Batch Compute

Best Practices

Product Documentation

Contents

Best Practices
Building Cluster Using Compute Environment

Last updated：2024-01-13 11:19:29

Scenario

With the capabilities of BatchCompute (Batch), you can easily and efficiently maintain the Cloud Virtual Machine (CVM) cluster. The Batch computing environment corresponds to common cluster concepts. This document describes how to use the capabilities of the Batch computing environment to create or terminate an ultra cost-effective resource cluster.

Prerequisites

You can get prepared as instructed by Preparation.

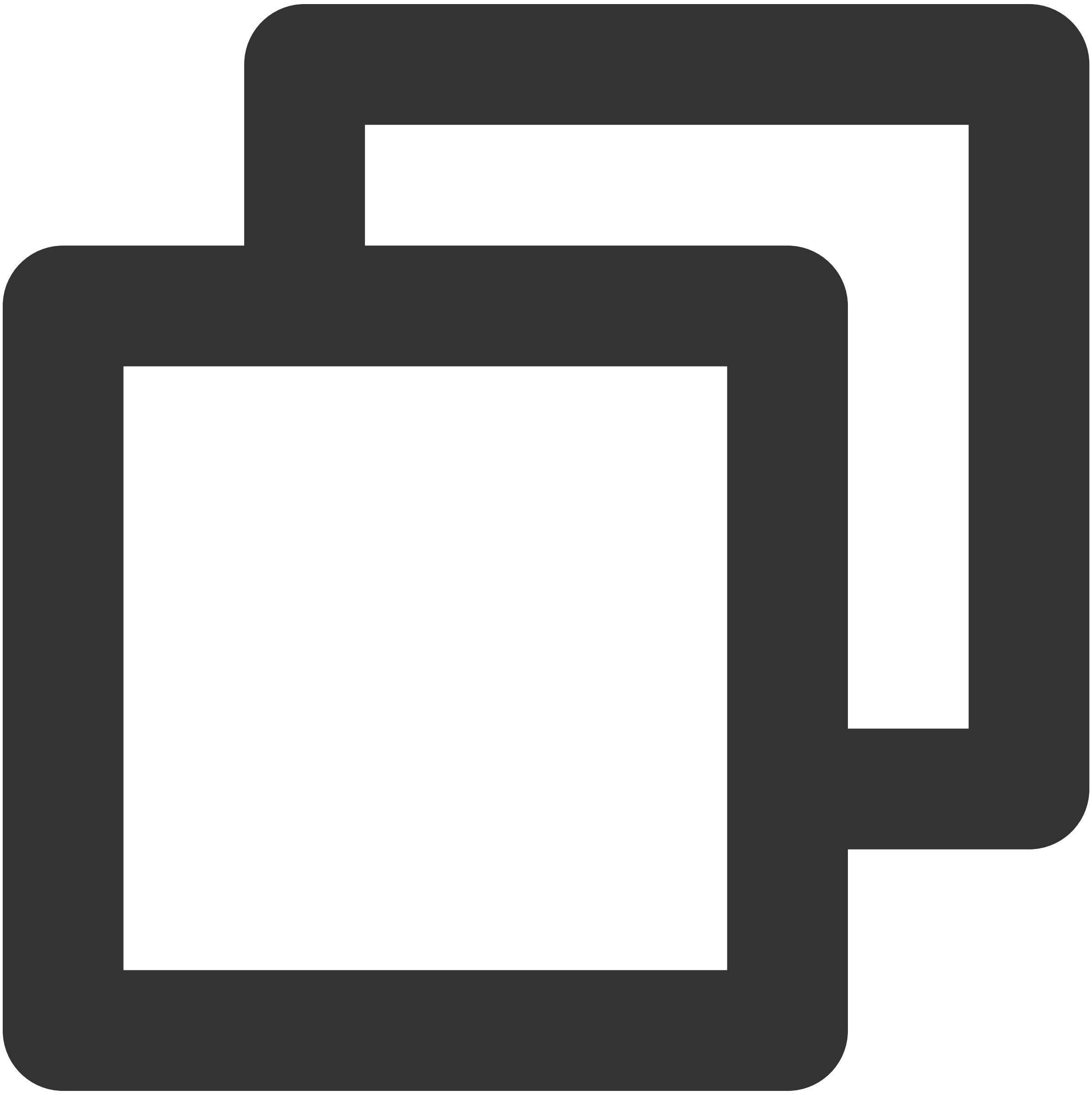
Steps

Installing and Configuring TCCLI

Note：
In the current computing environment, you can only call command lines. Install TCCLI by referring to the following steps.
1. Install TCCLI by referring to Preparation.
2. Run the following command to verify whether TCCLI is successfully installed:

```
tccli batch help
```

The returned result is as follows, indicating that TCCLI is successfully installed:



```
NAME
     batch
DESCRIPTION
     batch-2017-03-12
USEAGE
     tccli batch <action> [--param...]
OPTIONS
     help
     show the tccli batch help info
     --version
     specify a batch api version
```

```
AVAILABLE ACTION
     DescribeComputeEnv
     Used to query details of the computing environment
     CreateTaskTemplate
     Used to create a task template
```

3. Configure TCCLI by referring to Preparation.

Creating a Computing Environment

You can acquire and modify the official example to create a Batch computing environment under a personal account. Learn each configuration item in the computing environment by referring to the following information.
You can also refer to the APIs related to the computing environment, for example, CreateComputeEnv.
The following example shows how to create a cluster with ten BS1.LARGE8 instances (Standard BatchCompute model, 4-core CPU and 8 GB memory) in Guangzhou Zone 2:

```
tccli batch CreateComputeEnv --version 2017-03-12 --ComputeEnv '{
    "EnvName": "batch-env",           // Computing environment name
    "EnvDescription": "batch env demo",    // Computing environment description
    "EnvType": "MANAGED",                  // Computing environment type: MANAGED
    "EnvData": {                           // Specific configuration (Refer to the
        "InstanceType": "BS1.LARGE8",      // CVM instance type in a computing env
        "ImageId": "img-m4q71qnf",         // CVM image ID in a computing environm
        "LoginSettings": {
            "Password": "B1[habcd"         // CVM login password in a computing en
        },
        "InternetAccessible": {
```



```
            "PublicIpAssigned": "TRUE",      // Whether the CVM requires a public IP
            "InternetMaxBandwidthOut": 10   // CVM bandwidth cap in a computing env
        },
        "SystemDisk": {
            "DiskType": "CLOUD_BASIC",       // Type of a CVM disk in a computing en
            "DiskSize": 50                   // Size of a CVM disk in a computing en
        }
    },
    "DesiredComputeNodeCount": 10            // Number of desired compute nodes
}'
--Placement'{
    "Zone": "ap-guangzhou-2"                 // Availability zone (Guangzhou Zone 2
}'
```

Sample Request


```
tccli batch CreateComputeEnv --version 2017-03-12 --ComputeEnv '{"EnvName":"batch-e
```

Response Example

In the following return values, `EnvId` indicates the unique ID of a Batch computing environment.

The following will describe how to use Batch command line interface (CLI) to check the computing environment and instance information within it, and `EnvId` will be used. You need to record the returned value of `EnvId` .


```
{
    "EnvId": "env-jlatqfkn",
    "RequestId": "297ed003-7373-4950-9721-242d3d40b3ca"
}
```

You can view the created CVM in CVM Console or view and manage the CVM by using the CreateComputeEnv API.

Viewing the List of Computing Environments

You can use the Batch CLI to view the list of all created computing environments.

