

HiLens

User Guide

Issue 01
Date 2025-02-24



Copyright © Huawei Technologies Co., Ltd. 2025. 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 Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks 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 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.

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: <https://www.huawei.com>

Email: support@huawei.com

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process*. For details about this process, visit the following web page:

<https://www.huawei.com/en/psirt/vul-response-process>

For vulnerability information, enterprise customers can visit the following web page:

<https://securitybulletin.huawei.com/enterprise/en/security-advisory>

Contents

1 Who Uses HiLens.....	1
2 Procedures for Using HiLens.....	3
3 Permissions Management.....	9
3.1 Creating a User and Granting Permissions.....	9
3.2 Creating a Huawei HiLens Custom Policy.....	10
3.3 Permissions Policies and Supported Actions.....	11
3.3.1 Introduction.....	11
3.3.2 Device Management Permissions.....	12
3.3.3 Skill Development Permissions.....	14
3.3.4 Skill Market Permissions.....	16
3.3.5 Product Management Permissions.....	17
4 Creating a Workspace.....	18
5 Registering HiLens Kit Devices.....	22
5.1 Registering a HiLens Kit Device.....	22
5.2 Registering a Device Using SSH.....	25
5.2.1 Connecting a HiLens Kit Device to a PC.....	25
5.2.2 Upgrading the HiLens Kit System Firmware Version.....	28
5.2.3 Logging In to a HiLens Kit Device Using SSH.....	31
5.2.4 Wireless Network Configuration (SSH).....	34
5.2.5 Wired Network Configuration (SSH).....	37
5.2.6 Registering a HiLens Kit Device via SSH.....	40
5.3 Configure Networking.....	42
5.3.1 Networking Configuration Modes.....	42
5.3.2 Wireless Network (External Network) and Wired Network (Internal Network).....	43
5.4 Configuring a Firewall.....	49
5.5 Using an SD Card.....	50
6 Managing Devices.....	52
6.1 Device Management Overview.....	52
6.2 Managing Skills on Devices.....	53
6.2.1 Skill Management Overview.....	53
6.2.2 Installing a Skill.....	54

6.2.3 Adding Runtime Configurations.....	56
6.2.4 Subscribing to Skill Messages.....	56
6.2.5 Configuring Hard Example Settings.....	60
6.2.6 Starting or Stopping Skills.....	61
6.2.7 Uninstalling Skills.....	62
6.3 Viewing Device Information.....	63
6.4 Viewing Device Logs.....	64
6.5 Upgrading HiLens_Device_Agent Firmware.....	64
6.6 Configuring Cameras.....	65
6.7 Deregistering Devices.....	68
6.8 Viewing Device Alarms.....	69
6.9 Performing One-Click Health Check.....	71
7 Developing Skills on the Console.....	73
7.1 Skill Overview.....	73
7.2 Creating Skills.....	76
7.2.1 Creating a Skill Using a Skill Template.....	76
7.2.2 Creating a Skill Using an Empty Template.....	78
7.3 Obtaining Skill Templates.....	85
7.4 Managing Algorithm Models.....	87
7.4.1 Developing an Algorithm Model.....	88
7.4.2 Importing (Converting) Models.....	88
7.4.3 Model Input Directory Specifications.....	94
7.4.4 Model Output Directory Specifications.....	95
7.5 Writing the Code.....	95
7.6 Editing Skills.....	96
7.7 Installing and Debugging Skills.....	97
7.8 Deleting Skills.....	98
8 Using the Skill Market.....	99
8.1 Skill Market Overview.....	99
8.2 Searching for Skills in the Skill Market.....	101
8.3 Purchasing Skills.....	101
8.4 Managing Orders.....	103
9 Managing Data.....	104
10 Managing Products.....	108
10.1 Overview.....	108
10.2 Managing Products.....	108
10.3 Purchasing Skills (for Devices Running Hi35xx Series Chips).....	111
10.4 Distributing Skills.....	111
10.5 Adding or Deleting Skills.....	113
A Caffe Operator Boundaries.....	115

B TensorFlow Operator Boundaries..... 148

1 Who Uses HiLens

Huawei HiLens is a multimodal AI application development platform featuring a device-cloud synergy. It is intended for use by common users, AI application developers, and software and hardware vendors. It consists of AI inference cameras and a cloud development platform, placing skill development, device deployment and management, and skill market all on one portal, so that users can easily develop AI skills and install them on devices.

Different users may use Huawei HiLens for different purposes. [Table 1-1](#) describes the different types of Huawei HiLens users and the use scenarios. For more information, see [Procedures for Using HiLens](#).

Table 1-1 Introduction to Huawei HiLens users

Role	Typical User	Scenario	Procedure for Using Huawei HiLens
Common users	<ul style="list-style-type: none">• Family members• Shopping mall and supermarket owners• People in charge of construction sites	<ul style="list-style-type: none">• Home: Improve home security.• Shopping malls and supermarkets: Collect statistics on customer foot traffic.• Campus: Recognize vehicle attributes and license plates.• Construction site: Check whether workers are wearing safety helmets on construction sites.	For details, see Common Users Use Skills .

Role	Typical User	Scenario	Procedure for Using Huawei HiLens
Developers	<ul style="list-style-type: none"> • Technical personnel engaged in AI development • University students 	<p>Develop skills with AI capabilities and release them to the skill market to build a robust AI developer ecosystem.</p> <ul style="list-style-type: none"> • Developing new skills on the Console 	<p>Useful links:</p> <p>Developers Develop Skills on the Management Console</p>
Manufacturers	Manufacturers of cameras running HiSilicon Hi35xx series chips	<ul style="list-style-type: none"> • Camera manufacturers: Power AI capabilities for entry-level and mid-range cameras. 	<p>For details, see Manufacturers Manage Products.</p>

2 Procedures for Using HiLens

Huawei HiLens provides device management, skill development, data management, skill market, and product management functions for three types of users, helping them develop AI skills for computing devices.

NOTE

For details about how to use HiLens Kit devices, see the [HiLens Kit User Guide](#).

The functions users need vary depending on application scenarios. This section describes the procedures for using Huawei HiLens based on the three types of users and their application scenarios. You can quickly understand functions provided by Huawei HiLens by reading [Huawei HiLens Function Overview](#) and click the links in this table to go to the corresponding sections to learn details.

The procedures for using HiLens are as follows. Only the main procedures are listed here. For details about other management operations, see [Huawei HiLens Function Overview](#).

- [Common Users Use Skills](#)
- [Developers Develop Skills on the Management Console](#)
- [Manufacturers Manage Products](#)

Prerequisites

- Before using Huawei HiLens, you need to create a Huawei Cloud account. Through this account, you can have access to all Huawei Cloud services and pay only for the services you use.
- Huawei HiLens depends on other services. Before using this service, you need to obtain the permissions of other related services, including ModelArts, Object Storage Service (OBS), and Software Repository for Container (SWR). For details about the relationships between Huawei HiLens and other services, see [Related Services](#).

Common Users Use Skills

Common users refer to users who use HiLens Kit devices for intelligent surveillance in scenarios such as smart home, shopping malls, campuses, and construction sites.

Figure 2-1 Procedure for common users

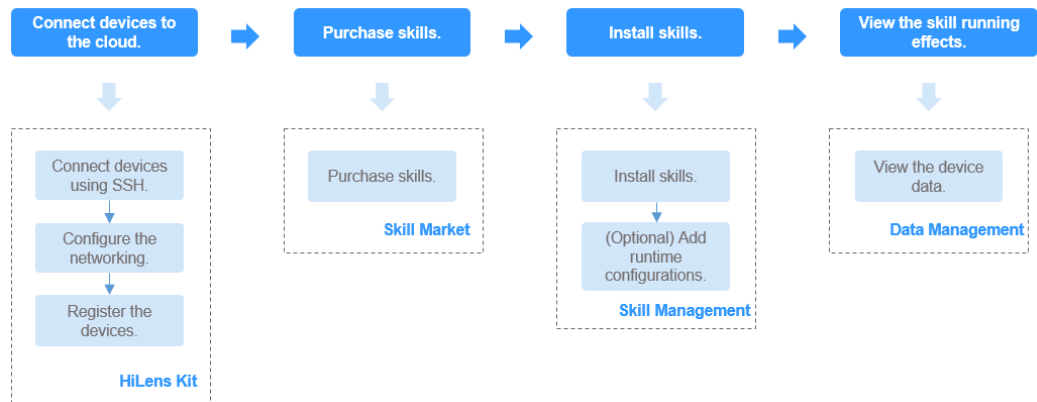


Table 2-1 Procedure for common users

Procedure	Description	Detailed Instructions
Registering HiLens Kit Devices	Connect to the HiLens Kit devices you purchased and register them with the Huawei HiLens platform to connect them to the cloud.	Registering a HiLens Kit Device Registering a Device Using SSH
Upgrading HiLens Device Agent Firmware	Upgrade the HiLens_Device_Agent to version 1.0.7 or later.	Upgrading HiLens_Device_Agent Firmware
Purchasing Skills	Purchase existing skills or custom skills you need from the skill market and install them onto your devices to deploy AI capabilities. In the skill market, select the skills designed to run on Ascend chips.	Purchasing Skills
Installing Skills	Install the purchased skills on your devices.	Installing a Skill
Viewing Skill Performance	View the device data and how the skills have been performing.	Managing Data

Developers Develop Skills on the Management Console

Huawei HiLens provides a skill development platform, on which you can develop skills to run on Ascend 310 or HiSilicon Hi35xx series chips based on site requirements. Skills tailored-made for Ascend 310 chips can be directly installed on HiLens Kit devices.

Devices running HiSilicon Hi35xx series chips typically have a small on-chip memory and small capacities in other dimensions as well. It is challenging to

develop skills that run well on such devices. You need to optimize the skill models. If you encounter any difficulty, contact Huawei HiLens platform personnel for support.

- In the process of developing a skill, you need to deploy the skill onto a test device to see how the skill performs. Therefore, developers are advised to purchase at least one HiLens Kit device.
- Developers need to train AI models locally or in ModelArts to develop skills.

Figure 2-2 shows the skill development procedure, and Table 2-2 describes the procedure in detail.

Figure 2-2 Procedure for skill development by developers

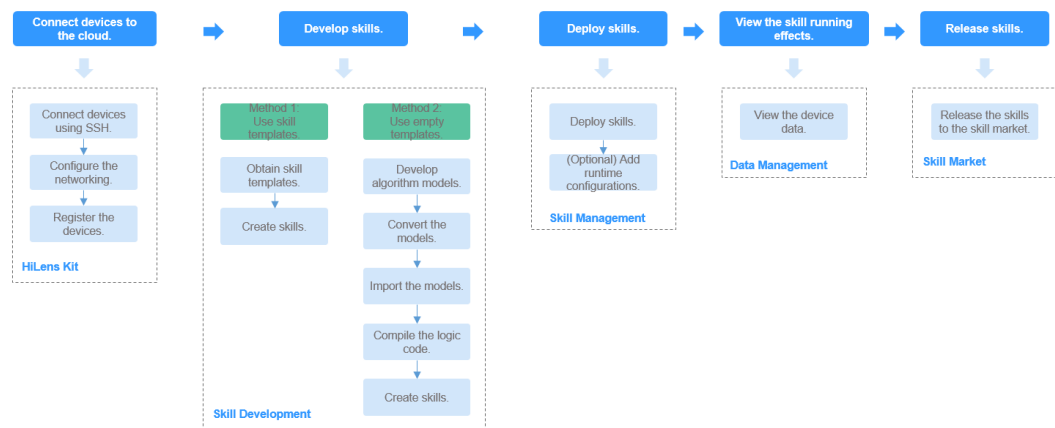


Table 2-2 Procedure for skill development by developers

Procedure	Description	Detailed Instructions
Registering Devices	Register the devices to the Huawei HiLens platform and connect the devices to the cloud. Huawei HiLens supports HiLens Kit devices and Atlas500, Atlas 800, IVS 1800, and more devices that are being tested for compatibility.	Registering HiLens Kit Devices
Developing Skills	Creating a Skill Using a Skill Template: Obtain a skill template (containing the algorithm model and logic code) provided by Huawei HiLens to quickly create a skill.	Obtaining Skill Templates Creating a Skill Using a Skill Template

Procedure	Description	Detailed Instructions
	<p>Creating a Skill Using an Empty Template: To meet more service requirements, you can develop an algorithm model, import the developed model to Huawei HiLens, compile the logic code, and create a skill based on the customized algorithm model and logic code.</p> <p>If your algorithm model does not meet the format requirements of Huawei HiLens, you can use the model conversion function.</p>	<p>Developing an Algorithm Model</p> <p>Importing (Converting) Models</p> <p>Writing the Code</p> <p>Creating a Skill Using an Empty Template</p>
Deploying Skills	Deploy skills on devices to facilitate debugging.	Installing and Debugging Skills
Viewing Skill Performance	View the device data and how the skills have been performing.	Managing Data

Manufacturers Manage Products

Huawei HiLens provides a product management process for manufacturers who produce cameras that run HiSilicon Hi35xx series chips. With Huawei HiLens, the manufacturers can manage devices, purchase skills, and distribute licenses to devices so that all of the devices can have AI capabilities.

