools, or ModelArts Data Labeling. The file name can be any valid file name. To facilitate the internal use of the ModelArts system, the file name generated by the ModelArts data labeling function consists of the following strings: **DatasetName-VersionName.manifest**. For example, **animal-v201901231130304123.manifest**.

Image Classification

```
{  
  "source": "s3://path/to/image1.jpg",  
  "usage": "TRAIN",  
  "hard": "true",  
  "hard-coefficient": 0.8,  
  "id": "0162005993f8065ef47eefb59d1e4970",  
  "annotation": [  
    {  
      "type": "modelarts/image_classification",  
      "name": "cat",  
      "property": {  
        "color": "white",  
        "kind": "Persian cat"  
      },  
      "hard": "true",  
      "hard-coefficient": 0.8,  
      "annotated-by": "human",  
      "creation-time": "2019-01-23 11:30:30"  
    },  
    {  
      "type": "modelarts/image_classification",  
      "name": "animal",  
      "annotated-by": "modelarts/active-learning",  
      "confidence": 0.8,  
      "creation-time": "2019-01-23 11:30:30"  
    }  
  ]  
}
```

```

    }},
    "inference-loc": "/path/to/inference-output"
}

```

Table 4-2 Parameters

Parameter	Mandatory	Description
source	Yes	URI of an object to be labeled. For details about data source types and examples, see Table 4-3 .
usage	No	By default, the parameter value is left blank. Possible values are as follows: <ul style="list-style-type: none"> • TRAIN: The object is used for training. • EVAL: The object is used for evaluation. • TEST: The object is used for testing. • INFERENCE: The object is used for inference. If the parameter value is left blank, you decide how to use the object.
id	No	Sample ID exported from the system. You do not need to set this parameter when importing the sample.
annotation	No	If the parameter value is left blank, the object is not labeled. The value of annotation consists of an object list. For details about the parameters, see Table 4-4 .
inference-loc	No	This parameter is available when the file is generated by the inference service, indicating the location of the inference result file.

Table 4-3 Data source types

Type	Example
OBS	"source": "s3://path-to-jpg"
Content	"source": "content://I love machine learning"

Table 4-4 annotation objects

Parameter	Mandatory	Description
type	Yes	Label type. Possible values are as follows: <ul style="list-style-type: none"> ● image_classification: image classification ● text_classification: text classification ● text_entity: named entity recognition ● object_detection: object detection ● audio_classification: sound classification ● audio_content: speech labeling ● audio_segmentation: speech paragraph labeling
name	Yes/No	This parameter is mandatory for the classification type but optional for other types. This example uses the image classification type.
id	Yes/No	Label ID. This parameter is mandatory for triplets but optional for other types. The entity label ID of a triplet is in E+number format, for example, E1 and E2 . The relationship label ID of a triplet is in R+number format, for example, R1 and R2 .
property	No	Labeling property. In this example, the cat has two properties: color and kind.
hard	No	Indicates whether the example is a hard example. True indicates that the labeling example is a hard example, and False indicates that the labeling example is not a hard example.
annotated-by	No	The default value is human , indicating manual labeling. <ul style="list-style-type: none"> ● human
creation-time	No	Time when the labeling job was created. It is the time when labeling information was written, not the time when the manifest file was generated.
confidence	No	Confidence score of machine labeling. The value ranges from 0 to 1.

Image Segmentation

```
{
  "annotation": [{
    "annotation-format": "PASCAL VOC",
    "type": "modelarts/image_segmentation",
    "annotation-loc": "s3://path/to/annotation/image1.xml",
    "creation-time": "2020-12-16 21:36:27",
    "annotated-by": "human"
  }],
  "usage": "train",
```



```
"source": "s3://path/to/image1.jpg",
"id": "16d196c19bf61994d7deccafa435398c",
"sample-type": 0
}
```

- The parameters such as **source**, **usage**, and **annotation** are the same as those described in [Image Classification](#). For details, see [Table 4-2](#).
- **annotation-loc** indicates the path for saving the label file. This parameter is mandatory for image segmentation and object detection but optional for other labeling types.
- **annotation-format** indicates the format of the label file. This parameter is optional. The default value is **PASCAL VOC**. Only **PASCAL VOC** is supported.
- **sample-type** indicates a sample format. Value **0** indicates image, **1** text, **2** audio, **4** table, and **6** video.

Table 4-5 PASCAL VOC format parameters

Parameter	Mandatory	Description
folder	Yes	Directory where the data source is located
filename	Yes	Name of the file to be labeled
size	Yes	Image pixel <ul style="list-style-type: none"> • width: image width. This parameter is mandatory. • height: image height. This parameter is mandatory. • depth: number of image channels. This parameter is mandatory.
segmented	Yes	Segmented or not
mask_source	No	Segmentation mask path

Parameter	Mandatory	Description
object	Yes	<p>Object detection information. Multiple object{} functions are generated for multiple objects.</p> <ul style="list-style-type: none"> • name: type of the labeled content. This parameter is mandatory. • pose: shooting angle of the labeled content. This parameter is mandatory. • truncated: whether the labeled content is truncated (0 indicates that the content is not truncated). This parameter is mandatory. • occluded: whether the labeled content is occluded (0 indicates that the content is not occluded). This parameter is mandatory. • difficult: whether the labeled object is difficult to identify (0 indicates that the object is easy to identify). This parameter is mandatory. • confidence: confidence score of the labeled object. The value ranges from 0 to 1. This parameter is optional. • bndbox: bounding box type. This parameter is mandatory. For details about the possible values, see Table 4-6. • mask_color: label color, which is represented by the RGB value. This parameter is mandatory.

Table 4-6 Bounding box types

Parameter	Shape	Labeling information
polygon	Polygon	Coordinates of points <x1>100<x1> <y1>100<y1> <x2>200<x2> <y2>100<y2> <x3>250<x3> <y3>150<y3> <x4>200<x4> <y4>200<y4> <x5>100<x5> <y5>200<y5> <x6>50<x6> <y6>150<y6> <x7>100<x7> <y7>100<y7>

Example:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<annotation>
  <folder>NA</folder>
  <filename>image_0006.jpg</filename>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>230</width>
    <height>300</height>
    <depth>3</depth>
  </size>
  <segmented>1</segmented>
  <mask_source>obs://xianao/out/dataset-8153-Jmf5ylljRmSacj9KevS/annotation/V001/segmentationClassRaw/image_0006.png</mask_source>
  <object>
    <name>bike</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <mask_color>193,243,53</mask_color>
    <occluded>0</occluded>
    <polygon>
      <x1>71</x1>
      <y1>48</y1>
      <x2>75</x2>
      <y2>73</y2>
      <x3>49</x3>
      <y3>69</y3>
      <x4>68</x4>
      <y4>92</y4>
      <x5>90</x5>
      <y5>101</y5>
      <x6>45</x6>
      <y6>110</y6>
    </polygon>
  </object>
</annotation>
```