Sample Request

Run the following command to view the list of computing environments:

```
tccli batch DescribeComputeEnvs --version 2017-03-12
```

Response Example

The following result contains information about the computing environment to be queried (some information is omitted):

```
{
    "TotalCount": 1,
    "ComputeEnvSet": [
        {
            "EnvId": "env-jlatqfkn",
            "ComputeNodeMetrics": {
                ...
            },
            "EnvType": "MANAGED",
            "DesiredComputeNodeCount": 2,
            "EnvName": "test compute env",
```



```
            "Placement": {
                ...
            },
            "CreateTime": "2019-10-08T08:55:12Z"
        }
    ],
    "RequestId": "7a1f9338-0118-46bf-b59f-60ace9f154f5"
}
```


Viewing the Specified Computing Environment and Its Node List

Sample Request

Run the following command to view the specified computing environment and its node list:

```
tccli batch DescribeComputeEnv --version 2017-03-12 --EnvId env-jlatqfkn
```

Response Example

The following result contains the overall computing environment and details of each node (some information is omitted):

```
{
    "EnvId": "env-jlatqfkn",
    "ComputeNodeMetrics": {
        ...
    },
    "EnvType": "MANAGED",
    "DesiredComputeNodeCount": 2,
    "ComputeNodeSet": [
        ...
    ],
    "RequestId": "407de39c-1c3d-489e-9a35-5257ae561e87",
```

```
    "Placement": {
        ...
    },
    "EnvName": "test compute env",
    "CreateTime": "2019-10-08T08:55:12Z"
}
```

Terminating a Computing Environment

Sample Request

Run the following command to terminate the computing environment. After you run the command, the computing environment automatically terminates all CVMs in the cluster.

```
tccli batch DeleteComputeEnv --version 2017-03-12 --EnvId env-jlatqfkn
```

Response Example

```
{
    "RequestId": "029becda-2a4e-4989-aa77-6fbb5a873555"
}
```


Example: 3ds Max 2018 Rendering

Last updated：2024-01-13 11:19:29

Getting Started

This document describes how to use BatchCompute console to submit a job, complete 3ds Max 2018 image rendering, and export the rendered image. The operation steps are presented below:

Step 1. Creating a custom image

1. For more information about how to create a custom image, see Windows Custom Images.
2. For more information about how to install 3ds Max 2018, visit the Autodesk official website.

Note：

Disable the Windows Firewall temporarily to avoid blocking software downloads.

Select a proper graphics card model to prevent graphics card initialization failures. Under normal circumstances, "Nitrous Software" is recommended. For more information, see Display Driver Selection Dialog.

Step 2. Preparing rendering files

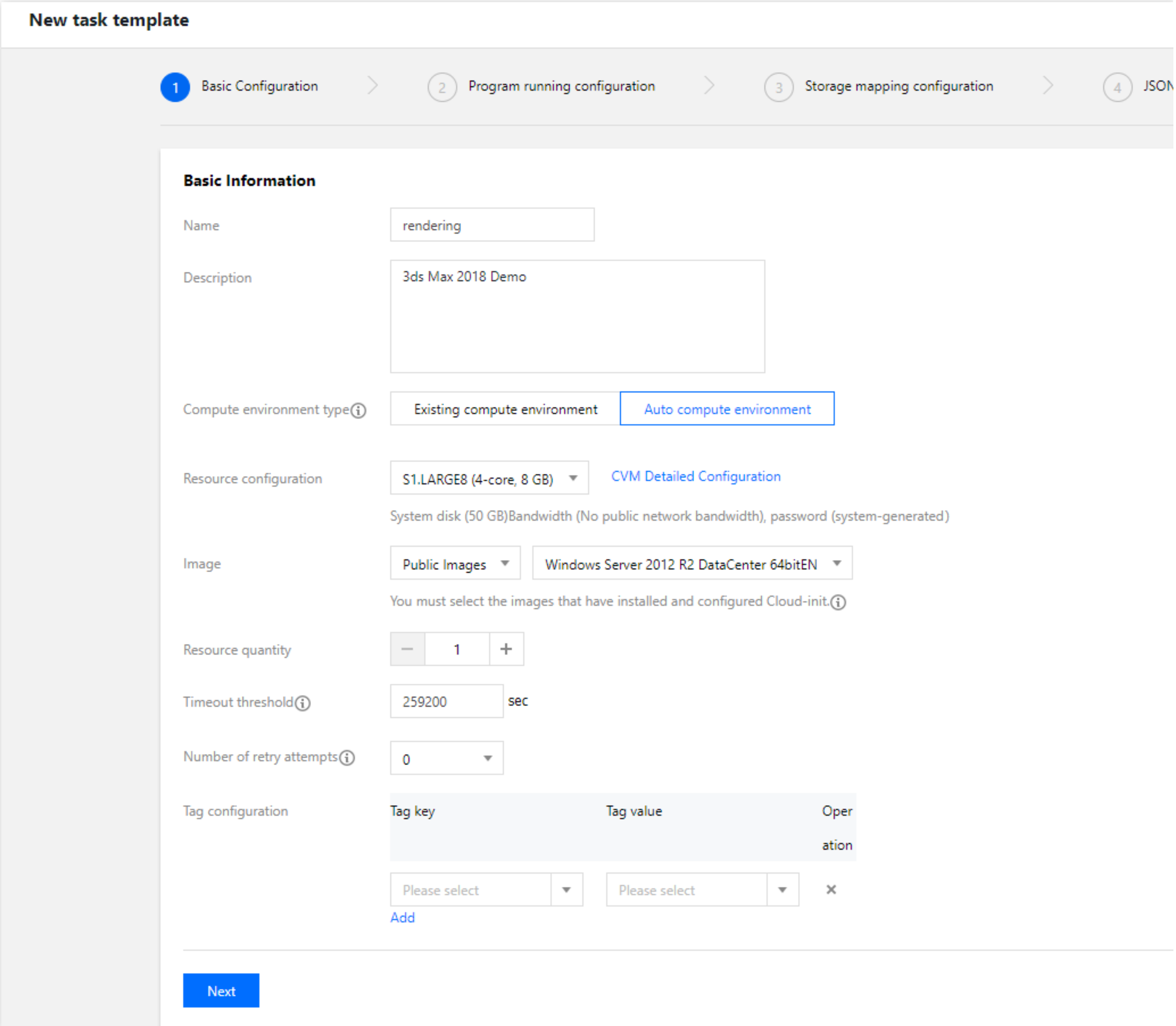
There are two main methods for storing rendering materials, namely Cloud Object Storage (COS) and Cloud File Storage (CFS). By configuring the mount parameters, BatchCompute locally mounts a COS bucket or a CFS instance before the rendering job runs. This way, the renderer can access the COS bucket or CFS instance in the way that it accesses local files.

If the rendering materials are small-sized, we recommend that you compress them into a .gzip package and upload the package to COS. For more information, see Uploading an Object.

While the rendering materials are large-sized, upload them to the CFS.

Step 3. Creating a task template

1. Log in to the BatchCompute console. In the left sidebar, click **Task Template**.
2. Select a target region at the top of the **Task Template** page.
3. Click **Create**. On the **New task template** page, create a template, as shown below:



Name: Enter a custom name, such as **rendering**.

Description: Enter a custom name, such as **3ds Max 2018 Demo**.

Compute environment type: Select a compute environment as needed. **Auto compute environment** is selected in this example.

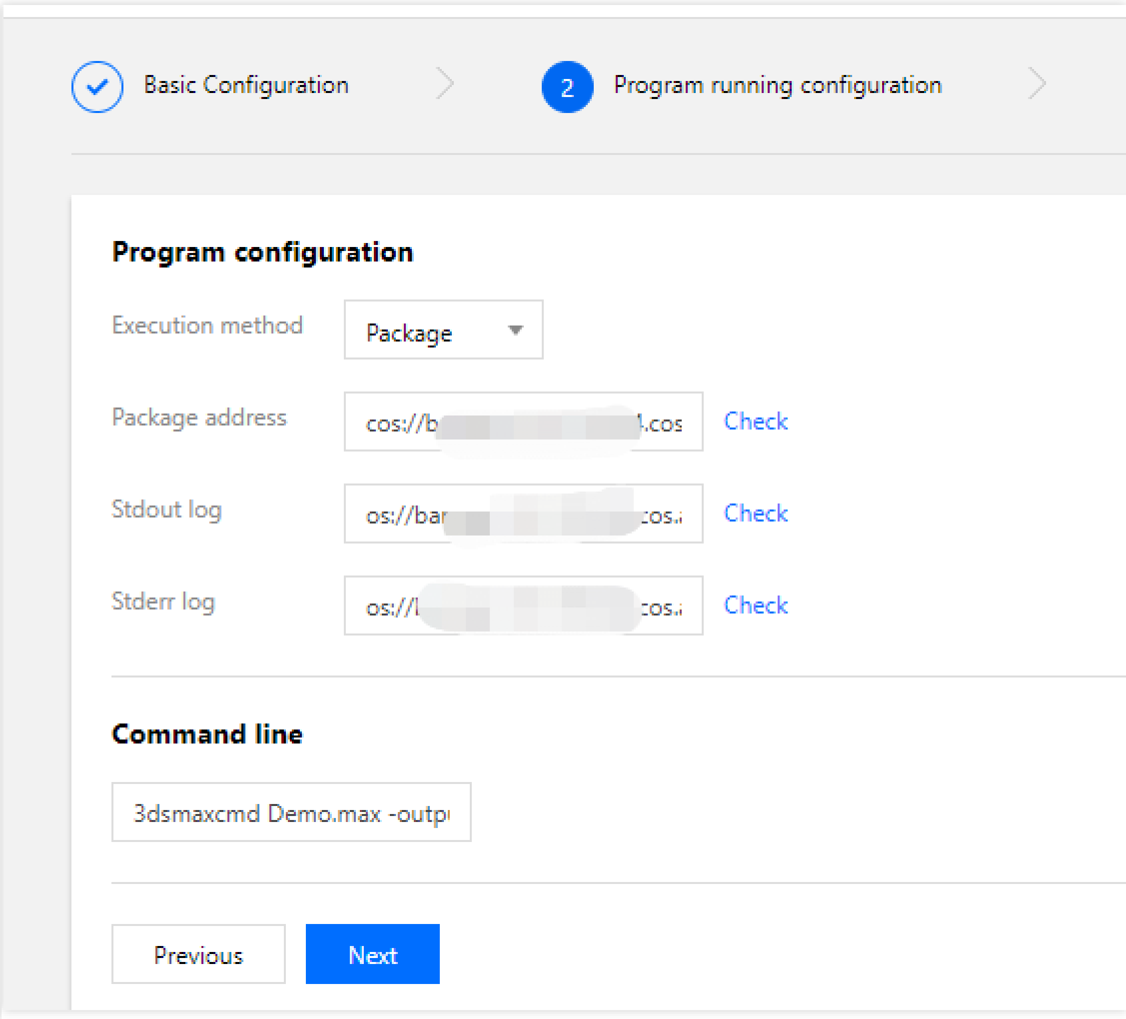
Resource configuration: Select **S1.LARGE8 (4-core, 8 GB)**.