[Figure 2-3](#) shows the product management procedure, and [Table 2-3](#) describes the procedure in detail.

Figure 2-3 Procedure for product management by manufacturers

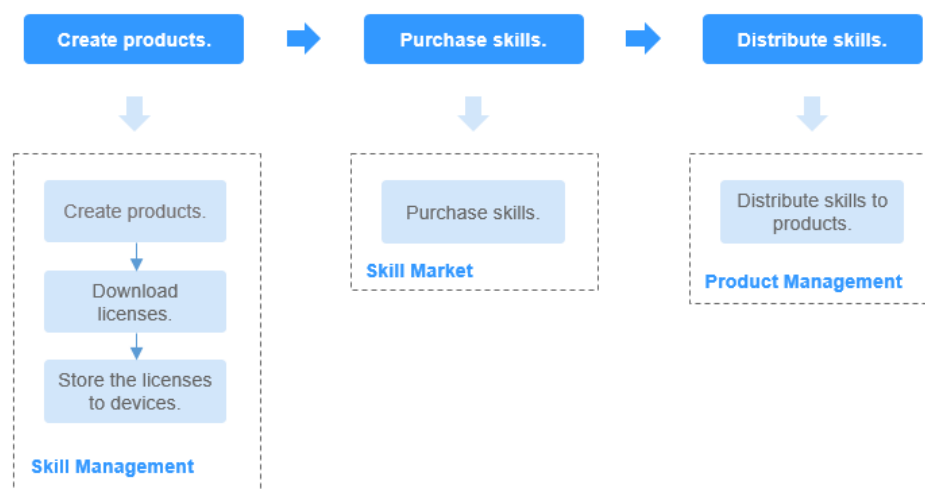


Table 2-3 Procedure for product management by manufacturers

Procedure	Description	Detailed Instructions
Creating Products	Create a product and associates the devices with the product.	Managing Products
Purchasing Skills	Purchase required skills (used for HiSilicon Hi35xx series chips) from the skill market and distribute them to user devices.	Purchasing Skills
Distributing Skills	Distribute the purchased skills to devices, download corresponding SDKs, and integrate the SDKs to the devices for users to use.	Distributing Skills

Huawei HiLens Function Overview

Table 2-4 Function overview

Chapter	User Role	Description
Registering an Account	All	Create a Huawei Cloud account. Through this account, you can have access to all Huawei Cloud services and pay only for the services you use. Before using Huawei HiLens, you also need to obtain the authorization of related services.
Registering HiLens Kit Devices	<ul style="list-style-type: none"> Common users Developers 	Before using or developing skills, you need to purchase devices and connect them to the Huawei HiLens console on the cloud. For details, see the operation guide provided.
Managing Devices	<ul style="list-style-type: none"> Common users Developers 	You can manage registered HiLens Kit devices on the Huawei HiLens console, including skill management, firmware upgrade, and camera configuration. Go to the Device Management page to see your device management options (both users and developers).

Chapter	User Role	Description
Developing Skills	Developers	<p>You can create skills on the Huawei HiLens platform by developing and combining algorithms, models, and logic code, or by using preset skill templates. In addition, you can debug a newly developed skill on a test device.</p> <p>Then, you can release the skills that have been debugged to the skill market for more users to use. In return, you will get the appropriate amount of reward.</p>
Skill Market	All	<p>The skill market is an open platform that provides skills tailor-made for different chips and application scenarios. You can purchase skills in the skill market. If you are a developer, you can release your skills to the skill market and get paid.</p>
Managing Data	<ul style="list-style-type: none"> • Common users • Developers 	<p>You can view the video data of devices registered with Huawei HiLens to see how skills perform.</p>
Managing Products	Manufacturers	<p>Manufacturers of devices that run HiSilicon Hi35xx series chips can use the Huawei HiLens platform to manage products, purchase the needed skills, distribute the skills to devices, download SDKs, and integrate the SDKs to the devices to enable them with AI capabilities.</p>

3 Permissions Management

3.1 Creating a User and Granting Permissions

This section describes how to use [Identity and Access Management \(IAM\)](#) to implement fine-grained permissions control for your Huawei HiLens. With IAM, you can:

- Create IAM users for employees based on the organizational structure of your enterprise. Each IAM user has their own security credentials for access to Huawei HiLens.
- Grant only the permissions required for users to perform a specific task.
- Entrust other Huawei Cloud accounts or cloud services to perform efficient O&M on your Huawei HiLens resources.

If your Huawei Cloud account does not require individual IAM users, skip this section.

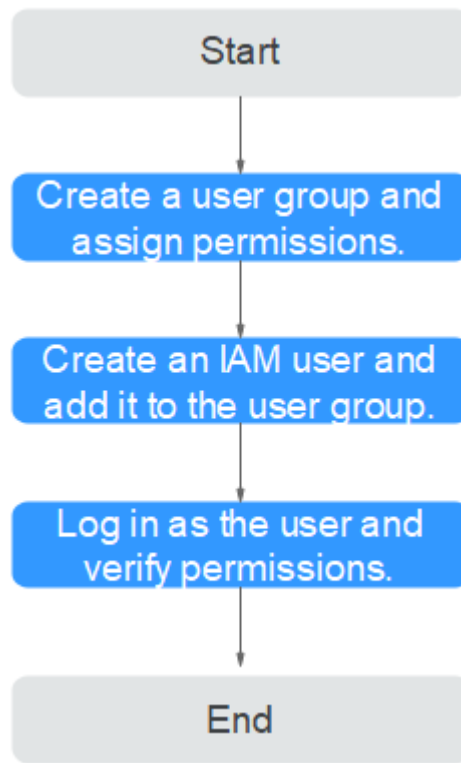
This section describes the procedure for granting permissions. See [Figure 3-1](#).

Prerequisites

Before granting permissions to a user group, you need to learn about the Huawei HiLens permissions that can be added to the user group and select the permissions based on site requirements. For details, see [Permissions Management](#). For the system policies of other services, see [System Permissions](#).

Procedure

Figure 3-1 Procedure for granting Huawei HiLens permissions



1. **Create a user group and assign permissions** to it.
Create a user group on the IAM console, and assign the **HiLens CommonOperations** policy to the group. To apply for the OBT, and set alarm receiving and skill messages, you must be granted the **SMN Administrator** role.
2. **Create an IAM user and add it to the user group.**
Create a user on the IAM console and add the user to the group created in **1**.
3. **Log in** as the user and verify permissions.
Log in to the Huawei HiLens console as the created user, switch to the authorized region, and verify the permissions.
Choose **Service List > Huawei HiLens** to enter the Huawei HiLens home page. Then, choose **Device Management > Devices**, locate the target device card, and choose **Operation > Deregister** (if the device needs to be activated after being registered, refund its management fee before deregistering it). If you cannot deregister the device (assume that the current permission contains only **HiLens CommonOperations**), **HiLens CommonOperations** has already taken effect.

3.2 Creating a Huawei HiLens Custom Policy

Custom policies can be created as a supplement to the system policies of Huawei HiLens. For the actions supported for custom policies, see [Introduction](#).

You can create custom policies in either of the following ways:

- **Visual editor:** Select cloud services, actions, resources, and request conditions without the need to know policy syntax.
- **JSON:** Edit JSON policies from scratch or based on an existing policy.

For details, see [Creating a Custom Policy](#). This section describes common custom policy examples of Huawei HiLens.

A Sample Custom Policy of Huawei HiLens

Example: Denying deletion of developed skills

A deny policy must be used in conjunction with other policies to take effect. If the permissions assigned to a user contain both Allow and Deny actions, the Deny actions take precedence over the Allow actions.

The following method can be used if you need to assign permissions of the **Huawei HiLens FullAccess** policy to a user but also forbid the user from deleting developed skills. You can create a custom policy for denying deletion of developed skills and assign both policies to the group the user belongs to. Then the user can perform all operations on Huawei HiLens except deleting developed skills. The following is an example deny policy:

```
{
  "Version": "1.1",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "hilens:project:deleteProject"
      ]
    }
  ]
}
```

3.3 Permissions Policies and Supported Actions

3.3.1 Introduction

You can use IAM to implement fine-grained permissions management for your Huawei HiLens resources. If your Huawei Cloud account does not need individual IAM users, then you may skip this section.

By default, new IAM users do not have permissions assigned. You need to add the users to one or more groups, and attach permissions policies or roles to these groups. Users inherit permissions from the groups to which they are added and can perform specified operations on cloud services based on the permissions.

You can grant users permissions by using [roles](#) and [policies](#). Roles: A type of coarse-grained authorization mechanism provided by IAM that defines permissions related to user responsibilities. Policies: Policies define API-based permissions for operations on specific resources under certain conditions, allowing for more fine-grained, secure access control of cloud resources.

 **NOTE**

If you want to allow or deny the access to an API, fine-grained authorization is a good choice.

An account has all the permissions required to call all APIs, but IAM users must be assigned the required permissions. The permissions required for calling an API are determined by the actions supported by the API. Only users who have been granted permissions allowing the actions can call the API successfully. For example, if an IAM user queries ECSs using an API, the user must have been granted permissions that allow the **hilens:servers:list** action.

Supported Actions

Huawei HiLens provides system-defined policies that can be directly used in IAM. You can also create custom policies and use them to supplement system-defined policies, implementing more refined access control. Operations supported by policies are specific to APIs. The following are common concepts related to policies:

- **Permission:** a statement in a policy that allows or denies certain operations.
- **Action:** added to a custom policy to control permissions for specific operations.
- **Authorization scope:** A custom policy can be applied to IAM projects or enterprise projects or both. Policies that contain actions supporting both IAM and enterprise projects can be assigned to user groups and take effect in both IAM and Enterprise Management. Policies that only contain actions for IAM projects can be used and only take effect for IAM. For details, see [Differences Between IAM and Enterprise Management](#).
- **API:** RESTful APIs that can be called in a custom policy.