```
<x7>71</x7>  
<y7>48</y7>  
</polygon>  
</object>  
</annotation>
```

Text Classification

```
{  
  "source": "content://I like this product ",  
  "id": "XGDVGS",  
  "annotation": [  
    {  
      "type": "modelarts/text_classification",  
      "name": " positive",  
      "annotated-by": "human",  
      "creation-time": "2019-01-23 11:30:30"  
    }  
  ]  
}
```

The **content** parameter indicates the text to be labeled (in UTF-8 encoding format, which can be Chinese). The other parameters are the same as those described in [Image Classification](#). For details, see [Table 4-2](#).

Named Entity Recognition

```
{  
  "source": "content://Michael Jordan is the most famous basketball player in the world.",  
  "usage": "TRAIN",  
  "annotation": [  
    {  
      "type": "modelarts/text_entity",  
      "name": "Person",  
      "property": {  
        "@modelarts:start_index": 0,  
        "@modelarts:end_index": 14  
      },  
      "annotated-by": "human",  
      "creation-time": "2019-01-23 11:30:30"  
    },  
    {  
      "type": "modelarts/text_entity",  
      "name": "Category",  
      "property": {  
        "@modelarts:start_index": 34,  
        "@modelarts:end_index": 44  
      },  
      "annotated-by": "human",  
      "creation-time": "2019-01-23 11:30:30"  
    }  
  ]  
}
```

The parameters such as **source**, **usage**, and **annotation** are the same as those described in [Image Classification](#). For details, see [Table 4-2](#).

[Table 4-7](#) describes the property parameters. For example, if you want to extract **Michael** from "**source**": "**content**://**Michael Jordan**", the value of **start_index** is **0** and that of **end_index** is **7**.

Table 4-7 property parameters

Parameter	Data type	Description
@modelarts:start_index	Integer	Start position of the text. The value starts from 0, including the characters specified by start_index .
@modelarts:end_index	Integer	End position of the text, excluding the characters specified by end_index .

Text Triplet

```
{
  "source":"content://"Three Body" is a series of long science fiction novels created by Liu Cix.",
  "usage":"TRAIN",
  "annotation":[
    {
      "type":"modelarts/text_entity",
      "name":"Person",
      "id":"E1",
      "property":{
        "@modelarts:start_index":67,
        "@modelarts:end_index":74
      },
      "annotated-by":"human",
      "creation-time":"2019-01-23 11:30:30"
    },
    {
      "type":"modelarts/text_entity",
      "name":"Book",
      "id":"E2",
      "property":{
        "@modelarts:start_index":0,
        "@modelarts:end_index":12
      },
      "annotated-by":"human",
      "creation-time":"2019-01-23 11:30:30"
    },
    {
      "type":"modelarts/text_triplet",
      "name":"Author",
      "id":"R1",
      "property":{
        "@modelarts:from":"E1",
        "@modelarts:to":"E2"
      },
      "annotated-by":"human",
      "creation-time":"2019-01-23 11:30:30"
    },
    {
      "type":"modelarts/text_triplet",
      "name":"Works",
      "id":"R2",
      "property":{
        "@modelarts:from":"E2",
        "@modelarts:to":"E1"
      },
      "annotated-by":"human",
      "creation-time":"2019-01-23 11:30:30"
    }
  ]
}
```

The parameters such as **source**, **usage**, and **annotation** are the same as those described in [Image Classification](#). For details, see [Table 4-2](#).

[Table 5 property parameters](#) describes the **property** parameters.

@modelarts:start_index and **@modelarts:end_index** are the same as those of named entity recognition. For example, when **source** is set to **content://"Three Body" is a series of long science fiction novels created by Liu Cix.**, **Liu Cix** is an entity person, **Three Body** is an entity book, the person is the author of the book, and the book is works of the person.

Table 4-8 property parameters

Parameter	Data type	Description
@modelarts:start_index	Integer	Start position of the triplet entities. The value starts from 0, including the characters specified by start_index .
@modelarts:end_index	Integer	End position of the triplet entities, excluding the characters specified by end_index .
@modelarts:from	String	Start entity ID of the triplet relationship.
@modelarts:to	String	Entity ID pointed to in the triplet relationship.

Object Detection

```
{
  "source":"s3://path/to/image1.jpg",
  "usage":"TRAIN",
  "hard":"true",
  "hard-coefficient":0.8,
  "annotation": [
    {
      "type":"modelarts/object_detection",
      "annotation-loc": "s3://path/to/annotation1.xml",
      "annotation-format":"PASCAL VOC",
      "annotated-by":"human",
      "creation-time":"2019-01-23 11:30:30"
    }
  ]
}
```

- The parameters such as **source**, **usage**, and **annotation** are the same as those described in [Image Classification](#). For details, see [Table 4-2](#).
- **annotation-loc** indicates the path for saving the label file. This parameter is mandatory for object detection and image segmentation but optional for other labeling types.
- **annotation-format** indicates the format of the label file. This parameter is optional. The default value is **PASCAL VOC**. Only **PASCAL VOC** is supported.

Table 4-9 PASCAL VOC format parameters

Parameter	Mandatory	Description
folder	Yes	Directory where the data source is located
filename	Yes	Name of the file to be labeled
size	Yes	Image pixel <ul style="list-style-type: none"> • width: image width. This parameter is mandatory. • height: image height. This parameter is mandatory. • depth: number of image channels. This parameter is mandatory.
segmented	Yes	Segmented or not
object	Yes	Object detection information. Multiple object{} functions are generated for multiple objects. <ul style="list-style-type: none"> • name: type of the labeled content. This parameter is mandatory. • pose: shooting angle of the labeled content. This parameter is mandatory. • truncated: whether the labeled content is truncated (0 indicates that the content is not truncated). This parameter is mandatory. • occluded: whether the labeled content is occluded (0 indicates that the content is not occluded). This parameter is mandatory. • difficult: whether the labeled object is difficult to identify (0 indicates that the object is easy to identify). This parameter is mandatory. • confidence: confidence score of the labeled object. The value ranges from 0 to 1. This parameter is optional. • bndbox: bounding box type. This parameter is mandatory. For details about the possible values, see Table 4-10.

Table 4-10 Bounding box types

Parameter	Shape	Labeling information
point	Point	Coordinates of a point <x>100<x> <y>100<y>

Parameter	Shape	Labeling information
line	Line	Coordinates of points <x1>100<x1> <y1>100<y1> <x2>200<x2> <y2>200<y2>
bndbox	Rectangle	Coordinates of the upper left and lower right points <xmin>100<xmin> <ymin>100<ymin> <xmax>200<xmax> <ymin>200<ymin>
polygon	Polygon	Coordinates of points <x1>100<x1> <y1>100<y1> <x2>200<x2> <y2>100<y2> <x3>250<x3> <y3>150<y3> <x4>200<x4> <y4>200<y4> <x5>100<x5> <y5>200<y5> <x6>50<x6> <y6>150<y6>
circle	Circle	Center coordinates and radius <cx>100<cx> <cy>100<cy> <r>50<r>

Example:

```
<annotation>
  <folder>test_data</folder>
  <filename>260730932.jpg</filename>
  <size>
    <width>767</width>
    <height>959</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>point</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
```



```
<occluded>0</occluded>
<difficult>0</difficult>
<point>
  <x1>456</x1>
  <y1>596</y1>
</point>
</object>
<object>
  <name>line</name>
  <pose>Unspecified</pose>
  <truncated>0</truncated>
  <occluded>0</occluded>
  <difficult>0</difficult>
  <line>
    <x1>133</x1>
    <y1>651</y1>
    <x2>229</x2>
    <y2>561</y2>
  </line>
</object>
<object>
  <name>bag</name>
  <pose>Unspecified</pose>
  <truncated>0</truncated>
  <occluded>0</occluded>
  <difficult>0</difficult>
  <bndbox>
    <xmin>108</xmin>
    <ymin>101</ymin>
    <xmax>251</xmax>
    <ymin>238</ymin>
  </bndbox>
</object>
<object>
  <name>boots</name>
  <pose>Unspecified</pose>
  <truncated>0</truncated>
  <occluded>0</occluded>
  <difficult>0</difficult>
  <hard-coefficient>0.8</hard-coefficient>
  <polygon>
    <x1>373</x1>
    <y1>264</y1>
    <x2>500</x2>
    <y2>198</y2>
    <x3>437</x3>
    <y3>76</y3>
    <x4>310</x4>
    <y4>142</y4>
  </polygon>
</object>
<object>
  <name>circle</name>
  <pose>Unspecified</pose>
  <truncated>0</truncated>
  <occluded>0</occluded>
  <difficult>0</difficult>
  <circle>
    <cx>405</cx>
    <cy>170</cy>
    <r>100</r>
  </circle>
</object>
</annotation>
```

Sound Classification

```
{
  "source":
```

```
"s3://path/to/pets.wav",
  "annotation": [
    {
      "type": "modelarts/audio_classification",
      "name": "cat",
      "annotated-by": "human",
      "creation-time": "2019-01-23 11:30:30"
    }
  ]
}
```

The parameters such as **source**, **usage**, and **annotation** are the same as those described in [Image Classification](#). For details, see [Table 4-2](#).

Speech Labeling

```
{
  "source": "s3://path/to/audio1.wav",
  "annotation": [
    {
      "type": "modelarts/audio_content",
      "property": {
        "@modelarts:content": "Today is a good day."
      },
      "annotated-by": "human",
      "creation-time": "2019-01-23 11:30:30"
    }
  ]
}
```

- The parameters such as **source**, **usage**, and **annotation** are the same as those described in [Image Classification](#). For details, see [Table 4-2](#).
- The **@modelarts:content** parameter in **property** indicates speech content. The data type is **String**.

Speech Paragraph Labeling

```
{
  "source": "s3://path/to/audio1.wav",
  "usage": "TRAIN",
  "annotation": [
    {
      "type": "modelarts/audio_segmentation",
      "property": {
        "@modelarts:start_time": "00:01:10.123",
        "@modelarts:end_time": "00:01:15.456",
        "@modelarts:source": "Tom",
        "@modelarts:content": "How are you?"
      },
      "annotated-by": "human",
      "creation-time": "2019-01-23 11:30:30"
    },
    {
      "type": "modelarts/audio_segmentation",
      "property": {
        "@modelarts:start_time": "00:01:22.754",
        "@modelarts:end_time": "00:01:24.145",
        "@modelarts:source": "Jerry",
        "@modelarts:content": "I'm fine, thank you."
      },
      "annotated-by": "human",
      "creation-time": "2019-01-23 11:30:30"
    }
  ]
}
```

- The parameters such as **source**, **usage**, and **annotation** are the same as those described in **Image Classification**. For details, see **Table 4-2**.
- **Table 4-11** describes the **property** parameters.

Table 4-11 **property** parameters

Parameter	Data type	Description
@modelarts:start_time	String	Start time of the sound. The format is hh:mm:ss.SSS . hh indicates the hour, mm indicates the minute, ss indicates the second, and SSS indicates the millisecond.
@modelarts:end_time	String	End time of the sound. The format is hh:mm:ss.SSS . hh indicates the hour, mm indicates the minute, ss indicates the second, and SSS indicates the millisecond.
@modelarts:source	String	Sound source
@modelarts:content	String	Sound content

Video Labeling