Image: Enter a custom image identifier.

Resource quantity: Enter the number of concurrent rendering instances. Example: 1.

Timeout threshold and **Number of retry attempts**: Keep the default values.

4. Click **Next**. Configure application information, as shown below:



Execution method: Select **Package**.

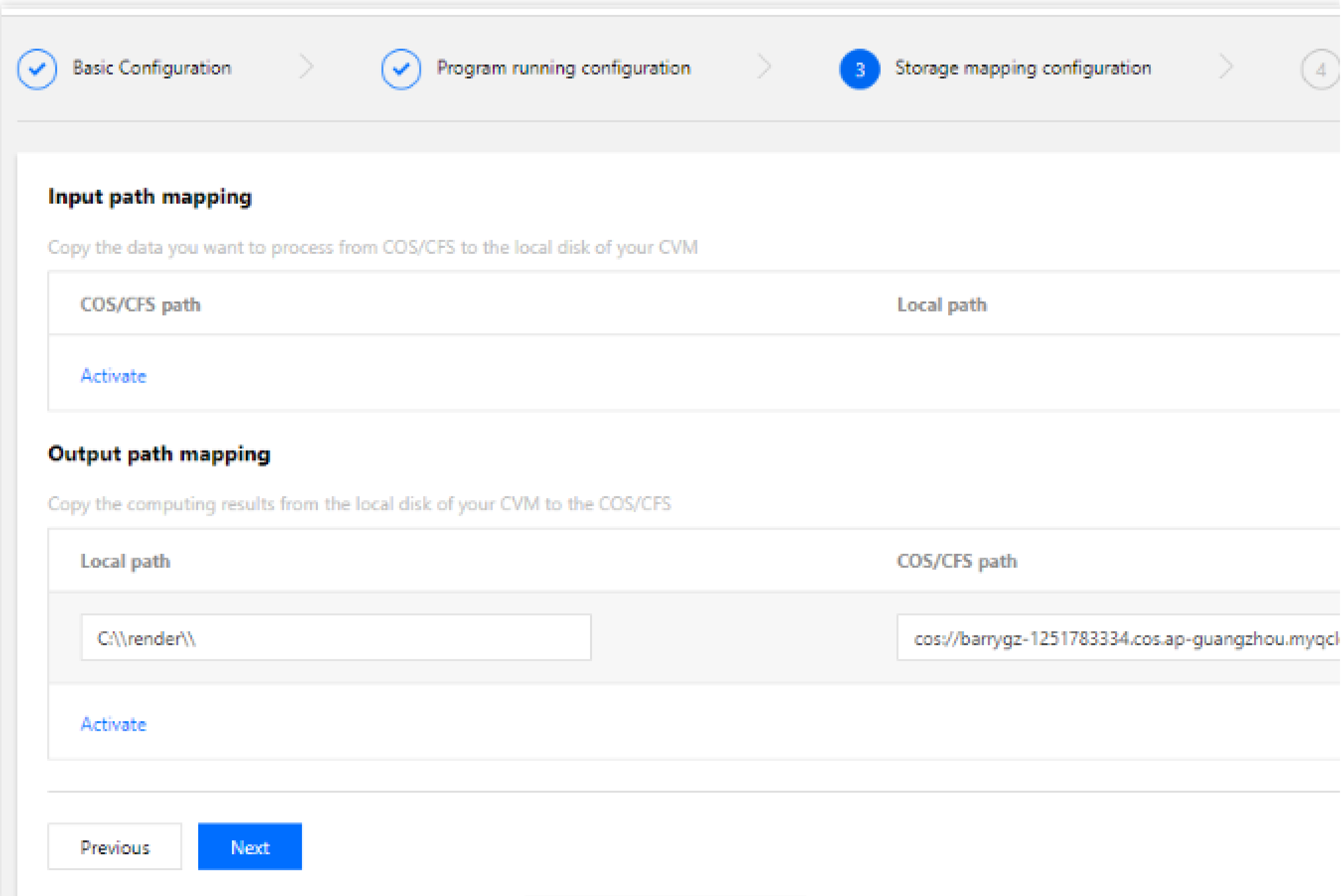
Package address: Example: `cos://barrygz-125xxxxxx4.cos.ap-guangzhou.myqcloud.com/render/max.tar.gz` .

Stdout log: For more information about the format, see Entering COS & CFS Paths.

Stderr log: The same as above.

Command line: Enter `3dsmaxcmd Demo.max -outputName:c:\\\\render\\\\image.jpg` .

5. Click **Next**. Configure the storage mapping, as shown below:



Local path under **Output path mapping**: Enter `C:\\\\render\\\\` .

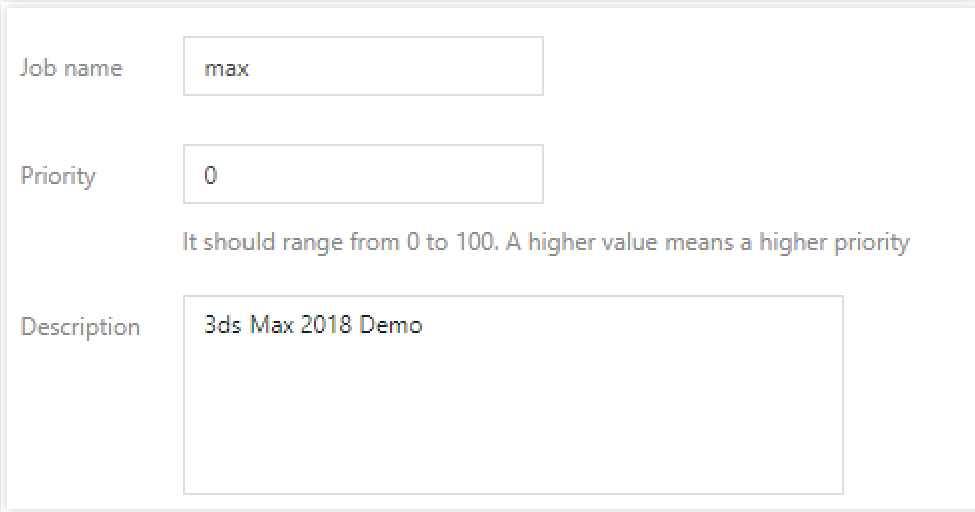
COS/CFS path under **Output path mapping**: For more information about the format, see Entering COS & CFS Paths.