Huawei HiLens supports the following actions that can be defined in custom policies:

- [Device Management Permissions](#)
- [Skill Development Permissions](#)
- [Skill Market Permissions](#)
- [Product Management Permissions](#)

3.3.2 Device Management Permissions

Table 3-1 Device management

Permission	Method	Action	Minimum Dependent Action
Deregister a device.	PUT	hilens:device:deleteDevice	-
Upgrade device firmware.	PUT	hilens:device:upgradeDeviceFirmware	-

Permission	Method	Action	Minimum Dependent Action
Register a device.	POST	hilens:device:registerDevice	-
Update the device information.	PUT	hilens:device:updateDevice	-
Update the device configuration information.	PUT	hilens:device:updateDeviceConfig	-
Obtain the device configuration information.	GET	hilens:device:getDeviceConfig	-
Obtain the list of devices.	GET	hilens:device:getDeviceList	-
Obtain the list of skills installed on a device.	GET	hilens:device:getDeviceSkillList	-
Uninstall a specified skill on a specified device.	DELETE	hilens:skill:uninstallSkill	-
Update the configuration information of a specified skill on a specified device.	PUT	hilens:skill:updateSkillConfig	-
Start a specified skill on a specified device.	POST	hilens:skill:startSkill	-
Stop a specified skill on a specified device.	POST	hilens:skill:stopSkill	-
Obtain the configuration information of a specified skill on a specified device.	GET	hilens:skill:getConfig	-

3.3.3 Skill Development Permissions

Table 3-2 Skill development permissions

Permission	Method	Action	Minimum Dependent Action
Create a model conversion job.	PUT	hilens:model:convertModel	-
Release a developed skill to the skill market.	PUT	hilens:project:publishProject	obs:object:GetObject obs:object:PutObject obs:object:PutObjectAcl
Download a skill template.	PUT	hilens:template:downloadTemplate	obs:object:GetObject
Delete a skill model.	DELETE	hilens:model:deleteModel	-
Create a skill.	POST	hilens:project:createProject	obs:object:GetObject obs:bucket:HeadBucket obs:bucket>CreateBucket obs:object:PutObject obs:bucket>ListAllMyBuckets obs:bucket>ListBucket
Edit a developed skill.	PUT	hilens:project:updateProject	obs:object:GetObject obs:bucket:HeadBucket obs:bucket>CreateBucket obs:object:PutObject obs:bucket>ListAllMyBuckets obs:bucket>ListBucket

Permission	Method	Action	Minimum Dependent Action
Import a skill model to Huawei HiLens.	POST	hilens:model:importModel	obs:object:GetObject obs:bucket:HeadBucket obs:bucket:ListAllMyBuckets obs:bucket:ListBucket
Delete a developed skill.	DELETE	hilens:project:deleteProject	-
Add a skill template to favorites.	POST	hilens:template:collectTemplate	-
Deploy a developed skill to a device.	PUT	hilens:project:deployProject	obs:object:GetObject
Remove a skill template from favorites.	DELETE	hilens:template:deleteCollectedTemplate	-
Obtain the list and details of model conversion jobs.	GET	hilens:model:getConvertJob	-
Obtain the details about a skill template.	GET	hilens:template:getTemplate	-
Obtain the details about a skill model.	GET	hilens:model:getModel	-
Obtain the details about a developed skill.	GET	hilens:project:getProject	-
Obtain the list of model conversion jobs.	GET	hilens:model:getConvertJobList	-
Obtain the list of developed skills.	GET	hilens:project:getProjectList	-
Obtain the list of skill models.	GET	hilens:model:getModelList	-

Permission	Method	Action	Minimum Dependent Action
Obtain the list of skill templates.	GET	hilens:template:getTemplateList	-

3.3.4 Skill Market Permissions

Table 3-3 Skill market management

Permission	Method	Action	Minimum Dependent Action
Download a skill from the skill market.	GET	hilens:market:downloadSkill	obs:object:GetObject
Create a skill order in the skill market.	POST	hilens:market:createSkillOrder	-
Install a skill purchased from the skill market.	POST	hilens:market:installSkill	-
Remove a released skill from the catalog of the skill market.	PUT	hilens:market:withdrawSkill	-
Obtain the skill billing information.	GET	hilens:market:getSkillBillingInfo	-
Obtain the skill details of the skill market.	GET	hilens:market:getSkillInfo	-
Obtain the skill order list of the skill market.	GET	hilens:market:getSkillOrderList	-
Obtain the skill list of the skill market.	GET	hilens:market:getSkillList	-

3.3.5 Product Management Permissions

Table 3-4 Product management

Permission	Method	Action	Minimum Dependent Action
Distribute skill orders to products.	POST	hilens:product:createProductLicense	-
Update the skill order information of a product.	PUT	hilens:product:updateProductLicense	-
Delete a product.	DELETE	hilens:product:deleteProduct	-
Delete the skill order information of a product.	DELETE	hilens:product:deleteProductLicense	-
Create a product.	POST	hilens:product:createProduct	-
Update the product information.	PUT	hilens:product:updateProduct	-
Obtain the list of products.	GET	hilens:product:getProductList	-
Obtain the skill order list of a product.	GET	hilens:product:getProductLicenseList	-

4 Creating a Workspace

Workspace is a way of isolating resources, such as models and skills, between different users.

Resource Isolation

Each Huawei Cloud account or IAM account has a default workspace, which can be used to view all resources created by the current account on Huawei HiLens.

The table below lists resources that are isolated between different workspaces.

Table 4-1 Isolated resources

Navigation Tree	Isolated Resource
Device Management > Devices	Registered HiLens Kit devices and device resources, including skills, cameras, alarms, and firmware versions
Skill Development > Models	Models imported (converted) to Huawei HiLens
Skill Development > Skills	Skill resource created on the console
Data Management (Beta)	Logs generated when devices in different workspaces run skills

The table below lists resources that are shared by all workspaces.

Table 4-2 Shared resources

Navigation Tree	Shared Resource
Skill Market	Skills that can be shared in the skill market and skills purchased by the current account

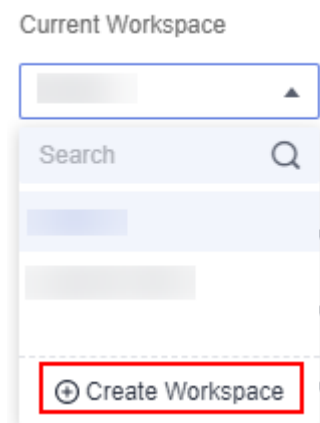
Navigation Tree	Shared Resource
Device Management > Products	Newly created cameras with HiSilicon Hi35xx series chips on Huawei HiLens
Skill Development > Skill Templates	Templates built in Huawei HiLens and templates added to My Favorites by the current account

Creating a Workspace

1. Log in to the Huawei HiLens management console. In the navigation pane, choose **Create Workspace** from the drop-down list box under **Current Workspace**.

The **Create Workspace** page is displayed.

Figure 4-1 Creating a workspace



2. Enter the workspace name and description, and click **Create Now** in the lower right corner. The **Workspaces** page is displayed. You can view the newly created workspace.

The workspace description cannot contain the following characters: #~^\$%&*<>{}[]\|

Figure 4-2 Creating a workspace-0

Allocating Workspace

You can move devices and their resources such as skills and cameras from a workspace to another workspace.

This section uses **test** and **test2** as an example to describe how to allocate workspace. **test** is the original workspace where device resources are located and **test2** is the target workspace for device resource allocation .

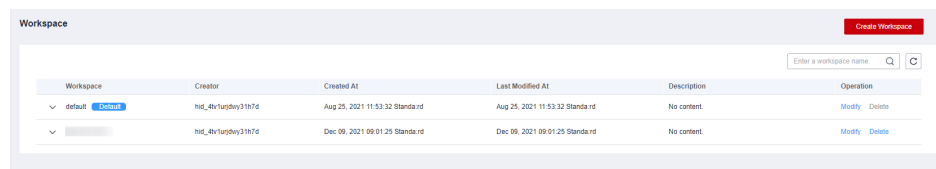
1. Log in to the Huawei HiLens management console. In the navigation pane, select the **test** workspace from the drop-down list box under **Current Workspace**.
2. In the navigation tree on the left, choose **Device Management > Devices**, select the devices to be allocated, and click the device cards.
The device details page is displayed.
3. Click **Allocate Workspace** in the upper right corner.
The **Allocate Workspace** dialog box is displayed.
4. Select workspace **test2** and click **OK**.
A message is displayed in the upper right corner, indicating that the devices are allocated successfully. You can switch to the workspace **test2** and view the allocated device resources on the **Device Management > Devices** page.

Modifying a Workspace

Currently, you can change the name and description of a custom workspace, but cannot change the name of the default workspace.

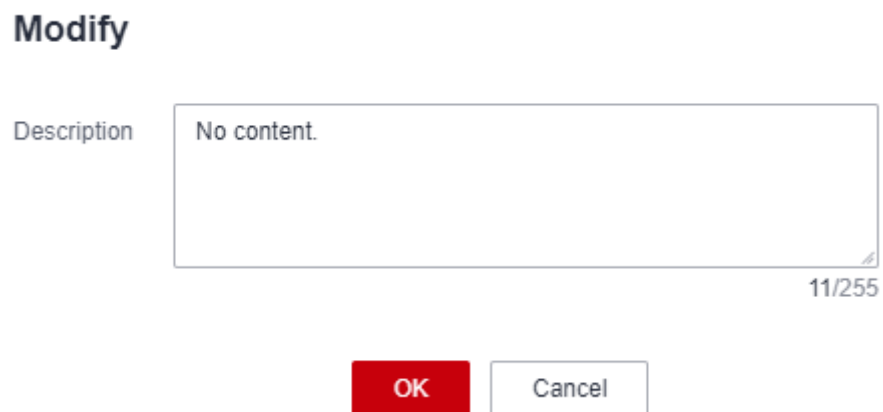
1. Log in to the Huawei HiLens console. In the navigation pane, choose **Workspaces**.
The **Workspaces** page is displayed.

Figure 4-3 Workspace



2. Select the workspace to be modified and click **Modify** in the **Operation** column.
3. In the dialog box, modify the workspace description, and click **OK**.

Figure 4-4 Modifying a workspace

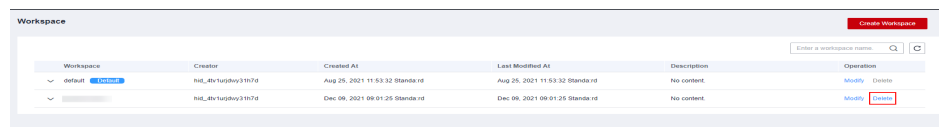


Deleting a Workspace

You can delete a workspace that is no longer used to release resources. The default workspace cannot be deleted.

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Workspace**.
The **Workspace** page is displayed.
2. Select the workspace to be deleted and click **Delete** in the **Operation** column.

Figure 4-5 Deleting a workspace

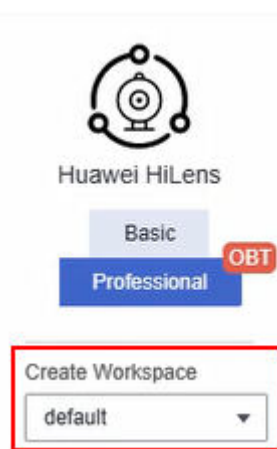


3. Confirm the information in the dialog box and click **OK**.

Follow-up Operations

When there are several workspaces, you can select a workspace from the drop-down list under **Current Workspace** in the navigation pane to use your Huawei HiLens resources. For more information, see [Procedures for Using HiLens](#).

Figure 4-6 Current workspace



5 Registering HiLens Kit Devices

5.1 Registering a HiLens Kit Device

Introduction to HiLens Kit

HiLens Kit is a brand of multimedia devices with AI inference capabilities. It packs powerful computing, HD camera access, and multiple ports into a compact kit. Its hardware has an integrated Atlas 200 AI Accelerator Module (Atlas 200 for short) for quick access to the network and maximized computing power of the Ascend 310 AI processor. The HiLens Kit device is perfect for AI developers in pre-research and development as well as video surveillance domains. Its built-in HiLens Framework provides abundant APIs for running skills.

A HiLens Kit device provides an onboard 32 GB eMMC and one micro SD slot for connecting an external micro SD card. The micro SD card supports a maximum rate of SDR50 and a maximum capacity of 2 TB. For details about how to connect a micro SD card to a HiLens Kit device, see [Using an SD Card](#). For detailed parameters, see [Table 5-1](#). For details about the product, see [HiLens Kit User Guide > Product Description](#).

Figure 5-1 HiLens Kit



Table 5-1 HiLens Kit parameters

Chip	CP U	Micro SD	Camera	Mic rop hon e	Net work Port	U SB	HD MI	Audi o Out	OS	W i- Fi
Asce nd 310 (8 GB RAM)	Hi 35 59 A (4 G B)	1	2- megapix el, 720p	2	1	2	1	1	Lin ux	Su pp or te d

Reference documents

When you obtain the documents for the first time, create a new account and [register your product](#) in the enterprise technical support website (Support-E website). Enter the product serial number (SN) of the HiLens Kit device to complete product registration. For details, see [Enhance My Account Privilege](#).

The SN is at the bottom of the HiLens Kit device and is a string of 20 characters, for example, **21023XXXXXXXXXXXXXXXXXX**.

After the product registration application is submitted, if the system displays a message indicating that the product registration is successful, go to the next step. If the system displays a message indicating that the product is to be reviewed, wait until the product is successfully reviewed and go to the next step. The product will be reviewed within one workday.

- [HiLens Kit User Guide](#)

This document describes the appearance, logical structure, and specifications of the Atlas 200 HiLens Kit and provides guidance for users to install, connect, power on, power off, and configure the Atlas 200 HiLens Kit.

- This document describes the system architecture of the Huawei HiLens intelligent edge system and guide users through managing and maintaining the Atlas 200 HiLens Kit using this system.

Methods for registering a HiLens Kit device

To enable HiLens Kit devices with AI skills, you need to register them with the Huawei HiLens console so that you can manage the devices and their skills on the console.

This section describes how to log in to a HiLens Kit device, register it with the cloud-based Huawei HiLens console, and manage it on the console.