```
{
  "annotation": [{
    "annotation-format": "PASCAL VOC",
    "type": "modelarts/object_detection",
    "annotation-loc": "s3://path/to/annotation1_t1.473722.xml",
    "creation-time": "2020-10-09 14:08:24",
    "annotated-by": "human"
  }],
  "usage": "train",
  "property": {
    "@modelarts:parent_duration": 8,
    "@modelarts:parent_source": "s3://path/to/annotation1.mp4",
    "@modelarts:time_in_video": 1.473722
  },
  "source": "s3://input/path/to/annotation1_t1.473722.jpg",
  "id": "43d88677c1e9a971eeb692a80534b5d5",
  "sample-type": 0
}
```

- The parameters such as **source**, **usage**, and **annotation** are the same as those described in **Image Classification**. For details, see **Table 4-2**.
- **annotation-loc** indicates the path for saving the label file. This parameter is mandatory for object detection but optional for other labeling types.
- **annotation-format** indicates the format of the label file. This parameter is optional. The default value is **PASCAL VOC**. Only **PASCAL VOC** is supported.
- **sample-type** indicates a sample format. Value **0** indicates image, **1** text, **2** audio, **4** table, and **6** video.

Table 4-12 property parameters

Parameter	Data type	Description
@modelarts:parent_duration	Double	Duration of the labeled video, in seconds
@modelarts:time_in_video	Double	Timestamp of the labeled video frame, in seconds
@modelarts:parent_source	String	OBS path of the labeled video

Table 4-13 PASCAL VOC format parameters

Parameter	Mandatory	Description
folder	Yes	Directory where the data source is located
filename	Yes	Name of the file to be labeled
size	Yes	Image pixel <ul style="list-style-type: none"> • width: image width. This parameter is mandatory. • height: image height. This parameter is mandatory. • depth: number of image channels. This parameter is mandatory.
segmented	Yes	Segmented or not

Parameter	Mandatory	Description
object	Yes	<p>Object detection information. Multiple object{} functions are generated for multiple objects.</p> <ul style="list-style-type: none"> • name: type of the labeled content. This parameter is mandatory. • pose: shooting angle of the labeled content. This parameter is mandatory. • truncated: whether the labeled content is truncated (0 indicates that the content is not truncated). This parameter is mandatory. • occluded: whether the labeled content is occluded (0 indicates that the content is not occluded). This parameter is mandatory. • difficult: whether the labeled object is difficult to identify (0 indicates that the object is easy to identify). This parameter is mandatory. • confidence: confidence score of the labeled object. The value ranges from 0 to 1. This parameter is optional. • bndbox: bounding box type. This parameter is mandatory. For details about the possible values, see Table 4-14.

Table 4-14 Bounding box types

Parameter	Shape	Labeling information
point	Point	<p>Coordinates of a point</p> <pre><x>100<x> <y>100<y></pre>
line	Line	<p>Coordinates of points</p> <pre><x1>100<x1> <y1>100<y1> <x2>200<x2> <y2>200<y2></pre>
bndbox	Rectangle	<p>Coordinates of the upper left and lower right points</p> <pre><xmin>100<xmin> <ymin>100<ymin> <xmax>200<xmax> <ymin>200<ymin></pre>

Parameter	Shape	Labeling information
polygon	Polygon	Coordinates of points <x1>100<x1> <y1>100<y1> <x2>200<x2> <y2>100<y2> <x3>250<x3> <y3>150<y3> <x4>200<x4> <y4>200<y4> <x5>100<x5> <y5>200<y5> <x6>50<x6> <y6>150<y6>
circle	Circle	Center coordinates and radius <cx>100<cx> <cy>100<cy> <r>50<r>

Example:

```

<annotation>
  <folder>test_data</folder>
  <filename>260730932_t1.473722.jpg.jpg</filename>
  <size>
    <width>767</width>
    <height>959</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>point</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <occluded>0</occluded>
    <difficult>0</difficult>
    <point>
      <x1>456</x1>
      <y1>596</y1>
    </point>
  </object>
  <object>
    <name>line</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <occluded>0</occluded>
    <difficult>0</difficult>
    <line>
      <x1>133</x1>
      <y1>651</y1>
      <x2>229</x2>
      <y2>561</y2>
    </line>
  </object>
</annotation>
    