6. Click **Next**. Preview the JSON file of the task.

7. Confirm that the configuration is correct and click **Save**.

Step 4. Submitting a job

1. In the left sidebar, click **Jobs** to go to the **Jobs** page.
2. Select a target region at the top of the **Jobs** page and click **Create**.
3. Configure the fundamental job information in the “New Job” window, as shown below:

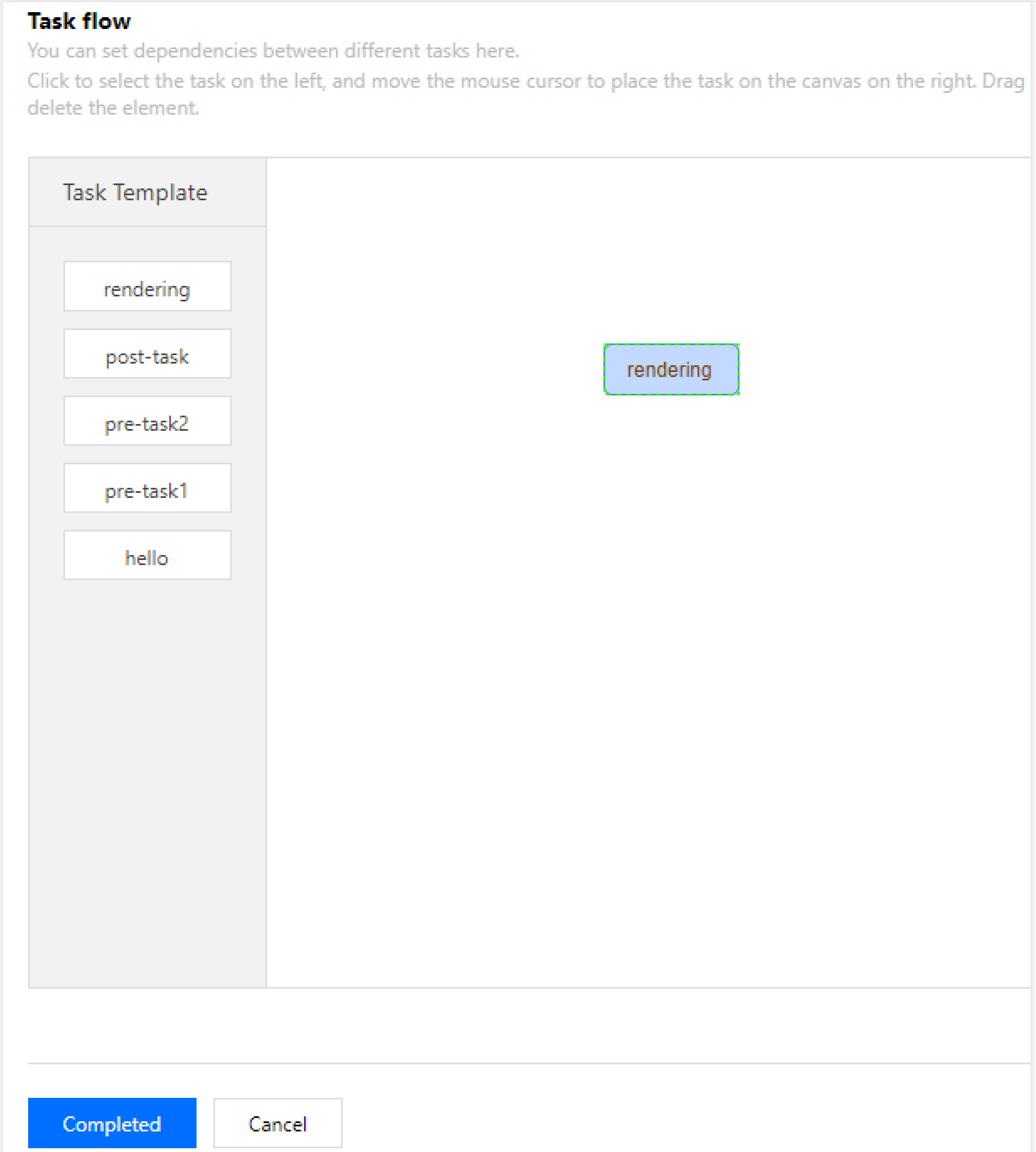


Job name: Enter **max**.

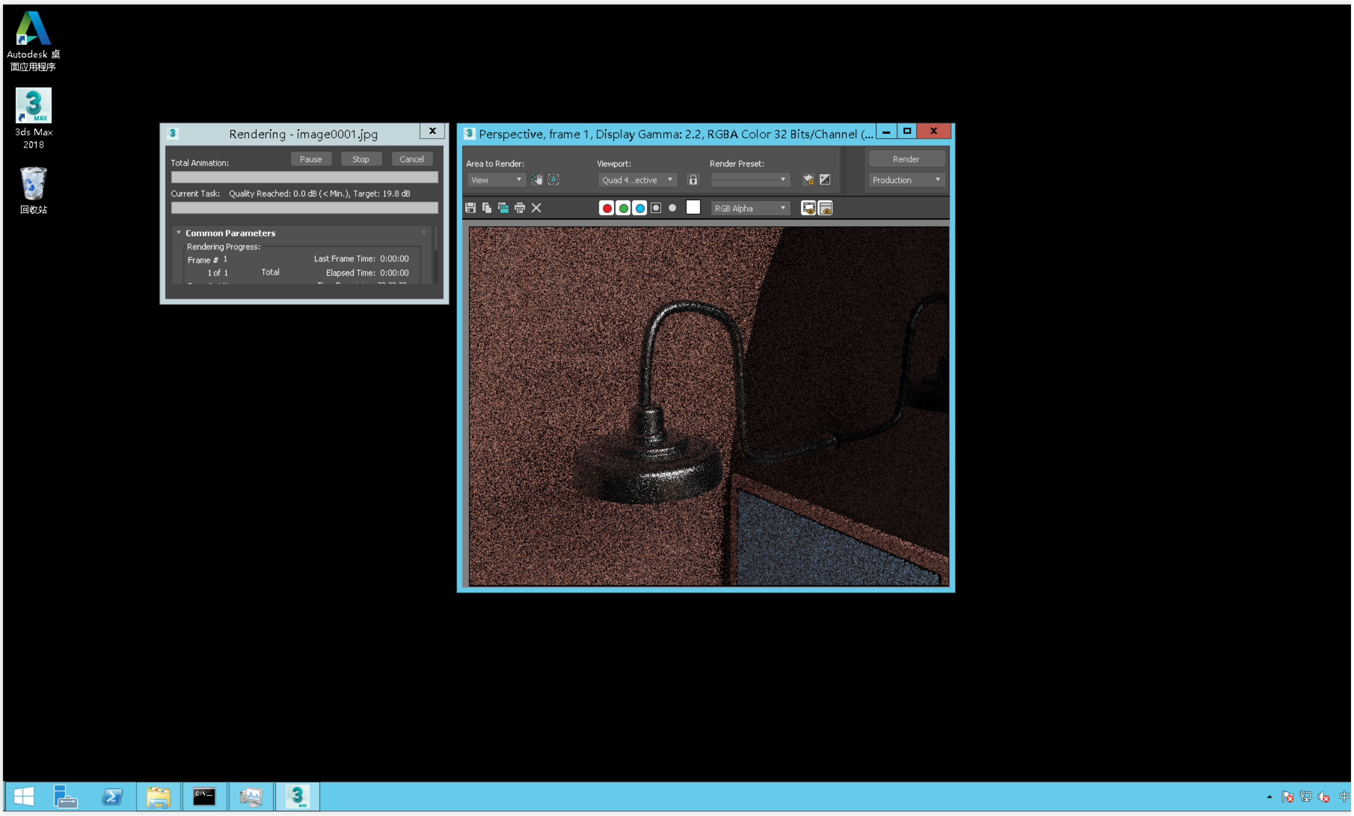
Priority: Keep the default value.

Description: Enter **3ds Max 2018 Demo**.

4. On the left side of the **Task flow** page, find the **rendering** task and drag it to the canvas on the right.



5. Confirm the configurations in the task details area on the right side of the **Task flow** page and click **Completed**.
6. For more information about how to query job running information, see Information Query.
7. Rendering Process Demo.



8. For more information about how to query rendering results, see Viewing Object Information.

Subsequent Operations

This document illustrates a simple rendering job to demonstrate fundamental BatchCompute capabilities. You can continue to test the advanced capabilities of BatchCompute as instructed in the Console User Guide.

Various CVM configurations: BatchCompute provides a variety of CVM configuration options. You can customize your own CVM configuration based on your business scenario.

Remote storage mapping: BatchCompute optimizes storage access and simplifies access to remote storage services into operations in the local file system.

Parallel rendering of multiple images: With BatchCompute, you can specify the number of concurrent rendering instances and use environment variables to differentiate rendering instances. Each instance reads different rendering materials to achieve parallel rendering.