- Currently, only HiLens Kit devices released by Huawei can be registered.
- You can install and use only skills "oriented to Ascend 310 chips" on HiLens Kit devices.

The following are the processes and operation guides:

- [Registering a Device Using SSH](#)

Common operations

Configuration Initialization: The HiLens Kit device comes with the pre-installed Euler operating system developed by Huawei. You do not need to install one. You can log in to the Huawei HiLens intelligent edge system using a browser to initialize the configuration.

Changing the Initial User Name and Password: For security purposes, change the initial password upon the first login and periodically change the password.

Configuring a Firewall: To prevent users beyond the allowed IP address range from accessing the HiLens Kit intelligent edge system, you can configure a firewall to prevent the system from being attacked.

Restoring Factory Settings: If the system of the HiLens Kit device is damaged by mistake and cannot be restored to normal use, you need to restore the factory settings.

Using an SD Card: When using a HiLens Kit device, insert an SD card to store skill data.

Registering a Device Using SSH

Registering a device using SSH indicates that you log in to the HiLens Kit system using SSH and run Linux commands to register a device. It allows you to configure system files of HiLens Kit.

Figure 5-2 shows the registration procedure. **Table 5-2** describes the procedure in detail.

Figure 5-2 Procedure for registering a device using SSH

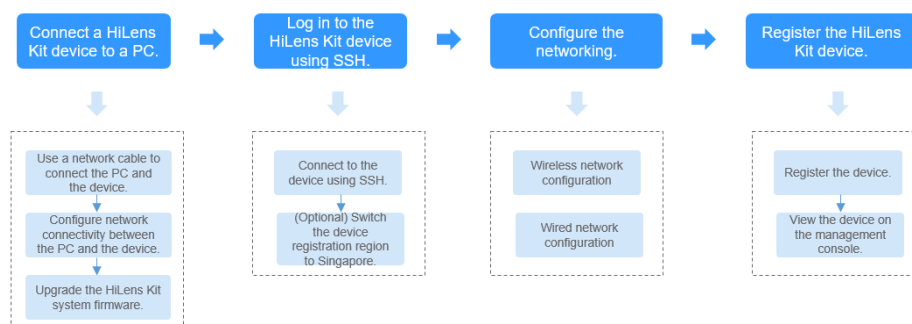


Table 5-2 Procedure description

Procedure	Description	Detailed Instructions
Connect a HiLens Kit device to a PC.	Before registering a HiLens Kit device with the Huawei HiLens console, you need to connect the HiLens Kit device to a PC, and upgrade the HiLens Kit system firmware version to 2.2.200.011.	Connecting a HiLens Kit Device to a PC
Configure the networking.	A HiLens Kit device can be connected to a router in wireless or wired mode. You can select either of the two networking modes. A router cannot be connected to a wired network and a wireless network at the same time. If you connect the router to a wireless network, the default gateway will be automatically deleted. If you connect the router to a wired network again, you need to configure the default gateway. You are advised to connect the router to a wireless network, in case you forget the device IP address after updating it.	Wireless Network Configuration (SSH) Wired Network Configuration (SSH)
Register the HiLens Kit device.	Register the device with the console and check the device status on the console.	Registering a HiLens Kit Device via SSH

5.2 Registering a Device Using SSH

5.2.1 Connecting a HiLens Kit Device to a PC

To operate configurations such as system files of a HiLens Kit device, you can register it using SSH, log in to it, and run corresponding Linux commands. The

following uses PuTTY as an example to describe how to log in to a HiLens Kit device using SSH in a Windows 7 environment.

Preparations

- You have at least one HiLens Kit device.
- Your account is not in arrears.
- If you use SSH to register a device, the system firmware version must be 2.2.200.011 or later. If your firmware version is earlier than 2.2.200.011, [upgrade it first](#).
To query the system firmware version, [log in to the HiLens IES web UI](#), and choose **Maintenance > Firmware Upgrade** to view the current version number.
- Ensure that you have obtained the following data:
 - IP address of the device to be connected
For details about the initial IP address of the device, see the **Default Value of Initial IP address of the management network port** in .
 - Username and password for logging in to the device to be connected
For details about the initial username and password of the device, see the **Default Value of Initial user name and password** in .
If this is not your first login, use your new account name and password.
For details about how to change the password, see [Changing the Initial User Name and Password](#).
- Ensure that you have downloaded PuTTY.exe. This is third-party software.

Connecting a HiLens Kit Device to a PC Using a Network Cable

[Figure 5-3](#) and [Table 5-3](#) show the ports on the rear panel of the HiLens Kit device.

Figure 5-3 Ports on the rear panel



Table 5-3 Description of ports on the rear panel



Port	Description
1	Power button

Port	Description
2	Power socket
3	Management network port

1. Connect a 12 V DC power adapter to the power socket on the rear panel of the HiLens Kit device.
2. Press and hold the power button for 1 to 2 seconds to power on the HiLens Kit device.
3. Connect one end of the network cable to the management network port of the device and the other end to an Ethernet port of the PC.

Interconnecting the Network Between the PC and the HiLens Kit Device

Set an IP address and a subnet mask or route for the PC so that the PC can properly communicate with the device.

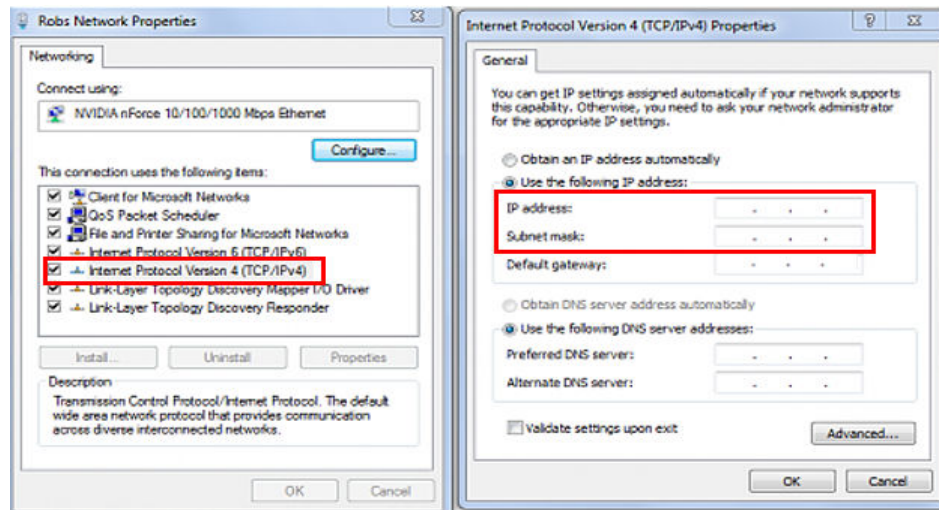
1. Click the network icon  in the lower right corner of the PC and click **Network & Internet settings**.
2. On the **Network & Internet settings** page, click **Change adapter settings** to enter the **Network Connections** page.
3. After the HiLens Kit device is connected to the PC using a network cable, a corresponding connection  is displayed on the **Network Connections** page of the PC. Right-click the connection (generally named **Local Area Connection**) and choose **Properties** from the shortcut menu. The **Local Area Connection Properties** dialog box is displayed.
4. Double-click **Internet Protocol Version 4** and select **Use the following IP address**. Then, enter an IP address that is in the **same network segment** as the device IP address and click the **Subnet mask** text box to automatically generate a subnet mask. Click **OK**.

For details about the initial IP address of the device, see the **Default Value of Initial IP address of the management network port** in [HiLens Kit User Guide > Default Credentials](#).

NOTE

- You can connect to the HiLens Kit device using SSH from your local PC only when the IP address configured in **Local Area Connection Properties > Internet Protocol Version 4 Properties** is in the same network segment as the device IP address. **The same network segment** means that the first three octets of the local connection IP address are the same as that of the device IP address. For example, if the device IP address is 192.168.2.111, the IP address (in the same network segment as the device IP address) configured in **Local Area Connection Properties > Internet Protocol Version 4 Properties** can be 192.168.2.x, where x is an integer from 2 to 255 except 111.
- If the device IP address has been modified, configure the IP address in **Local Area Connection Properties > Internet Protocol Version 4 Properties** to an IP address in the same network segment as the new device IP address. For details about how to modify an IP address, see [Wired Network Configuration \(SSH\)](#).

Figure 5-4 Modifying the Attributes of a Network in the VPC



If you are registering a HiLens Kit device for the first time, connect the HiLens Kit device to the PC and perform operations described in [Upgrading the HiLens Kit System Firmware Version](#).

Follow-up Operations

Log in to the HiLens Kit device using SSH. For details, see [Connecting a HiLens Kit Device to a PC](#).

5.2.2 Upgrading the HiLens Kit System Firmware Version

Before registering a device, upgrade the firmware of HiLens Kit to 2.2.200.011 (TR6). Before the upgrade, carefully read the following content.

Upgrade Prerequisites and Precautions

- If the system firmware version is earlier than 2.2.200.011, you can upgrade it. To query the system firmware version, [log in to the HiLens IES web UI](#), and choose **Maintenance > Firmware Upgrade** to view the current version number.

For details, see the [HiLens Kit Update Guide](#).

- After the HiLens Kit system firmware is upgraded, its version becomes 2.2.200.011, and the HiLens_Device_Agent firmware will also be upgraded to 1.0.6. (If the HiLens Kit device is in use and the current HiLens_Device_Agent firmware version is later than 1.0.6, the latest HiLens_Device_Agent firmware version is retained.)
- If your HiLens Kit device is newly registered and used for the first time, you are advised to upgrade its firmware to the latest version to ensure better system stability.
- If your HiLens Kit device has been registered and is in use, you are advised to carefully read [upgrade risks](#) and determine whether to perform the upgrade.

Upgrade Risks

Before the upgrade, read the following risks carefully and exercise caution when performing the upgrade.

- Before upgrading the firmware version of a HiLens Kit device that has been registered and is in use, contact the contact person to evaluate the risk of the upgrade on service continuity.
- Upgrading the HiLens Kit system firmware version will reinstall the entire system directory. As a result, the software installed in the system directory will be lost. To locate the system directory, run the **df -h** command to view the current partition information. The root directory corresponding to **/dev/mmcbk0p2** is the system directory. This directory is the default software installation directory. During the upgrade, the software installed in it will be lost. Files in other non-system directories, such as the **/dev/mmcbk0p*** directory, are not affected during the upgrade.

Tips: To prevent the system directory from affecting the upgrade, you can redirect software packages that support redirection installation to the **/opt** directory of the HiLens Kit device system to avoid file loss in subsequent system upgrades. For example, you can run the **yum --installroot=/opt install xxx** command to redirect the YUM installation software package **XXX**. Software packages that do not support redirection installation, such as RPM packages involved in YUM and ROS installation, can be deployed only in the system directory and will be lost after the upgrade.

- After the HiLens Kit system firmware version is upgraded, some of the original skills may fail to run. Exercise caution when performing this operation.

Upgrade Advantages

- The device can be logged in using user **admin** and keep online for up to one hour.

After the upgrade, you can run the **timeout** command as user **admin** to set the timeout period. The maximum timeout period is 1 hour. For example, to set the timeout period to 1 hour, run the **timeout 3600** command.

- You must change the default username and password upon the first login to the intelligent edge system to prevent security risks caused by the use of the default username and password.
- Hardware and resource monitoring is also available. When the hardware is abnormal or the resource usage exceeds the alarm threshold, an alarm is reported.
- Security upgrade is performed on third-party components.

Prerequisites

Use a network cable to connect a HiLens Kit device to a PC. For details, see [Connecting a HiLens Kit Device to a PC](#).

Upgrade Procedure

If you have registered and used your HiLens Kit devices, carefully read [Upgrade Risks](#) before the upgrade.

Upgrade the system firmware version **twice**. The operations are as follows:

For details, see the [HiLens Kit Update Guide](#).

1. When you obtain the upgrade package for the first time, you need to register an account and **register products** with the enterprise technical support website (Support-E website), enhance your account privilege, and enter the product serial number (SN) of the HiLens Kit device to complete product registration. For details, see [Enhance My Account Privilege](#).

The SN is marked at the bottom of the HiLens Kit device and is a string of 20 characters, for example, **21023XXXXXXXXXXXXXXXXXX**.

After the product registration application is submitted, if the system displays a message indicating that the product registration is successful, go to the next step. If the system displays a message indicating that the product is to be reviewed, wait until the product is successfully reviewed and go to the next step. The product will be reviewed within one workday.