```

```
<object>
  <name>bag</name>
  <pose>Unspecified</pose>
  <truncated>0</truncated>
  <occluded>0</occluded>
  <difficult>0</difficult>
  <bndbox>
    <xmin>108</xmin>
    <ymin>101</ymin>
    <xmax>251</xmax>
    <ymax>238</ymax>
  </bndbox>
</object>
<object>
  <name>boots</name>
  <pose>Unspecified</pose>
  <truncated>0</truncated>
  <occluded>0</occluded>
  <difficult>0</difficult>
  <hard-coefficient>0.8</hard-coefficient>
  <polygon>
    <x1>373</x1>
    <y1>264</y1>
    <x2>500</x2>
    <y2>198</y2>
    <x3>437</x3>
    <y3>76</y3>
    <x4>310</x4>
    <y4>142</y4>
  </polygon>
</object>
<object>
  <name>circle</name>
  <pose>Unspecified</pose>
  <truncated>0</truncated>
  <occluded>0</occluded>
  <difficult>0</difficult>
  <circle>
    <cx>405</cx>
    <cy>170</cy>
    <r>100</r>
  </circle>
</object>
</annotation>
```

4.3 Importing Data from DLI

Data importing from DLI is supported for table datasets.

To import data from DLI, select the DLI queue, database, and table name. The schema (column name and type) of the selected table must be the same as that of the dataset. The schema of the selected table can be automatically obtained.

- **Queue Name:** All DLI queues of the current account are automatically displayed. Select the required queue from the drop-down list.
- **Database Name:** All databases are displayed based on the selected queue. Select the required database from the drop-down list.
- **Table Name:** All tables in the selected database are displayed. Select the required table from the drop-down list.

 NOTE

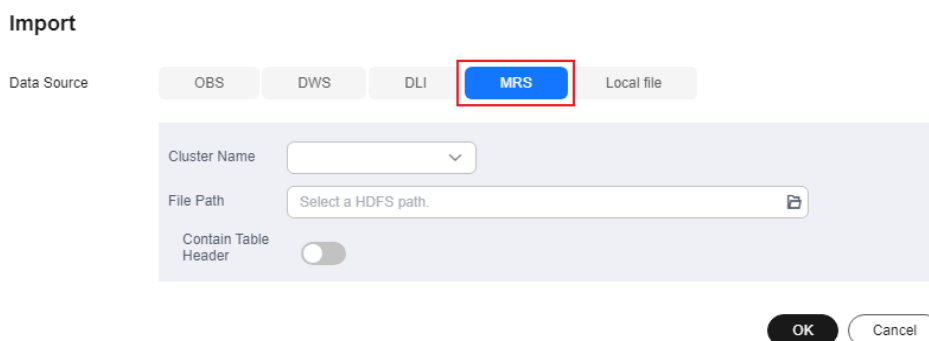
The default queue of DLI is used only for experience. Different accounts may preempt resources. Therefore, resources need to be queued. You may not be able to obtain required resources each time to perform related operations.

DLI supports schema mapping. That is, the schema field name of the imported table can be different from that of the dataset, but the type must be the same.

4.4 Importing Data from MRS

To import data in CSV format stored on HDFS from MRS, select an existing MRS cluster and select the file name or directory from the HDFS file list. The number of columns in the imported file must be the same as that of the dataset schema.

Figure 4-7 Importing data from MRS



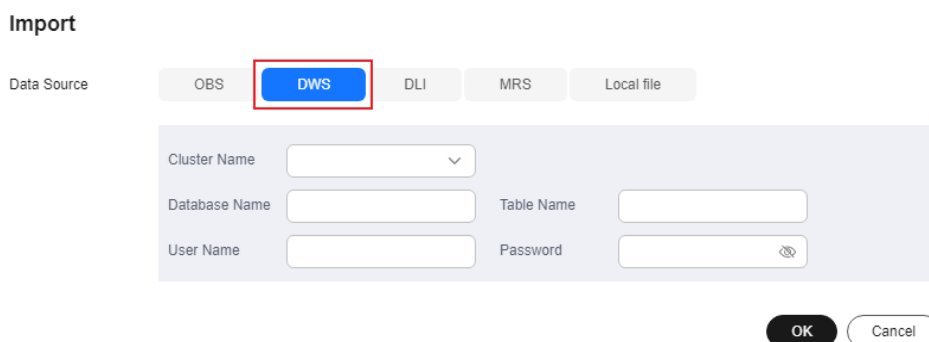
The screenshot shows the 'Import' dialog box with 'MRS' selected as the data source. The 'Data Source' section includes buttons for OBS, DWS, DLI, MRS (highlighted with a red box), and Local file. Below this, there is a 'Cluster Name' dropdown menu, a 'File Path' input field with a file selection icon, and a 'Contain Table Header' toggle switch. At the bottom right, there are 'OK' and 'Cancel' buttons.

- **Cluster Name:** All MRS clusters of the current account are automatically displayed. However, streaming clusters do not support data import. Select the required cluster from the drop-down list.
- **File Path:** Enter the HDFS file path based on the selected cluster.
- **Contain Table Header:** If this setting is enabled, the imported file contains table headers.

4.5 Importing Data from DWS

To import data from DWS, select a DWS cluster and enter the database name, table name, username, and password. The schema (column name and type) of the imported table must be the same as that of the dataset.

Figure 4-8 Importing data from DWS



The screenshot shows the 'Import' dialog box with 'DWS' selected as the data source. The 'Data Source' section includes buttons for OBS, DWS (highlighted with a red box), DLI, MRS, and Local file. Below this, there is a 'Cluster Name' dropdown menu, and four input fields: 'Database Name', 'Table Name', 'User Name', and 'Password' (with a visibility toggle icon). At the bottom right, there are 'OK' and 'Cancel' buttons.

- **Cluster Name:** All DWS clusters of the current account are automatically displayed. Select the required DWS cluster from the drop-down list.
- **Database Name:** Enter the name of the database where the data is located based on the selected DWS cluster.
- **Table Name:** Enter the name of the table where the data is located based on the selected database.
- **User Name:** Enter the username of the DWS cluster administrator.
- **Password:** Enter the password of the DWS cluster administrator.

 **NOTE**

To import data from DWS, use DLI functions. If you do not have the permission to access DLI, create a DLI agency as prompted.

4.6 Importing Data from Local Files

Prerequisites

- You have created a dataset.
- You have created an OBS bucket. The OBS bucket and ModelArts are in the same region and you can operate the bucket.

Import Operation

Both file and table data can be uploaded from local files. The data uploaded from local files should be stored in an OBS directory. You must have created an OBS bucket.

In a single batch upload, a maximum of 100 files can be uploaded at a time, and the total size of the files cannot exceed 5 GB.

The parameters on the GUI for data import vary according to the dataset type. The following uses a dataset of the image classification type as an example.

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. Locate the row that contains the desired dataset and click **Import** in the **Operation** column.


Alternatively, you can click the dataset name to go to the **Dashboard** tab page of the dataset, and click **Import** in the upper right corner.

3. In the **Import** dialog box, set the parameters as follows and click **OK**.
 - **Data Source:** **Local file**
 - **Storage Path:** Select an OBS path.
 - **Uploading Data:** Click **Upload data**, upload local data, and click **OK**.

Figure 4-9 Importing data from local files

Import

* Data Source OBS Local file

* Storage Path 

You can import up to 199,990 samples and 100,000 labels.

Uploading Data Upload data

* Labeling Status Unlabeled Labeled

5 Data Analysis and Preview

5.1 Auto Grouping

To improve the precision of auto labeling algorithms, you can evenly label multiple classes. ModelArts provides built-in grouping algorithms. You can enable auto grouping to improve data labeling efficiency.

Auto grouping can be understood as data labeling preprocessing. Clustering algorithms are used to cluster unlabeled images, and images are labeled or cleaned by group based on the clustering result.

For example, a user searches for *XX* through a search engine, downloads and uploads related images to the dataset, and then uses the auto grouping function to classify *XX* images, such as papers, posters, images confirmed as *XX*, and others. The user can quickly remove unwanted images from a group or select all images of a type and add labels to the images.

NOTE

Only datasets of image classification, object detection, and image segmentation types support the auto grouping function.

Starting Auto Grouping Tasks

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Label Data**.
2. In the labeling job list, select a labeling job of the object detection or image classification type and click the labeling job name to go to the labeling job details page.
3. On the **All statuses** tab page of the dataset details page, choose **Auto Grouping > Start Task**.

NOTE

- You can start auto group tasks or view task history only on the **All** tab page.
4. In the displayed **Auto Grouping** dialog box, set parameters and click **OK**.
 - **Groups**: Enter an integer from 2 to 200. The parameter value indicates the number of groups into which images are divided.

- **Result Processing Method:** Select **Update attribute** or **Save to OBS**.
- **Attribute Name:** If you select **Update attribute**, you need to enter an attribute name.
- **Result Storage Path:** If you select **Save to OBS**, specify an OBS path.
- **Advanced Feature Settings:** After this function is enabled, you can select **Clarity**, **Brightness**, and **Color** dimensions for the auto grouping function so that the grouping is based on the image brightness, color, and clarity. You can select multiple options.

Figure 5-1 Auto grouping

Auto Grouping

The screenshot shows a configuration dialog box for 'Auto Grouping'. It contains the following fields and options:

- * Groups:** A text input field with a placeholder. Below it, a note reads: 'Enter an integer that is less than or equal to the number of samples. If this condition is met, the value ranges from 2 to 200.'
- * Result Processing Method:** Two radio buttons: 'Update attribute' (selected) and 'Save to OBS'. Below it, a note reads: 'Add the auto grouping result to the value of an attribute, which can be used as a filter criterion.'