Example: Deep Learning

Last updated：2024-01-13 11:19:29

Getting Started

This document describes how to write a multilayer perceptron (MLP) BP algorithm based on a Scikit-learn machine learning library to predict the probability of winning and losing between two football teams by modeling historical international football matches, team rankings, physical and skill metrics of players, and the FIFA 2018 group match results. Below are the detailed directions.

Step 1. Creating a custom image

1. Create a custom image. For more information, see Creating a Custom Image.
2. Install the dependency package. Take CentOS 7.2 64-bit as an example.

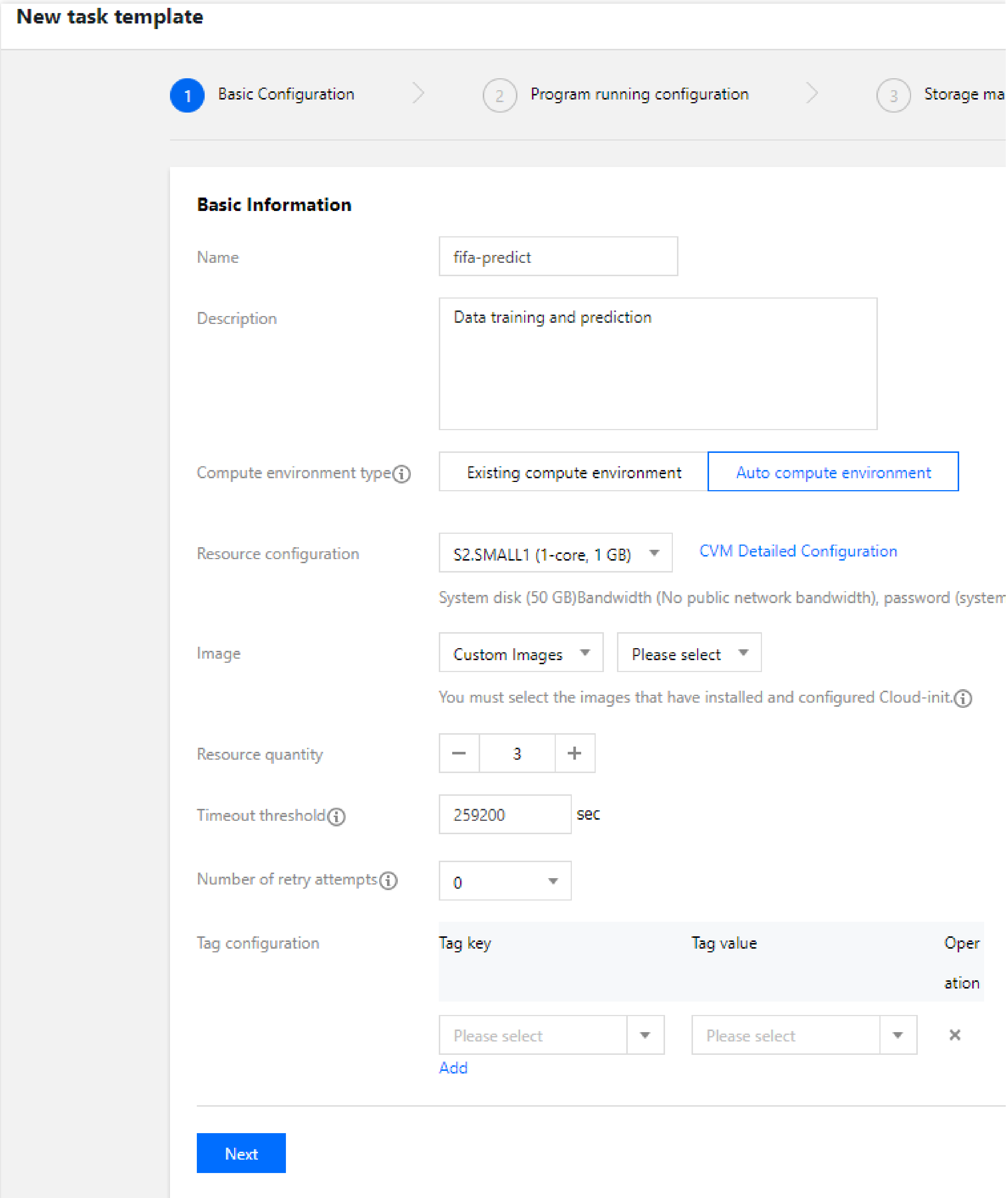
```
yum -y install gcc
yum -y install python-devel
yum -y install tkinter
yum -y install python-pip
pip install --upgrade pip
pip install pandas
pip install numpy
pip install matplotlib
pip install seaborn
pip install sklearn
pip install --upgrade python-dateutil
```


Step 2. Downloading the application package

Click here to download the application package, and upload it to COS. After you specify the COS endpoint of the package, BatchCompute downloads the package to a CVM instance before a job starts, and automatically decompresses and executes the package.

Step 3. Creating a task template named fifa-predict

1. Log in to the BatchCompute console. In the left sidebar, click **Task Template**.
2. Select a target region at the top of the **Task Template** page.
3. Click **Create**. On the **New task template** page, create a template, as shown below:



Name: Enter **fifa-predict**.

Description: Enter **Data training and prediction**.

Compute environment type: Select a compute environment as needed. **Auto compute environment** is selected in this example.

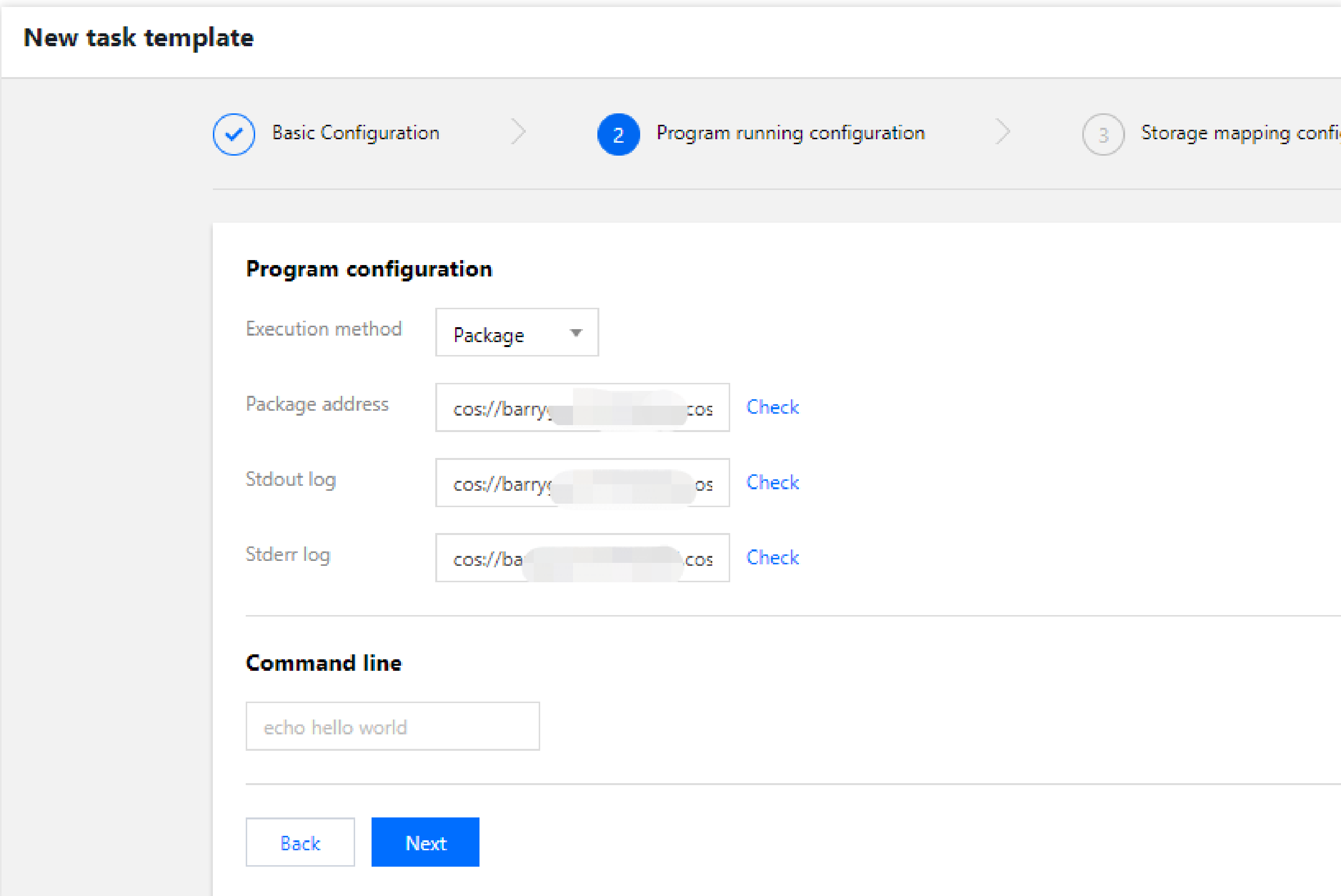
Resource configuration: Select **S2.SMALL1 (1-core, 1 GB)**. Public network bandwidth is charged on a pay-as-you-go basis.