2. Log in to the [Huawei Enterprise website](#), choose **Technical Support > Product Support > Server - Intelligent Computing > Ascend Computing**. Under **Intelligent Edge Hardware**, click [A200-3000HiLens](#).

If an incorrect product is selected, the SN fails to pass the check and the upgrade cannot be performed.

3. Click **Software Download**. In the software version list, select **A200-3000HiLens-FWV2.2.200.011.hpm** and download it to the local PC.


4. Log in to the Huawei HiLens intelligent edge system using a browser on the local PC. Enter the address of the Huawei HiLens IES in the address box. The address format is "**https://Huawei HiLens IES IP address**". The default IP address is **192.168.2.111**. Press **Enter**.

 **NOTE**

If "There is a problem with this website's security certificate" is displayed, click **Continue to this website (not recommended)**.

5. On the login page, enter login information.
Enter the username and password.
For details about the username and password, see .
6. On the main menu, choose **Maintenance > Firmware Update > System Firmware**.

The **System Firmware** page is displayed.

7. Click  on the right of **Update File**, and select a file, that is, the upgrade package downloaded in step 2.

The system displays a message indicating that the file has been added.

 **NOTE**

Do not close the current page or switch to another page during the upload process. Otherwise, the upload will fail.

8. Click **Upgrade**. In the confirmation box that is displayed, select **After the update is complete, the system restarts automatically**.

If you do not select this option, you need to manually restart the system for the update to take effect.

9. Click **OK**.
You can view the update package version and progress on the page.
10. Wait for about 10 minutes. The system displays a message indicating that the upgrade is successful.

If you perform the upgrade for the first time, log in to Huawei HiLens intelligent edge system again and **repeat** step 3 to step 8.

 **NOTE**

To ensure successful device registration, use the same upgrade package when performing the second upgrade.

If you have performed the upgrade twice, the HiLens Kit firmware has been upgraded.

5.2.3 Logging In to a HiLens Kit Device Using SSH

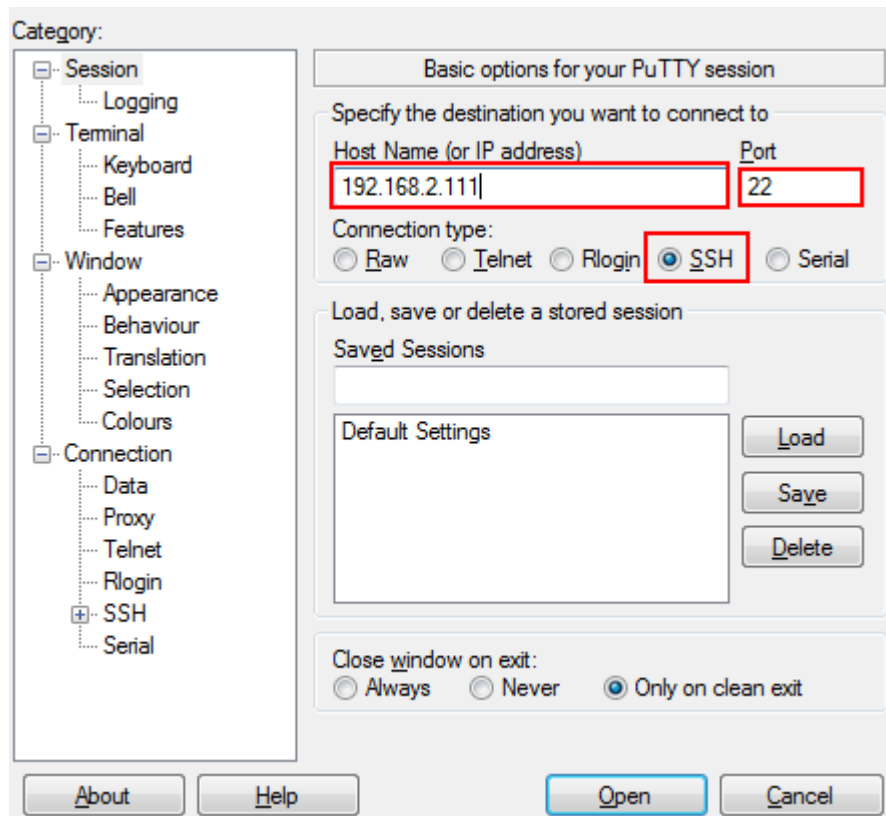
Prerequisites

You have connected a HiLens Kit device to a PC. For details, see [Connecting a HiLens Kit Device to a PC](#).

Procedure

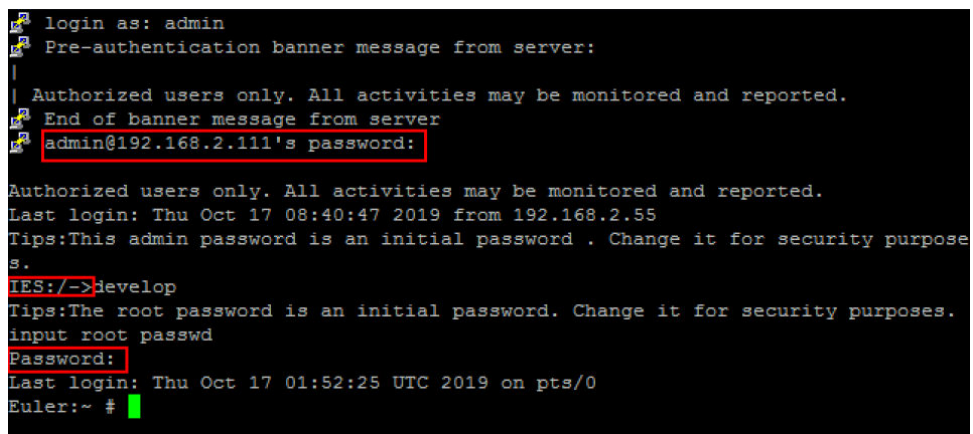
1. Use SSH to remotely connect to the HiLens Kit device.
 - a. Start PuTTY, click **Session**, enter the device IP address in the **Host Name (or IP address)** text box, and enter the port number in the **Port** text box. Assume that the device IP address is 192.168.2.111 and the port number is 22. [Figure 5-5](#) shows the page.

Figure 5-5 Using PuTTY to log in to the HiLens Kit device



- b. Click **Open**.
Log in to the device.
2. Use SSH to remotely connect to the device system.
After the **admin@192.168.2.111's password** prompt, enter the **admin** password. For details about the preset password for the first login, see .
3. Use SSH to remotely log in to the developer CLI.
 - a. After the **IES:/->** prompt, run the **develop** command.
 - b. After the **Password** prompt, enter the **root** password. For details about the preset password for the first login, see .

Figure 5-6 Connecting to the device using SSH



4. Use SSH to remotely modify the device time.
 - a. Run the following command to change the time zone to that of :
timedatectl set-timezone
 - b. Modify the device time. Assume that the current time is 19:19:19 on October 17, 2019. Run the following commands:
date -s "2019-10-17 19:19:19"
hwclock -w
reboot
Restarts the device.

NOTE

By default, a device is registered in region **CN North-Beijing4**. To register the device in **Singapore**, [switch the device registration region](#).

(Optional) Switching the Device Registration Region to Singapore

By default, a device is registered in region **CN North-Beijing4**. If you need to register the device in **Singapore**, modify the configuration file of the device. The procedure is as follows:

1. Run the following command on PuTTY to open the DNS file:
vi /home/hilens/hda/etc/hda.conf
2. Change the device registration region. Under the comment **#Select the registration region on Huawei Cloud, default to CN North-Beijing4**, change **region=cn-north-4** to:
region = ap-southeast-3
Save the modification.

NOTE

You must use a Huawei Cloud (International) account to register a device in Singapore. For details, see [Differences Between HUAWEI CLOUD Websites \(Chinese Mainland and International\)](#). If you do not have an account on Huawei Cloud (International), create one first. For details, see [Registering a HUAWEI ID and Enabling HUAWEI CLOUD Services](#).

3. Run the following command to restart the device:
reboot

Follow-up Operations

Network configuration: A HiLens Kit device can be connected to a router in wireless or wired mode. You can select either of the two networking modes.

The following sections describe how to use the two networking modes.

- [Wireless Network Configuration \(SSH\)](#)
- [Wired Network Configuration \(SSH\)](#)

A router cannot be connected to a wired network and a wireless network at the same time. If you connect the router to a wireless network, the default gateway will be automatically deleted. If you connect the router to a wired network again,

you need to configure the default gateway. **You are advised to connect the router to a wireless network, in case you forget the device IP address after updating it.**

5.2.4 Wireless Network Configuration (SSH)

A HiLens Kit device can be connected to a router in wireless or wired mode. This section describes the wireless network configuration mode.

NOTE

A router cannot be connected to a wired network and a wireless network at the same time. If you connect the router to a wireless network, the default gateway will be automatically deleted. If you connect the router to a wired network again, you need to configure the default gateway. **You are advised to connect the router to a wireless network, in case you forget the device IP address after updating it.**

If you select the wired network configuration mode, see [Wired Network Configuration \(SSH\)](#).

Configuration Requirements

To connect to a router through a wireless network, enter the network's password first.

Currently, HiLens Kit devices support only **2.4 GHz** wireless networks with general **encryption types**. The **name of a wireless network** consists of 8 to 63 characters but cannot contain single or double English quotation marks.

- The 2.4 GHz wireless networks comply with IEEE802.11n, IEEE802.11g, and IEEE802.11b protocols.
- The supported wireless network encryption types include WEP, WPA-PSK/WPA2-PSK, and AES.
- Wireless networks that need to be verified are not supported.
- TKIP encryption is not supported.
- The IP address of the router gateway cannot be set to **xxx.xxx.0.xxx**, for example, **192.168.0.1**. In addition, the router gateway and the HiLens Kit device cannot be in the same network segment. For example, if the device IP address is **192.168.2.111**, the IP address of the router gateway cannot be set to **192.168.2.xxx**.

Prerequisites

- Use a network cable to connect the PC and the device. For details, see [Connecting a HiLens Kit Device to a PC](#).
- Upgrade the HiLens Kit system firmware version. For details, see [Upgrading the HiLens Kit System Firmware Version](#).

Procedure

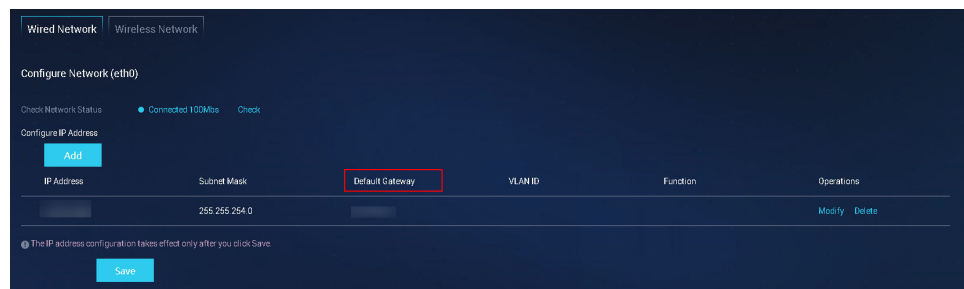
1. Log in to the Huawei HiLens intelligent edge system using a browser on the local PC. Enter the address of the Huawei HiLens IES in the address box. The address format is "**https://Huawei HiLens IES IP address**". The default IP address is **192.168.2.111**. Press **Enter**.

 **NOTE**

If "There is a problem with this website's security certificate" is displayed, click **Continue to this website (not recommended)**.

2. On the login page, enter login information.
Enter the username and password.
For details about the username and password, see .
3. Choose **Management > Network > Wired Network**.
The **Wired Network** page is displayed.
4. Under the **Configure IP Address** area, check whether the **IP Address** has a **Default Gateway**. See [Figure 5-7](#).