- * Attribute Name:** A text input field with a placeholder: 'Enter an attribute name.'
- * Advanced Feature Settings:** A toggle switch is turned on. Below it are three checkboxes: 'Clarity', 'Brightness', and 'Color', each with a help icon.
- At the bottom, there are two buttons: 'OK' (red) and 'Cancel' (white).

5. After the task is submitted, the task progress is displayed in the upper right corner of the page. After the task is complete, you can view the history of the auto grouping tasks to learn task status.

Viewing the Auto Grouping Result

On the **All** tab page of the dataset details page, expand **Filter Criteria**, set **Sample Attribute** to the attribute name of the auto grouping task, and set the sample attribute value to filter the grouping result.

Figure 5-2 Viewing the auto grouping result

Viewing Auto Grouping Task History

On the **All** tab page of the dataset details page, choose **Auto Grouping > View Task History**. In the **View Task History** dialog box, basic information about the auto grouping tasks of the current dataset is displayed.

Figure 5-3 Auto grouping task history

View Task History

If the Result Processing Method is Update attribute, you can select attribute values based on the sample attribute as a filter criterion to obtain the result. If the Result Processing Method is Save to OBS, you can view or download the grouping result in the storage path.

Created	Groups	Result Processin...	Storage Path/Att...	Status	Opera...
2020-03-13 09:20...	2	Update attribute	sunflowers	Running[The j...	Stop

5.2 Data Filtering

On the **Dashboard** tab page of the dataset, the summary of the dataset is displayed by default. In the upper right corner of the page, click **Label**. The dataset details page is displayed, showing all data in the dataset by default. On the **All**, **Unlabeled**, or **Labeled** tab page, you can add filter criteria in the filter criteria area to quickly filter the data you want to view.

The following filter criteria are supported. You can set one or more filter criteria.

- **Example Type:** Select **Hard example** or **Non-hard example**.
- **Label:** Select **All** or one or more labels you specified.
- **Sample Creation Time:** Select **Within 1 month**, **Within 1 day**, or **Custom** to customize a time range.
- **File Name** or **Path:** Filter files by file name or file storage path.
- **Labeled By:** Select the name of the user who labeled the image.

- **Sample Attribute:** Select the attribute generated by auto grouping. This filter criterion can be used only after **auto grouping** is enabled.
- **Data Attribute:** This criterion is not supported.

Figure 5-4 Filter criteria

Filter Criteria: No filter criteria selected.

Example Type: Hard Example, Non-hard example

Label: All

Sample Creation Time: Within 1 month, Within 1 day, Custom

File Name: Enter a keyword and press Enter to create a filter criterion.

Labeled By: Select an annotator.

Sample Attribute: --Select-- No attributes available. Click Auto Grouping and select Start Task to create data management attributes.

Data Attribute: Select an attribute., Select an attribute ...

5.3 Data Feature Analysis

Images or target bounding boxes are analyzed based on image features, such as blurs and brightness to draw visualized curves to help process datasets.

You can also select multiple versions of a dataset to view their curves for comparison and analysis.

Background

- Data feature analysis is only available for image datasets of the image classification and object detection types.
- Data feature analysis is only available for the published datasets. The published dataset versions in **Default** format support data feature analysis.
- A data scope for feature analysis varies depending on the dataset type.
 - In a dataset of the object detection type, if the number of labeled samples is 0, the **View Data Feature** tab page is unavailable and data features are not displayed after a version is published. After the images are labeled and the version is published, the data features of the labeled images are displayed.
 - In a dataset of the image classification type, if the number of labeled samples is 0, the **View Data Feature** tab page is unavailable and data features are not displayed after a version is published. After the images are labeled and the version is published, the data features of all images are displayed.
- The analysis result is valid only when the number of images in a dataset reaches a certain level. Generally, more than 1,000 images are required.
- Image classification supports the following data feature metrics: **Resolution**, **Aspect Ratio**, **Brightness**, **Saturation**, **Blur Score**, and **Colorfulness**. Object detection supports all data feature metrics. [Supported Data Feature Metrics](#) provides all data feature metrics supported by ModelArts.

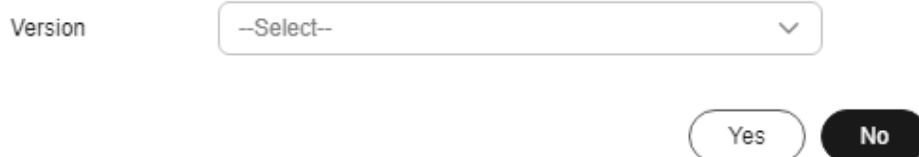
Data Feature Analysis

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. Locate the target dataset, click **More** in the **Operation** column, and select **View Data Feature**. The **View Data Feature** tab of the dataset is displayed. You can also click a dataset name to go to the dataset page and click the **View Data Feature** tab.
3. By default, feature analysis is not started for published datasets. You need to manually start feature analysis tasks for each dataset version. On the **View Data Feature** tab, click **Analyze Features**.
4. In the dialog box that is displayed, configure the dataset version for feature analysis and click **Yes** to start analysis.

Version: Select a published version of the dataset.

Figure 5-5 Starting a data feature analysis task

Analyze Features



Version

5. After a data feature analysis task is started, it takes a certain period of time to complete, depending on the data volume. If the selected version is displayed in the **Version** drop-down list and can be selected, the analysis is complete.
6. View the data feature analysis result.
Version: Select the version to be compared from the drop-down list. You can also select only one version.
Type: Select the type to be analyzed. The value can be **all**, **train**, **eval**, or **inference**.
Data Feature Metric: Select metrics to be displayed from the drop-down list. For details, see [Supported Data Feature Metrics](#).
Then, the selected version and metrics are displayed on the page. The displayed chart helps you understand data distribution for better data processing.
7. View historical records of the analysis task.
After data feature analysis is complete, you can click **Task History** on the right of the **Data Features** tab page to view historical analysis tasks and their statuses in the dialog box that is displayed.

Supported Data Feature Metrics

Table 5-1 Data feature metrics

Metric	Description	Explanation
Resolution	Image resolution. An area value is used as a statistical value.	Metric analysis results are used to check whether there is an offset point. If an offset point exists, you can resize or delete the offset point.
Aspect Ratio	An aspect ratio is a proportional relationship between an image's width and height.	The chart of the metric is in normal distribution, which is generally used to compare the difference between the training set and the dataset used in the real scenario.
Brightness	Brightness is the perception elicited by the luminance of a visual target. A larger value indicates better image brightness.	The chart of the metric is in normal distribution. You can determine whether the brightness of the entire dataset is high or low based on the distribution center. You can adjust the brightness based on your application scenario. For example, if the application scenario is night, the brightness should be lower.
Saturation	Color saturation of an image. A larger value indicates that the entire image color is easier to distinguish.	The chart of the metric is in normal distribution, which is generally used to compare the difference between the training set and the dataset used in the real scenario.
Blur Score Clarity	Image clarity, which is calculated using the Laplace operator. A larger value indicates clearer edges and higher clarity.	You can determine whether the clarity meets the requirements based on the application scenario. For example, if data is collected from HD cameras, the clarity must be higher. You can sharpen or blur the dataset and add noises to adjust the clarity.