Image:Select the custom image identifier from the image created in Step 1. The output content is in markdown format.

Resource quantity: Enter the number of concurrent rendering instances. Example: 3, which means to train 3 neural network models concurrently.

Timeout threshold and **Number of retry attempts**: Keep the default values.

4. Click **Next**. Configure application information, as shown below:



Execution method: Select **Package**.

Package address: Example: `cos://barrygz-1251783334.cosgz.myqcloud.com/fifa/fifa.2018.tar.gz`.

Stdout log: For more information about the format, see Entering COS & CFS Paths.

Stderr log: The same as above.

Command line: Enter `python predict.py "Japan" "Senegal"`.

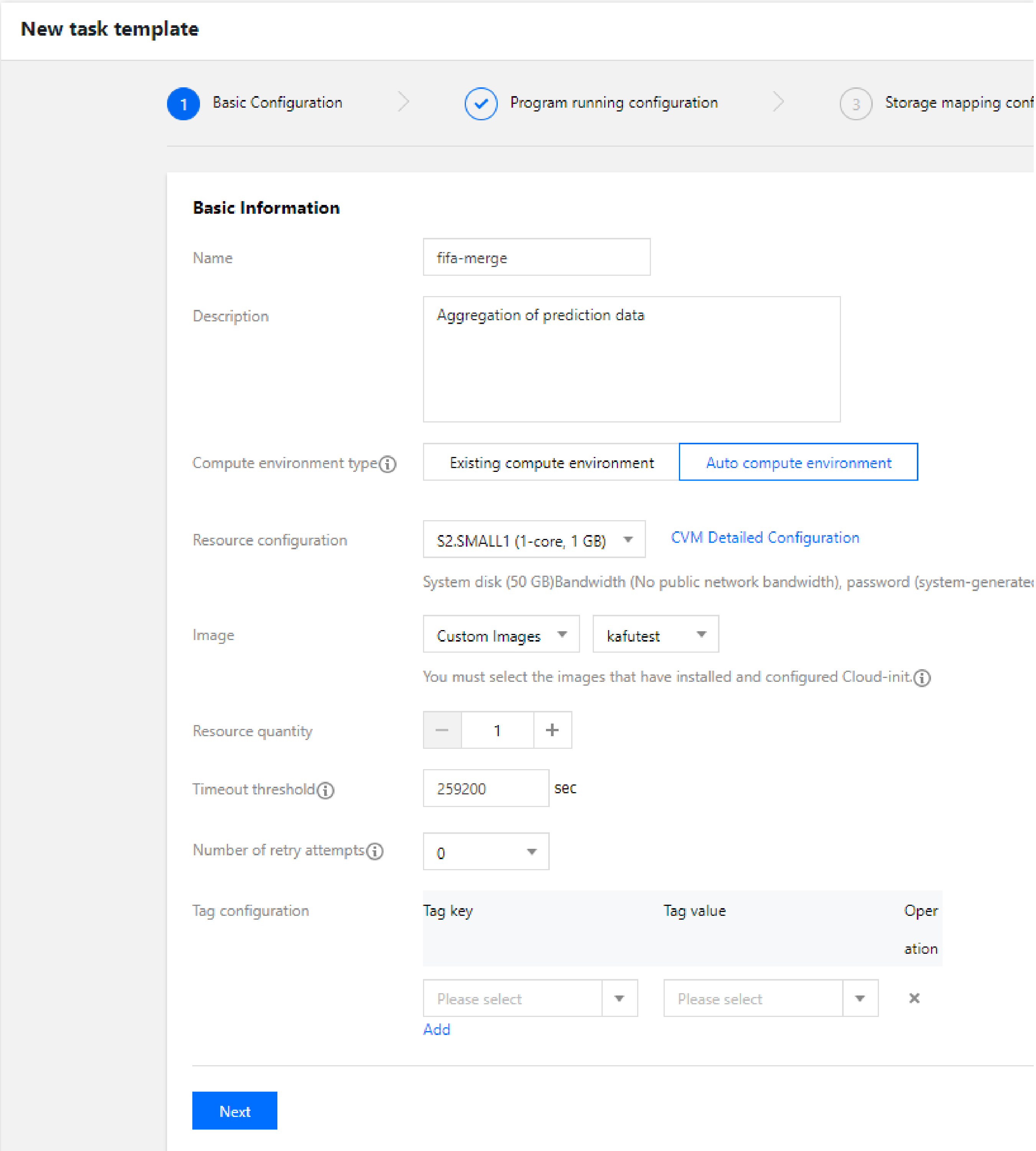
Team list: 'Saudi Arabia', 'Egypt', 'Uruguay', 'Portugal', 'Spain', 'Morocco', 'Iran', 'France', 'Australia', 'Peru', 'Denmark', 'Argentina', 'Iceland', 'Croatia', 'Nigeria', 'Brazil', 'Switzerland', 'Costa Rica', 'Serbia', 'Germany', 'Mexico', 'Sweden', 'Korea Republic', 'Belgium', 'Panama', 'Tunisia', 'England', 'Poland', 'Senegal', 'Colombia', 'Japan'.

5. Skip the storage mapping configuration step and click **Next**.

6. Preview the JSON file of the task, and click **Save** after confirmation.

Step 4. Creating a task template named fifa-merge

1. Log in to the BatchCompute console. In the left sidebar, click **Task Template**.
2. Select a target region at the top of the **Task Template** page.
3. Click **Create**. On the "New task template" page, create a template, as shown below:



Name: Enter **fifa-merge**.

Description: Enter **Aggregation of prediction data**.

Compute environment type: Select a compute environment as needed. **Auto compute environment** is selected in this example.

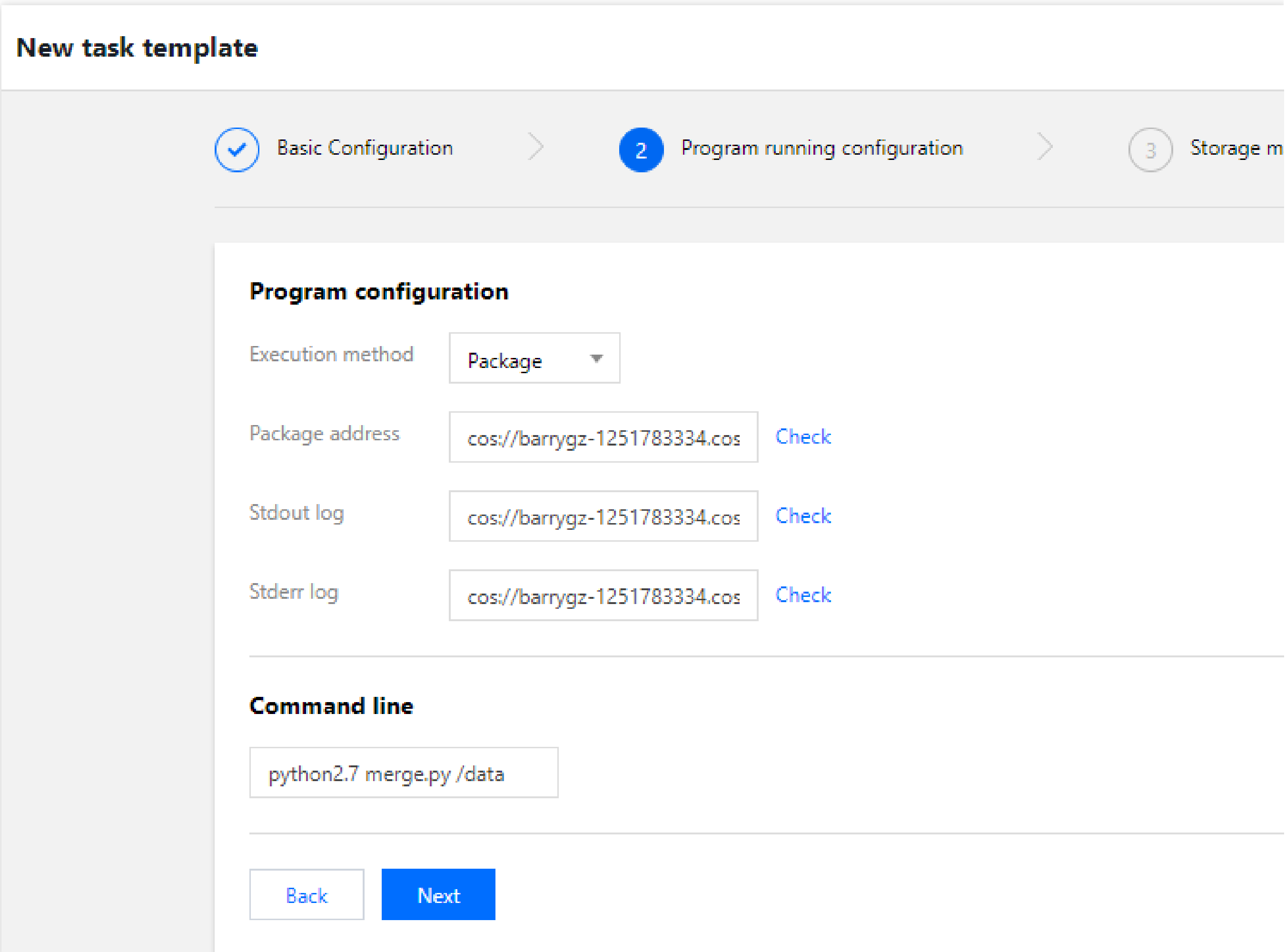
Resource configuration: Select **S2.SMALL1 (1-core, 1 GB)**. Public network bandwidth is charged on a pay-as-you-go basis.