Figure 5-7 Checking the default gateway






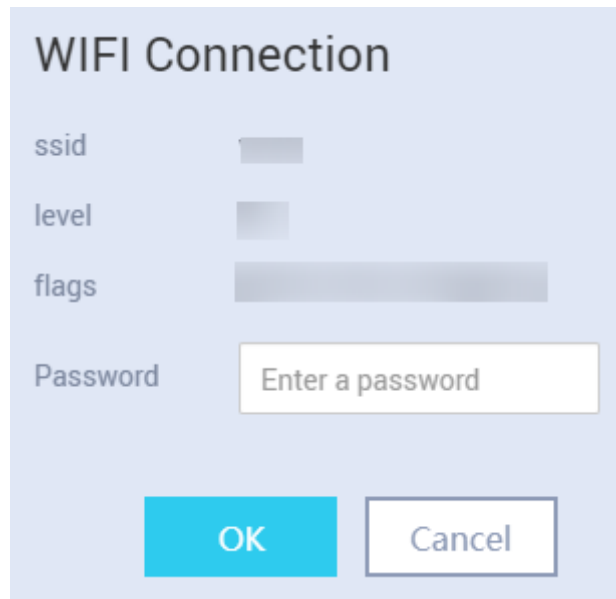
- If no default gateway exists, go to the next step.
 - If the default gateway exists, perform the following operations:
Click **Modify** in the **Operations** column. In the displayed **Modify IP Address** dialog box, delete the existing value in the **Default Gateway** text box and click **OK**.
Go to the next step.
5. Choose **Management > Network > Wireless Network**.
The **Wireless Network** tab page is displayed.
 6. Click  to enable the Wi-Fi.
The system searches for the nearby Wi-Fi hotspots.
 7. Click  to refresh the hotspot information.
 8. Click .
 - The **WIFI Connection** dialog box is displayed.
 9. Enter the Wi-Fi password in the **Password** text box, as shown in [Figure 5-8](#).
Click **OK**.
A message is displayed in the upper right corner, indicating that the wireless network is successfully connected to the router.

Figure 5-8 Connecting to a router through a wireless network



Checking the IP Address of a Wireless Network

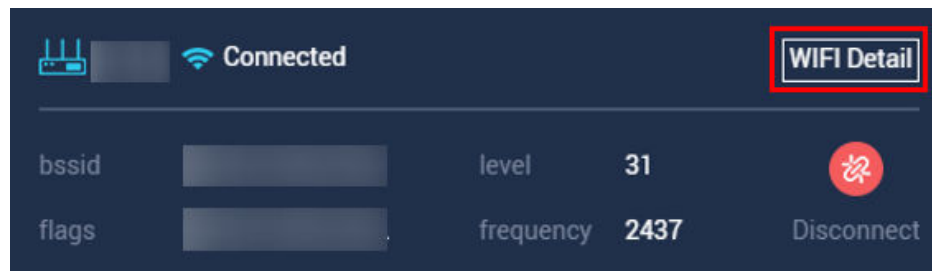
After a wireless network is configured, you can obtain its IP address from the Huawei HiLens intelligent edge system.

1. Log in to the Huawei HiLens intelligent edge system using a browser on the local PC. Enter the address of the Huawei HiLens IES in the address box. The address format is "**https://Huawei HiLens IES IP address**". The default IP address is **192.168.2.111**. Press **Enter**.

NOTE

If "There is a problem with this website's security certificate" is displayed, click **Continue to this website (not recommended)**.

2. On the login page, enter login information.
Enter the username and password.
For details about the username and password, see .
3. Choose **Management > Network > Wireless Network**.
The **Wireless Network** tab page is displayed.
4. In the upper right corner of the card of the wireless network you are connected to, click **WIFI Detail**.
The **WIFI detail** dialog box is displayed.
IP is the IP address of the wireless network.

Figure 5-9 Checking the IP address of a wireless network

Follow-up Operations

Register the HiLens Kit device. For details, see [Registering a HiLens Kit Device via SSH](#).

5.2.5 Wired Network Configuration (SSH)

A HiLens Kit device can be connected to a router in wireless or wired mode. This section describes the wired network configuration mode.

NOTE

A router cannot be connected to a wired network and a wireless network at the same time. If you connect the router to a wireless network, the default gateway will be automatically deleted. If you connect the router to a wired network again, you need to configure the default gateway. **You are advised to connect the router to a wireless network, in case you forget the device IP address after updating it.**

For details about the wireless network configuration, see [Wireless Network Configuration \(SSH\)](#).

Explanation

If a wired network is used to directly connect to a router, you need to modify the device IP address to ensure that the device IP address and the router IP address are in the same network segment.

The same network segment means that the first three octets of the device IP address are the same as that of the router IP address. For example, if the device IP address is 192.168.2.111, the router IP address can be 192.168.2.x, where x is an integer from 2 to 255 except 111.

Prerequisites

- Use a network cable to connect the PC and the device. For details, see [Connecting a HiLens Kit Device to a PC](#).
- Upgrade the HiLens Kit system firmware version. For details, see [Upgrading the HiLens Kit System Firmware Version](#).

Procedure

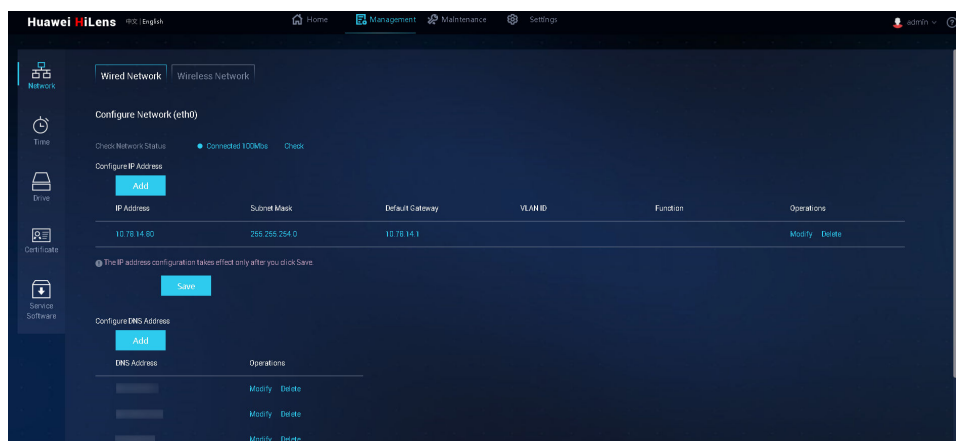
1. Log in to the Huawei HiLens intelligent edge system using a browser on the local PC. Enter the address of the Huawei HiLens IES in the address box. The address format is "[https://Huawei HiLens IES IP address](#)". The default IP address is **192.168.2.111**. Press **Enter**.

NOTE

If "There is a problem with this website's security certificate" is displayed, click **Continue to this website (not recommended)**.

2. On the login page, enter login information.
Enter the username and password.
For details about the username and password, see .
3. Choose **Management > Network > Wired Network**.
The **Wired Network** page is displayed.

Figure 5-10 Connecting to a router through a wired network



4. Click **Check** after **Check Network Status** to check whether the network is connected.
5. Change the IP address.
 - a. Click **Modify** in the row containing the IP address you want to change.
 - b. Change **IP Address** by referring to [Table 5-4](#). In this example, the router IP address is **192.168.2.1**, as shown in [Figure 5-11](#). After changing the IP address, click **OK**.
 - c. Click **Save** and restart the system for the IP settings to take effect.

Table 5-4 Parameters for changing an IP address

Parameter	Description
Function	Usage of the IP address. The value contains 1 to 32 characters, including letters, digits, and underscores (_).
IP Address	IPv4 address to be changed. The IP address must be in the same network segment as the router IP address.
Default Gateway	Default gateway corresponding to the new IP address. NOTE Ensure that the default gateway is globally unique.

Figure 5-11 Changing the IP address

Modify IP Address

Function: ZhiLian

IP Address: 192 · 168 · 2 · 111

Subnet Mask: 255 · 255 · 255 · 0

Default Gateway: 192 · 168 · 2 · 1

VLAN ID:

If the network configuration contains IP addresses of the same network segment, please ensure that the configured IP address is correct and does not conflict with other IP addresses on the network, otherwise network exceptions will be caused.

OK Cancel

6. Remove the network cable from the PC, disconnect the device from the PC, and then connect the device to the router using the network cable.
7. After the connection is complete, log in to the device using SSH. Run the following command to check whether the connection is successful:

ping 8.8.8.8

or

ping www.huaweicloud.com

If the device is connected successfully, the following information is displayed:

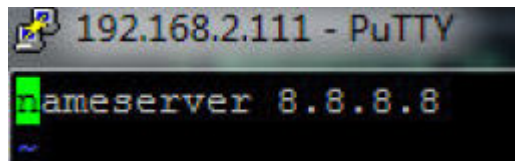
Figure 5-12 Wired connection prompt

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=50 time=86.5 ms  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=50 time=172 ms  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=50 time=125 ms  
^C  
--- 8.8.8.8 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2003ms  
rtt min/avg/max/mdev = 86.476/127.914/171.826/34.887 ms
```

If the device connection fails, perform the next step.

8. Remotely configure DNS using SSH.
 - a. Run the following command on PuTTY to open the DNS file:
vi /etc/resolv.conf
 - b. Delete the original content and change the parameter to the following:
nameserver 8.8.8.8
 - c. Save the settings and exit. See [Figure 5-13](#). Repeat the previous step to check whether the connection is successful.

Figure 5-13 Configuring DNS



Follow-up Operations

Register the HiLens Kit device. For details, see [Registering a HiLens Kit Device via SSH](#).

5.2.6 Registering a HiLens Kit Device via SSH

Prerequisites

- Use a network cable to connect the PC and the device. For details, see [Connecting a HiLens Kit Device to a PC](#).
- Upgrade the HiLens Kit system firmware version. For details, see [Upgrading the HiLens Kit System Firmware Version](#).
- Select either of the following network configuration modes:
 - [Wireless Network Configuration \(SSH\)](#)
 - [Wired Network Configuration \(SSH\)](#)

A router cannot be connected to a wired network and a wireless network at the same time. If you connect the router to a wireless network, the default gateway will be automatically deleted. If you connect the router to a wired network again, you need to configure the default gateway. **You are advised to connect the router to a wireless network, in case you forget the device IP address after updating it.**

Procedure

1. Run the following command to remotely register the HiLens Kit device with the Huawei HiLens console using SSH:

```
hdactl register -uUsername -dAccountName -nDeviceName
```

Press **Enter**. The **Password** prompt is displayed.

Username is the IAM username, and *Account name* is your HUAWEI CLOUD account name. For more information, see [Basic Concepts](#) of IAM. If no IAM account is available, the *Account name* is the same as the *Username*.

 NOTE

- The account you use to register a HiLens Kit device **must be a HUAWEI CLOUD account, rather than a HUAWEI ID.**

HUAWEI ID and HUAWEI CLOUD account are different. If you do not have a HUAWEI ID, you are advised to register one and enable HUAWEI CLOUD services. For details, see [Registering a HUAWEI ID and Enabling HUAWEI CLOUD Services](#).

If your account is a HUAWEI ID, use this account to create a sub-account. For details, see [Creating an IAM User](#), and add the admin permissions to the sub-account.

- When registering a device in Singapore, use the account name and username of your HUAWEI CLOUD (International) account. For details, see [Differences Between HUAWEI CLOUD Websites \(Chinese Mainland and International\)](#). If you do not have an account on HUAWEI CLOUD (International), create one first. For details, see [Registering a HUAWEI ID and Enabling HUAWEI CLOUD Services](#).

Change the **Device name** by yourself.

The username, account name, and device name used for registering a HiLens Kit device can contain only letters, digits, and underscores (_). The names cannot start with a digit or contain digits only.

2. Enter the password of your Huawei Cloud account at the **password** prompt and press **Enter** to complete the registration.

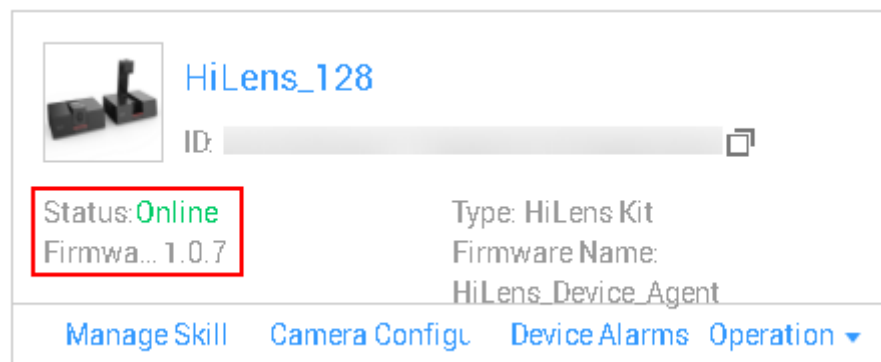
If the IAM username and Huawei Cloud account name are different, enter the IAM user password following the **Password** prompt.

If an error message indicating that the device cannot be registered, [restore the device to factory settings](#) and register it again.

 NOTE

- IAM users are not allowed to register devices with sub-projects. For detailed concepts, see [IAM Basic Concepts](#).
 - After the device is registered, you can log in to the Huawei HiLens console and choose **Device Management > Devices** to view the device status. Your device is offline for a short period of time.
3. Log in to the Huawei HiLens console. In the navigation pane, choose **Device Management > Devices** to view the device status, as shown in [Figure 5-14](#).

Figure 5-14 Device status



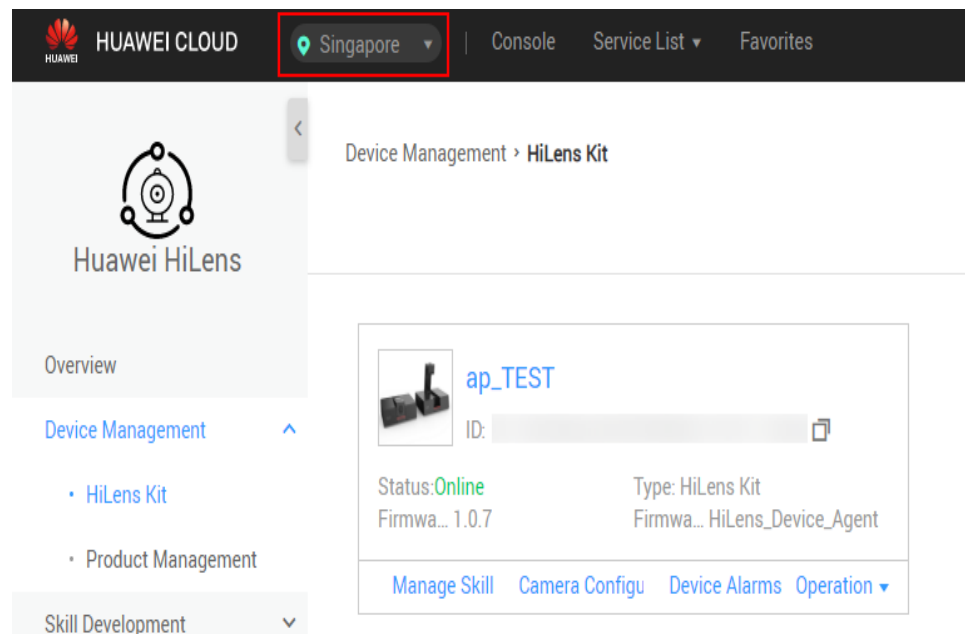
- a. Wait 3 minutes. If the device is online, the device has restarted successfully.
- b. Wait for 3 minutes. If the device is still offline, check whether the device time is consistent with the current time. For details, see **Device Is Offline**.

Viewing the Registered Device on the Huawei HiLens Console

Log in to the Huawei HiLens console. Switch the region to **Singapore** in the upper left corner of the console. In the navigation pane, choose **Device Management > Devices**. All registered devices are displayed in the device list by default. If you can view the device you registered and its status is **Online**, the device registration is successful.

If the registration fails or the device is offline, rectify the fault by following the instructions provided in [HiLens Kit Registration Failed](#).

Figure 5-15 Device registration status



NOTICE

When you check the device registration status:

- After registering the device, upgrade the HiLens_Device_Agent firmware to version 1.0.7 or later. For details, see [Upgrading HiLens_Device_Agent Firmware](#).

5.3 Configure Networking

5.3.1 Networking Configuration Modes

HiLens Kit devices can be connected to routers in wired or wireless mode. You can select either mode to configure the network, or use both wired and wireless

modes to connect to an external network and the internal network of a company or school.

- Scenario 1: Use a wireless network.
[Wireless Network Configuration \(SSH\)](#)
- Scenario 2: Use a wired network.
[Wired Network Configuration \(SSH\)](#)
- Scenario 3: Use a wireless network to connect to an external network and a wired network to connect to the internal network of a company or school.
[Wireless Network \(External Network\) and Wired Network \(Internal Network\)](#)

 NOTE

You are advised not to use a wireless network and a wired network to connect to the same router at the same time. You are advised to use a wireless network to avoid forgetting or losing the device IP address after updating it.

5.3.2 Wireless Network (External Network) and Wired Network (Internal Network)

This section describes how to use a wireless network (Wi-Fi) to connect to an external network and use a wired network to connect to the internal network of a company or school.

Method 1: Configure the wireless network, and then configure the wired network. Add the deleted default routing entries of the wireless network IP address.

Method 2: Configure the wired network, and then configure the wireless network. Add the deleted default routing entries of the wired network IP address.

This section uses method 1 to describe how to configure networking using both wireless and wired modes for a HiLens Kit device.

Preparations

1. Use a network cable to connect the HiLens Kit device to the local PC.
2. Enter **https://XXX.XXX.XXX.XXX** in the address box of a browser and press **Enter** to open the login page of the HiLens intelligent edge system.
XXX.XXX.XXX.XXX is the default IP address of the device. For details about the default parameters, see [Default Parameters](#).
3. Enter the username and password. For details about the default parameters, see [Default Parameters](#).
4. Use the default IP address to log in to the HiLens Kit device through SSH and go to the CLI with the administrator permission (**develop**). For details, see [Logging In to a HiLens Kit Device Using SSH](#).
If the IP address assigned to the internal network of a company or school needs to be bound to the MAC address of the connected HiLens Kit device, fix the MAC address of the wired network port (eth0) on the device.
 - a. Run the **ifconfig** command on the CLI with the administrator (develop) permissions to view the MAC address of the connected port on the HiLens Kit device.

As shown in [Figure 5-16](#), the content in the red box is the target MAC address.

Figure 5-16 Viewing the MAC address

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.111 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::aeb3:b5ff:fe2e:8763 prefixlen 64 scopeid 0x20<link>
    ether ac:b3:b5:2e:87:63 txqueuelen 1000 (Ethernet)
    RX packets 452 bytes 44420 (43.3 KiB)
    RX errors 0 dropped 4 overruns 0 frame 0
    TX packets 524 bytes 36961 (36.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 40
```

- b. On the CLI, open the `vi /etc/rc.d/rc.local` file and add the following content to the file:

```
Ifconfig eth0 down
```

```
Ifconfig eth0 hw ether xx:xx:xx:xx:xx:xx (xx:xx:xx:xx:xx:xx indicates the MAC address.)
```

```
Ifconfig eth0 up
```

- c. Restart the device for the modification to take effect.

Step 1: Configuring the Wireless Network

Use the wireless network to connect to the external network and dynamically obtain the IP address of the wireless network. To connect to a router through a wireless network, enter the network's password first.

- **Procedure**

For details about the requirements and procedure for configuring the wireless network, see [Wireless Network Configuration \(SSH\)](#).

- **Network Verification**

- a. On the CLI, run the `ifconfig` command to view the IP address of the HiLens Kit device.

For example, in the following figure, the IP address of the wireless network is **192.168.137.26**, and `wlan0` is the wireless network port.

Figure 5-17 Device IP address

```
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.137.26 netmask 255.255.255.0 broadcast 192.168.137.255
    inet6 fe80::20f:ff:fe2f:c2df prefixlen 64 scopeid 0x20<link>
    ether 00:0f:00:c2:df txqueuelen 1000 (Ethernet)
    RX packets 260 bytes 30626 (29.9 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 157 bytes 18923 (18.4 KiB)
```

- b. Run the `route` command to check routing entries.

Assume that the wireless network IP address is **192.168.137.26** and the allocated default gateway is **192.168.137.1**.

Figure 5-18 Routing entry information-1

```
Euler:~ # route
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
default          192.168.137.1  0.0.0.0        UG    0     0     0 wlan0
link-local       0.0.0.0        255.255.0.0    U     1002  0     0 eth0
172.17.0.0       0.0.0.0        255.255.0.0    U     0     0     0 docker0
192.168.2.0      0.0.0.0        255.255.255.0  U     0     0     0 eth0
192.168.137.0   0.0.0.0        255.255.255.0  U     0     0     0 wlan0
Euler:~ #
```

- c. Run the following command to check whether the wireless network is successfully connected:

ping 8.8.8.8

If the device is connected successfully, the following information is displayed:

Figure 5-19 Wireless connection prompt-1

```
PING www. (103.235.46.39) 56(84) bytes of data:
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=3 ttl=43 time=80.3 ms
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=13 ttl=43 time=528 ms
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=15 ttl=43 time=199 ms
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=16 ttl=43 time=86.9 ms
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=17 ttl=43 time=166 ms
```

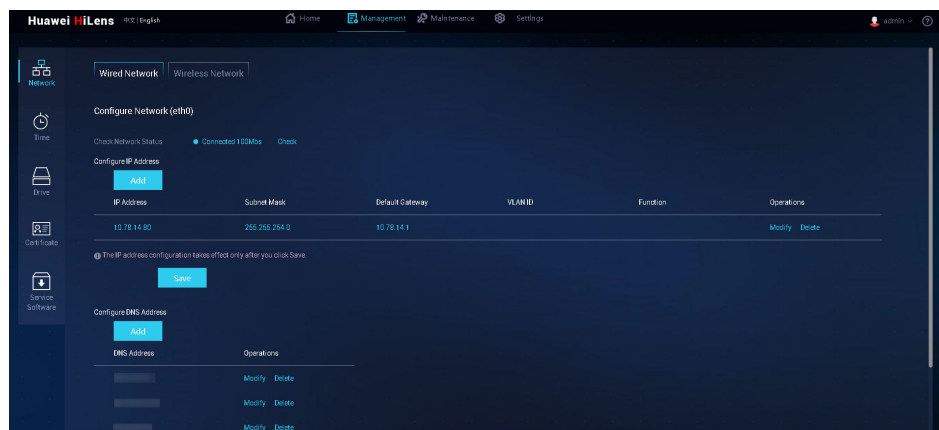
Step 2: Configuring the Wired Network

- **Procedure**

If you use a wired network to connect to the internal network of a company or school, you need to add an internal network IP address on the wired network port. If domain name resolution is required, configure the corresponding DNS.

- Enter **https://XXX.XXX.XXX.XXX** in the address box of a browser and press **Enter** to open the login page of the HiLens intelligent edge system. **XXX.XXX.XXX.XXX** is the default IP address of the device. For details about the default parameters, see [Default Parameters](#).
- Enter the username and password. For details about the default parameters, see [Default Parameters](#).
- Choose **Management > Network > Wired Network**.
The **Wired Network** page is displayed.

Figure 5-20 Wired network configuration



- d. Under **Configure IP Address**, click **Add**. In the displayed dialog box, enter the internal IP address information according to [Table 5-5](#) and click **OK**.
- e. Under **Configure IP Address**, click **Save**.

Table 5-5 New IP address parameters

Parameter	Description
Function	Usage of the IP address. The value contains 1 to 32 characters, including letters, digits, and underscores (_).
IP Address	IPv4 address to be changed. The IP address must be in the same network segment as the router IP address.
Default Gateway	Default gateway corresponding to the new IP address. NOTE Ensure that the default gateway is globally unique.

Figure 5-21 Adding an IP address

Add IP Address

Function: wired_network

IP Address: 10 · 78 · 14 · 58

Subnet Mask: 255 · 255 · 254 · 0

Default Gateway: 10 · 78 · 14 · 1

VLAN ID:

If the network configuration contains IP addresses of the same network segment, please ensure that the configured IP address is correct and does not conflict with other IP addresses on the network, otherwise network exceptions will be caused.

OK Cancel

- f. Configure the DNS address for the internal network. The DNS configuration method varies depending on whether the HiLens Kit system firmware has been upgraded.

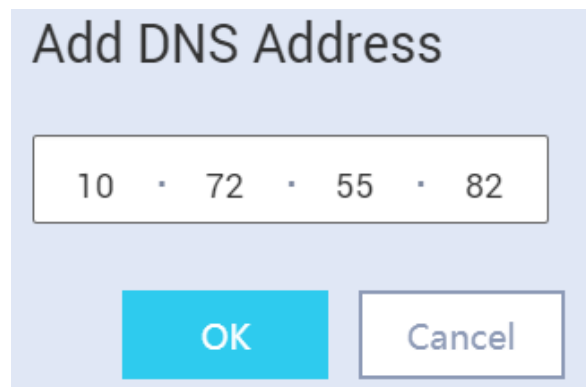
- **The HiLens Kit system firmware has been upgraded.**

The HiLens Kit system firmware version has been upgraded to 2.2.200.011. For details, see [Upgrading the HiLens Kit System Firmware Version](#). You can configure DNS as follows:

- 1) On the **Management > Network > Wired Network** page of the intelligent edge system, click **Add** under **Configure DNS Address**.