Metric	Description	Explanation
Colorfulness	Horizontal coordinate: Colorfulness of an image. A larger value indicates richer colors. Vertical coordinate: Number of images	Colorfulness on the visual sense, which is generally used to compare the difference between the training set and the dataset used in the real scenario.
Bounding Box Number	Horizontal coordinate: Number of bounding boxes in an image Vertical coordinate: Number of images	It is difficult for a model to detect a large number of bounding boxes in an image. Therefore, more images containing many bounding boxes are required for training.
Std of Bounding Boxes Area Per Image Standard Deviation of Bounding Boxes Per Image	Horizontal coordinate: Standard deviation of bounding boxes in an image. If an image has only one bounding box, the standard deviation is 0. A larger standard deviation indicates higher bounding box size variation in an image. Vertical coordinate: Number of images	It is difficult for a model to detect a large number of bounding boxes with different sizes in an image. You can add data for training based on scenarios or delete data if such scenarios do not exist.
Aspect Ratio of Bounding Boxes	Horizontal coordinate: Aspect ratio of the target bounding boxes Vertical coordinate: Number of bounding boxes in all images	The chart of the metric is generally in Poisson distribution, which is closely related to application scenarios. This metric is mainly used to compare the differences between the training set and the validation set. For example, if the training set is a rectangle, the result will be significantly affected if the validation set is close to a square.

Metric	Description	Explanation
Area Ratio of Bounding Boxes	Horizontal coordinate: Area ratio of the target bounding boxes, that is, the ratio of the bounding box area to the entire image area. A larger value indicates a higher ratio of the object in the image. Vertical coordinate: Number of bounding boxes in all images	The metric is used to determine the distribution of anchors used in the model. If the target bounding box is large, set the anchor to a large value.
Marginalization Value of Bounding Boxes	Horizontal coordinate: Marginalization degree, that is, the ratio of the distance between the center point of the target bounding box and the center point of the image to the total distance of the image. A larger value indicates that the object is closer to the edge. (The total distance of an image is the distance from the intersection point of a ray (that starts from the center point of the image and passes through the center point of the bounding box) and the image border to the center point of the image.) Vertical coordinate: Number of bounding boxes in all images	Generally, the chart of the metric is in normal distribution. The metric is used to determine whether an object is at the edge of an image. If a part of an object is at the edge of an image, you can add a dataset or do not label the object.

Metric	Description	Explanation
Overlap Score of Bounding Boxes Overlap Score of Bounding Boxes	Horizontal coordinate: Overlap degree, that is, the part of a single bounding box overlapped by other bounding boxes. The value ranges from 0 to 1. A larger value indicates that more parts are overlapped by other bounding boxes. Vertical coordinate: Number of bounding boxes in all images	The metric is used to determine the overlapping degree of objects to be detected. Overlapped objects are difficult to detect. You can add a dataset or do not label some objects based on your needs.
Brightness of Bounding Boxes Brightness of Bounding Boxes	Horizontal coordinate: Brightness of the image in the target bounding box. A larger value indicates brighter image. Vertical coordinate: Number of bounding boxes in all images	Generally, the chart of the metric is in normal distribution. The metric is used to determine the brightness of an object to be detected. In some special scenarios, the brightness of an object is low and may not meet the requirements.
Blur Score of Bounding Boxes Clarity of Bounding Boxes	Horizontal coordinate: Clarity of the image in the target bounding box. A larger value indicates higher image clarity. Vertical coordinate: Number of bounding boxes in all images	The metric is used to determine whether the object to be detected is blurred. For example, a moving object may become blurred during collection and its data needs to be collected again.

6 Labeling Data

Model training requires a large amount of labeled data. Therefore, before training a model, label data. You can create a manual labeling job labeled by one person or by a group of persons (team labeling), or enable auto labeling to quickly label images. You can also modify existing labels, or delete them and re-label.

- Manual labeling: allows you to manually label data.
- Auto labeling: allows you to automatically label remaining data after a small amount of data is manually labeled.
- Team labeling: allows you to perform collaborative labeling for a large amount of data.

For details about data labeling, see [Introduction to Data Labeling](#).

7 Publishing Data

7.1 Introduction to Data Publishing

ModelArts distinguishes data of the same source according to versions processed or labeled at different time, which facilitates the selection of dataset versions for subsequent model building and development.

About Dataset Versions

- For a newly created dataset (before publishing), there is no dataset version information. The dataset must be published before being used for model development or training.
- The default naming rules of dataset versions are V001 and V002 in ascending order. You can customize the version number during publishing.
- You can set any version to the current version. Then the details of the version are displayed on the dataset details page.
- You can obtain the dataset in the manifest file format corresponding to each dataset version based on the value of **Storage Path**. The dataset can be used when you import data or filter hard examples.
- The version of a table dataset cannot be changed.

7.2 Publishing a Data Version

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. Locate the row containing the target dataset and click **Publish** in the **Operation** column. Alternatively, click the dataset name to go to the **Dashboard** tab page of the dataset, and click **Publish** in the upper right corner.
3. In the displayed dialog box, set the parameters and click **OK**.

Table 7-1 Parameters for publishing a dataset

Parameter	Description
Version	The naming rules of V001 and V002 in ascending order are used by default. A version name can be customized. Only letters, digits, hyphens (-), and underscores (_) are allowed.
Format	Only table datasets support version format setting. Available values are CSV and CarbonData . NOTE If the exported CSV file contains any command starting with =, +, -, or @, ModelArts automatically adds the Tab setting and escapes the double quotation marks (") for security purposes.
Splitting	Only image classification, object detection, text classification, and sound classification datasets support data splitting. By default, this function is disabled. After this function is enabled, set the training and validation ratios. Enter a value ranging from 0 to 1 for Training Set Ratio . After the training set ratio is set, the validation set ratio is determined. The sum of the training set ratio and the validation set ratio is 1. NOTE To ensure the model accuracy, you are advised to set the training set ratio to 0.8 or 0.9. The training set ratio is the ratio of sample data used for model training. The validation set ratio is the ratio of the sample data used for model validation. The training and validation ratios affect the performance of training templates.
Description	Description of the current dataset version.
Hard Example	Only image classification and object detection datasets support hard example attributes. By default, this function is disabled. After this function is enabled, information such as the hard example attributes of the dataset are written to the corresponding manifest file.

Directory Structure of Dataset Versions

Datasets are managed based on OBS directories. After a new version is published, the directory is generated based on the new version in the output dataset path.

Take an image classification dataset as an example. After the dataset is published, the directory structure of related files generated in OBS is as follows:

```
|-- user-specified-output-path
    |-- DatasetName-datasetId
        |-- annotation
            |-- VersionName1
                |-- VersionName1.manifest
```

```
|-- VersionMame2  
...  
|-- ...
```

The following uses object detection as an example. If a manifest file is imported to the dataset, the following provides the directory structure of related files after the dataset is published:

```
|-- user-specified-output-path  
  |-- DatasetName-datasetId  
    |-- annotation  
      |-- VersionMame1  
        |-- VersionMame1.manifest  
        |-- annotation  
          |-- file1.xml  
      |-- VersionMame2  
        ...  
    |-- ...
```

Take video labeling as an example. After the dataset is published, the labeling result file (XML) is stored in the dataset output directory.

```
|-- user-specified-output-path  
  |-- DatasetName-datasetId  
    |-- annotation  
      |-- VersionMame1  
        |-- VersionMame1.manifest  
        |-- annotations  
          |-- images  
            |-- videoName1  
              |-- videoName1.timestamp.xml  
            |-- videoName2  
              |-- videoName2.timestamp.xml  
      |-- VersionMame2  
        ...  
    |-- ...
```

The key frames for video labeling are stored in the dataset input directory.

```
|-- user-specified-input-path  
  |-- images  
    |-- videoName1  
      |-- videoName1.timestamp.jpg  
    |-- videoName2  
      |-- videoName2.timestamp.jpg
```

7.3 Managing Data Versions

During data preparation, you can publish data into multiple versions for dataset management. You can view version updates, set the current version, and delete versions.

Viewing Dataset Version Updates

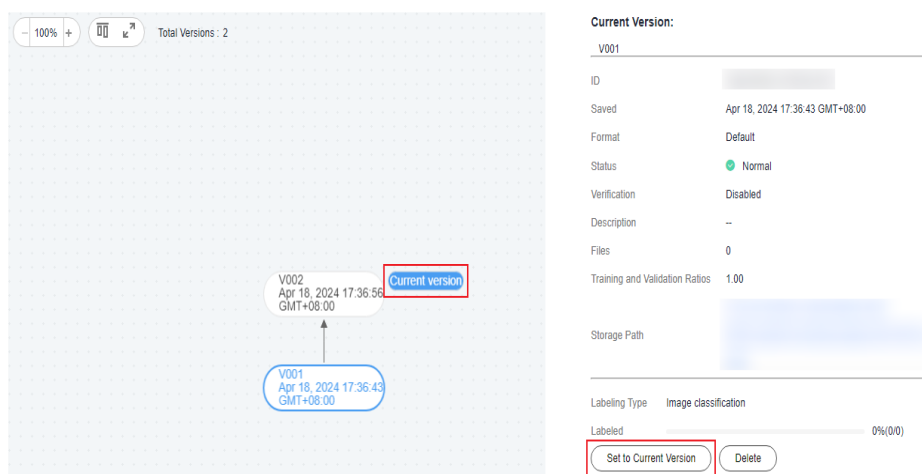
1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. In the dataset list, choose **More > Manage Version** in the **Operation** column. The **Manage Version** tab page is displayed.

You can view basic information about the dataset, and view the versions and publish time on the left.

Setting to Current Version

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. In the dataset list, choose **More > Manage Version** in the **Operation** column. The **Manage Version** tab page is displayed.
3. On the **Manage Version** tab page, select the desired dataset version, and click **Set to Current Version** in the basic information area on the right. After the setting is complete, **Current version** is displayed to the right of the version name.

Figure 7-1 Setting to current version



NOTE

Only the version in **Normal** status can be set to the current version.

Deleting a Dataset Version

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. In the dataset list, choose **More > Manage Version** in the **Operation** column. The **Manage Version** tab page is displayed.
3. Locate the row that contains the target version, and click **Delete** in the **Operation** column. In the dialog box that is displayed, click **OK**.

NOTE

Deleting a dataset version does not remove the original data. Data and its labeling information are still stored in the OBS directory. However, this affects version management. Exercise caution when performing this operation.

8 Exporting Data

8.1 Introduction to Exporting Data

You can select data or filter data based on the filter criteria in a dataset and export to a new dataset or the specified OBS path. The historical export records can be viewed in task history.

Only datasets of image classification, object detection, and image segmentation types can be exported.

- For image classification datasets, only the label files in TXT format can be exported.
- For object detection datasets, only XML label files in Pascal VOC format can be exported.
- For image segmentation datasets, only XML label files in Pascal VOC format and mask images can be exported.

8.2 Exporting Data to a New Dataset

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. In the dataset list, select an image dataset and click the dataset name to go to the **Dashboard** tab page of the dataset.
3. Click **Export** in the upper right corner. In the displayed **Export To** dialog box, enter the related information and click **OK**.

Data Source: Select **New Dataset**.

Name: name of the new dataset

Storage Path: input path of the new dataset, that is, the OBS path where the data to be exported is stored

Output Path: output path of the new dataset, that is, the output path after labeling is complete. The output path cannot be the same as the storage path, and the output path cannot be a subdirectory of the storage path.

Figure 8-1 Exporting to a new dataset

Export To

The dialog box titled "Export To" has a "Data Source" section with two buttons: "New Dataset" (highlighted in blue) and "OBS" (greyed out). Below this are three input fields: "Name" with the placeholder "Enter a dataset name.", "Storage Path" with a question mark icon and the placeholder "Select an OBS path.", and "Output Path" with a question mark icon and the placeholder "Select an OBS path.". At the bottom right are two buttons: "OK" (black) and "Cancel" (white with black border).

4. After the data is exported, view it in the specified path. After the data is exported, you can view the new dataset in the dataset list.
5. On the **Dashboard** tab page, click **Export History** in the upper right corner. In the displayed dialog box, view the task history of the dataset.

8.3 Exporting Data to OBS

1. Log in to the [ModelArts management console](#). In the navigation pane, choose **Data Management > Datasets**.
2. In the dataset list, select an image dataset and click the dataset name to go to the **Dashboard** tab page of the dataset.
3. Click **Export** in the upper right corner. In the displayed **Export To** dialog box, enter the related information and click **OK**.

Data Source: Select **OBS**.

Storage Path: path where the data to be exported is stored. You are advised not to save data to the input or output path of the current dataset.

Figure 8-2 Exporting data to OBS

Export To

The dialog box titled "Export To" has a "Data Source" section with two buttons: "New Dataset" (greyed out) and "OBS" (highlighted in blue). Below this is one input field: "Storage Path" with a question mark icon and the placeholder "Select an OBS path.". At the bottom right are two buttons: "OK" (black) and "Cancel" (white with black border).

4. After the data is exported, view it in the specified path.
5. On the **Dashboard** tab page, click **Export History** in the upper right corner. In the displayed dialog box, view the task history of the dataset.

Figure 8-3 Viewing the task history

View Task History

Task ID	Created	Type	Path	Total	Status
TwBaNfmTN7z8zbyo4b	Apr 18, 2024 17:27:50 GMT+08:00	OBS	/modelarts-xj/test/	249	Successful