Image: Enter a custom image identifier. Use the image created in Step 1.

Resource quantity: 1.

Timeout threshold and **Number of retry attempts**: Keep the default values.

4. Click **Next**. Configure application information, as shown below:



Execution method: Select **Package**.

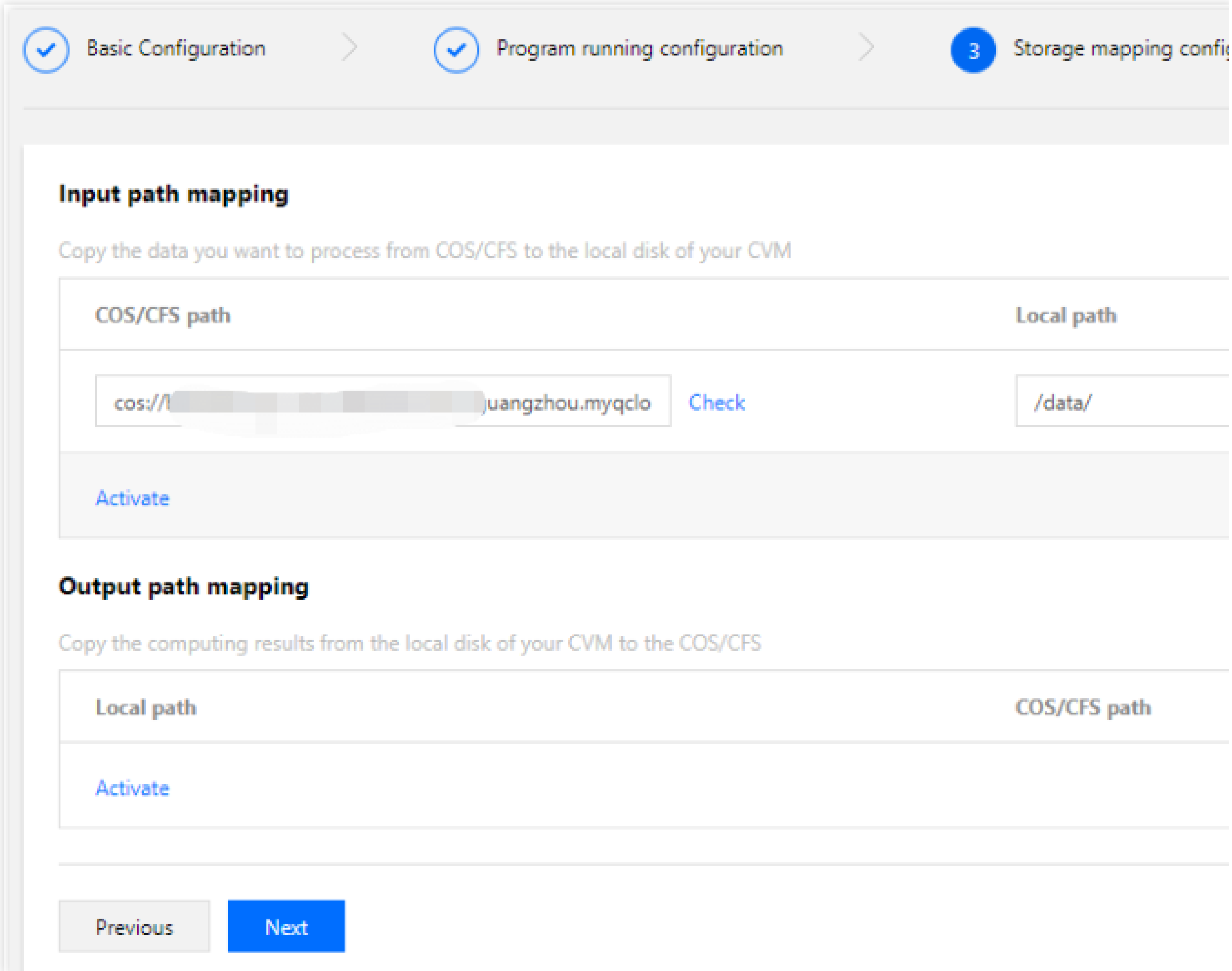
Package address: Example: `cos://barrygz-1251783334.cosgz.myqcloud.com/fifa/fifa.2018.tar.gz` .

Stdout log: For more information about the format, see Entering COS & CFS Paths.

Stderr log: The same as above.

Command line: Enter `python merge.py /data` .

5. Click **Next**. Configure the storage mapping, as shown below:



COS/CFS path under **Input path mapping**: Enter the Stdout log path of the **fifa-predict** template.

Local path under **Input path mapping**: Enter `/data` .

6. Preview the JSON file of the task, and click **Save** after confirmation.

Step 5. Submitting a job

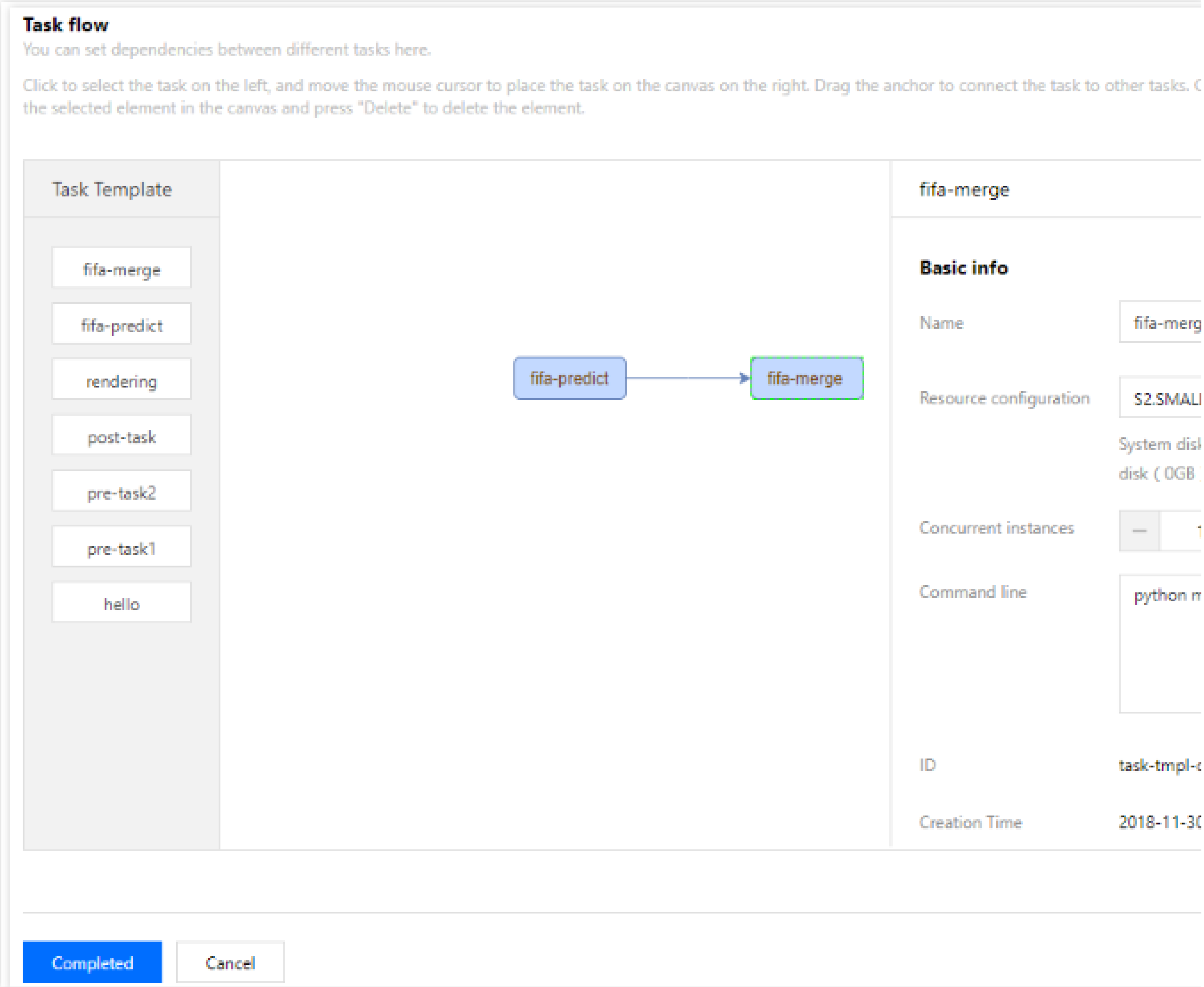
1. In the left sidebar, click **Jobs** to go to the **Jobs** page.
2. Select a target region at the top of the **Jobs** page and click **Create**.
3. On the **New job** page, configure job information, as shown below:

Job name: Enter **fifa**.

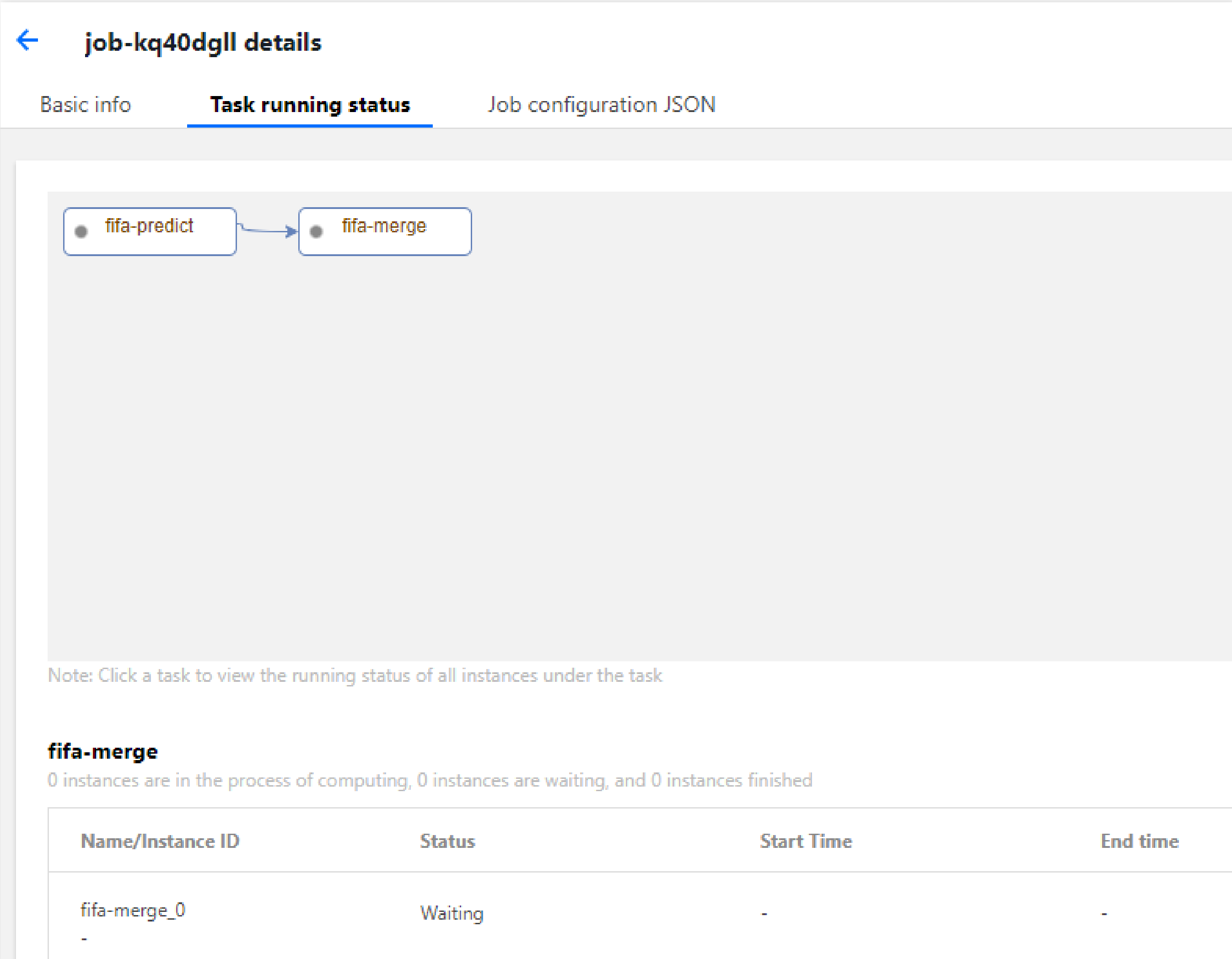
Priority: Keep the default value.

Description: Enter **fifa 2018 model**.

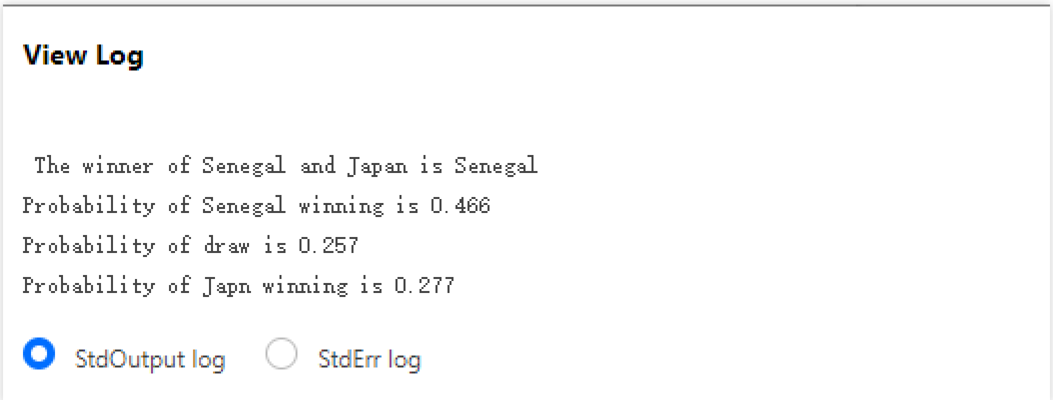
4. On the left side of the **Task flow** page, find the **fifa-predict** and **fifa-merge** tasks and drag them to the canvas on the right. Click the **fifa-predict** task anchor and drag it to the **fifa-merge** task.



5. Confirm the configurations in the task details area on the right side of the **Task flow** page and click **Completed**.
6. For more information about how to query job running information, see Information Query.



7. For more information about how to query rendering results, see Viewing Object Information.



Subsequent Operations

This document illustrates a simple machine learning job to demonstrate basic BatchCompute capabilities. You can continue to test the advanced capabilities of BatchCompute as instructed in the Console User Guide.

Various CVM configurations: BatchCompute provides a variety of CVM configuration options. You can customize your own CVM configuration based on your business scenario.
Remote storage mapping: BatchCompute optimizes storage access and simplifies access to remote storage services into operations in the local file system.
Parallel training of multiple models: With BatchCompute, you can specify the number of concurrent instances and use environment variables to differentiate instances. Each instance reads different training data to achieve parallel modeling.