The **Add DNS Address** dialog box is displayed.

Figure 5-22 Adding DNS address



- 2) In the dialog box that is displayed, enter the DNS address of the internal network of a company or school, and click **OK**.
- 3) Under **Configure DNS Address**, click **Save**.

- **The HiLens Kit system firmware has not been upgraded.**

Remotely configure DNS using SSH.

Open the `vi /etc/resolv.conf` file in PuTTY and add the DNS address.
nameserver x.x.x.x (*x.x.x.x* indicates the newly added DNS address.)

Save the file and exit.

- **Network Verification**

- a. Use the default IP address to log in to the HiLens Kit device using SSH. For details, see [Logging In to a HiLens Kit Device Using SSH](#).
- b. Run the `ifconfig` command to view the IP address of the device. For example, in the following figure, the IP address of the wired network is **10.78.14.58**, **eth0** is the wired network port, and **eth0:1** is the newly added IP address.

Figure 5-23 Device IP address

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.111 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::4f5:81ff:fe49:e92b prefixlen 64 scopeid 0x20<link>
    ether 06:f5:81:49:e9:2b txqueuelen 1000 (Ethernet)
    RX packets 5864 bytes 1320409 (1.2 MiB)
    RX errors 0 dropped 2 overruns 0 frame 0
    TX packets 4706 bytes 16730604 (15.9 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 40

eth0:1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.78.14.58 netmask 255.255.254.0 broadcast 10.78.15.255
    ether 06:f5:81:49:e9:2b txqueuelen 1000 (Ethernet)
    device interrupt 40
```

- c. Run the **route** command to check routing entries.

For example, if the IP address of the wired network is **10.78.14.58** and the allocated default gateway is **192.168.137.1**, the default routing information corresponding to the IP address allocated in **Step 1: Configuring the Wireless Network** is missing. As a result, the external network cannot be accessed.

Figure 5-24 Routing entry information

```
Euler:~ # route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
default 10.78.14.1 0.0.0.0 UG 0 0 0 eth0
10.78.14.0 0.0.0.0 255.255.254.0 U 0 0 0 eth0
link-local 0.0.0.0 255.255.0.0 U 1002 0 0 eth0
172.17.0.0 0.0.0.0 255.255.0.0 U 0 0 0 docker0
192.168.2.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
192.168.137.0 0.0.0.0 255.255.255.0 U 0 0 0 wlan0
```

Step 3: Adding Routing Entries

Manually add the default routing entries corresponding to the IP address allocated in **Step 1: Configuring the Wireless Network**.

- **Procedure**

- a. Use the default IP address to log in to the HiLens Kit device through SSH and go to the CLI with the administrator permission (**develop**). For details, see **Logging In to a HiLens Kit Device Using SSH**.
- b. Run the corresponding command on the CLI to add the default routing entries.

For example, the default gateway corresponding to the IP address allocated in **Step 1: Configuring the Wireless Network** is **192.168.137.1**.

```
route add default gw 192.168.137.1
```

- c. Remove the network cable from the PC and disconnect the network between the PC and HiLens Kit device. Then, use the network cable to connect the HiLens Kit device to the internal network.

- **Network Verification**

- a. On the PC in the internal network, use the new internal network IP address (**10.78.14.58** in this example) to log in to the HiLens Kit device through SSH. For details, see **Logging In to a HiLens Kit Device Using SSH**.

- b. Run the **ping** command to check whether the wired network (external network) and wireless network (internal network) are successfully connected.

If the device is connected successfully, the following information is displayed:

Figure 5-25 Wired connection prompt

```
Euler:~ # ping 10.78.15.147
PING 10.78.15.147 (10.78.15.147) 56(84) bytes of data.
64 bytes from 10.78.15.147: icmp_seq=1 ttl=127 time=0.897 ms
64 bytes from 10.78.15.147: icmp_seq=2 ttl=127 time=0.891 ms
64 bytes from 10.78.15.147: icmp_seq=3 ttl=127 time=0.907 ms
```

Figure 5-26 Wireless connection prompt

```
Euler:~ # ping www. .com
PING www.wshifen.com (104.193.88.77) 56(84) bytes of data.
64 bytes from 104.193.88.77 (104.193.88.77): icmp_seq=1 ttl=44 time=220 ms
64 bytes from 104.193.88.77 (104.193.88.77): icmp_seq=2 ttl=44 time=224 ms
```

5.4 Configuring a Firewall

To prevent users beyond the allowed IP address range from accessing the HiLens Kit intelligent edge system, you can remotely configure a firewall through SSH to prevent the system from being attacked.

Configuration Mode

- Configure a hardware firewall.
If a hardware firewall exists, contact O&M personnel to configure it to allow access to the IP addresses of HiLens Kit devices.
- Configure a software firewall.
If no hardware firewall is deployed, configure a software firewall. For details, see [Configuring a Software Firewall](#).

Configuring a Software Firewall

1. Connect your HiLens Kit device to your PC using a network cable and log in to the device in SSH mode. For detailed operations, see [Connecting a HiLens Kit Device to a PC](#).
2. Run the following command on PuTTY:
iptables -I INPUT -p tcp --dport 443 -j DROP
Disable all users to access the HiLens Kit intelligent edge system through port 443.
3. Configure the IP addresses that are allowed to access the HiLens Kit intelligent edge system through port 443 as required.
 - Run the following command on PuTTY to configure the IP address of a HiLens Kit device to access the HiLens Kit intelligent edge system (assume that the device IP address is 192.168.2.111):
iptables -I INPUT -s 192.168.2.111 -p tcp --dport 443 -j ACCEPT

- Run the following command on PuTTY to configure one IP address to access the HiLens Kit intelligent edge system (assume that the allowed IP address is 10.61.120.127):
iptables -I INPUT -s 10.61.120.127 -p tcp --dport 443 -j ACCEPT
 - Run the following command on PuTTY to configure a network segment to access the HiLens Kit intelligent edge system (assume that the network segment is 10.61.120.*, where * is an integer from 0 to 255):
iptables -I INPUT -s 10.61.120.0/24 -p tcp --dport 443 -j ACCEPT
4. Go to the **/home/data/user** directory on PuTTY and write the configuration commands in step 2 and step 3 to the **user_init.sh** file to prevent the configurations in step 2 and step 3 from becoming invalid after the HiLens Kit device is restarted.
- If the **user_init.sh** file does not exist in the **/home/data/user** directory, create one.

5.5 Using an SD Card

HiLens Kit is multi-modal AI development suite featuring device-cloud synergy. It generates video and image data during use. Generally, data is stored in OBS through APIs. To store data locally, an external SD card is required because the disk space of HiLens Kit devices is limited.

Procedure

1. Insert an SD card into the port (Micro SD) on the rear panel of the HiLens Kit device.
2. Log in to the HiLens Kit device using SSH. For details, see [Logging In to a HiLens Kit Device Using SSH](#).
3. Run the following command to locate the SD card:
ls -al /dev/mmcblk*
The command output is as follows:
dev/mmcblk1
4. Run the following command to save and back up the data stored in the SD card and format the SD card.
mkfs.ext4 /dev/mmcblk1
Run the command at the EulerOS prompt.
If the SD card has been formatted, skip this step.
5. Run the following command to create the **sd** path:
mkdir /mnt/sd
The device is mounted to the **mount /dev/ mmcblk1/mnt/sd** path.
6. Run the **df** command to query the file system.
Run the following commands to perform a read/write test on the local file **mount /dev/ mmcblk1/mnt/sd**:
 - Write command: **cp test.txt /mnt/sd**
 - Read command: **cp /mnt/sd/test.txt ./**

 **NOTE**

Before removing the SD card, run the following command to unmount the partition from the device:

umount /mnt/sd

This prevents files from being damaged.

6 Managing Devices

6.1 Device Management Overview

The platform supports registration, management, and deregistration of HiLens Kit devices and Atlas 500 AI edge stations. Manufacturers can use the Huawei HiLens console to manage devices that run HiSilicon Hi35xx series chips in batches and deploy AI capabilities onto them. For details, see [Overview](#). The following operations apply only to HiLens Kit cameras.

Before managing the devices, you need to register devices on the Huawei HiLens console. For detailed descriptions and connection guides, see [Registering HiLens Kit Devices](#).

Device Management Operations

- **Managing Skills on Devices:** Install skills, configure runtime parameters, and start, stop, or uninstall skills on registered devices.
- **Viewing Device Information:** Check the basic information or upgrade status of a device.
- **Viewing Device Logs:** After using a HiLens skill, you can view the device agent logs and skill logs.
- **Upgrading HiLens_Device_Agent Firmware:** Upgrade the firmware version when a new version is available.
- **Configuring Cameras:** Connect more IP cameras to a registered device and manage them.
- **Deregistering Devices:** Deregister a device to release resources when the device is no longer used.
- **Viewing Device Alarms:** You can check the device's alarm information on the Huawei HiLens console and handle the alarm on the device.
- **Viewing Device Data:** In the navigation pane of the Huawei HiLens console, click **Data Management**. On the **Data Management** page that is displayed, you can select a device to download and view all of its data.

6.2 Managing Skills on Devices

6.2.1 Skill Management Overview

For registered devices in the **Online** state, you can manage their skills on the Huawei HiLens platform, including installing new skills, adding runtime configurations, and starting, stopping, and uninstalling skills.

Skill

A skill is an AI application ready to run on a camera or another computing device. It consists of a model and logic code. The logic code is the skill's framework that governs how the skill behaves, including data reading, model import, model inference, and result output. The model is an AI algorithm trained using big data and handles inference while the skill is running.

The **Skill Market** of Huawei HiLens provides extensive skills for users.

- The application scenarios for HiLens skills include smart campus, smart home, smart vehicle-mounted terminals, smart shopping mall, and more.
- Skills are classified into two types by device: skills for Ascend chips and skills for HiSilicon Hi35xx series chips.

Figure 6-1 Skill card



Ascend310

Funny_Face_Effe...

Add effects to your face. This skill will adds different items to your face which depends

Before You Start

- Before managing skills, upgrade the HiLens_Device_Agent firmware to version 1.0.7 or later. For details, see [Upgrading HiLens_Device_Agent Firmware](#).
- A maximum of five skills can be installed on a device.
- No skill information is displayed on the skill management page of a newly registered device. You need to [install new skills](#) before using the device.
- The HiLens console can manage only HiLens Kit devices. Therefore, only skills oriented to Ascend chips can be managed here.

- You can purchase and install skills on devices from the [Skill Market](#).
- You can also develop skills on the [HiLens console](#).
- In the device list, click the device card to enter the device details page. The **Skill Management** page is displayed by default, on which you can view the skills installed on the device. The skill statuses include **Installing**, **Faulty**, **Stopped**, and **Running**.

Supported Operations

Go to the Skill Management page: In the navigation pane of the HiLens console, choose **Device Management > Devices**. In the device list, select the device to be managed and click the device card. The device details page displays the **Skill Management** tab by default.

On the **Skill Management** tab, you can view the information about skills installed on the device. If the skill is in container image format, you can click \vee on the left of the skill name to view the status of each instance of the skill.

- [Installing a Skill](#)
- [Adding Runtime Configurations](#)
- [Starting or Stopping Skills](#)
- [Uninstalling Skills](#)
- **View skill run logs on the device:** On the **Skill Management** tab page, choose **More > View Log** in the row of the target skill to go to the **Data Management** page and view the real-time or historical running data of the skill. For details, see [Managing Data](#).
- **Configure a subscription message for a skill:** On the **Skill Management** tab, choose **More > View Skill Message** to switch to the **Skill Messages** tab, where you can directly configure a subscription object for the skill. For details, see [Subscribing to Skill Messages](#).

6.2.2 Installing a Skill

You can purchase required skills from the skill market and install them onto your devices to enable the devices with AI capabilities.

Prerequisites

- At least one device exists on the **Device Management > Devices** page and the device status is **Online**.
- At least one skill exists on the **Subscribe > Order Management > Skill Orders** or **AI Gallery** page. If you have not purchased any skills, no information is displayed on this page. You are advised to purchase skills by referring to [Purchasing Skills](#).

Procedure

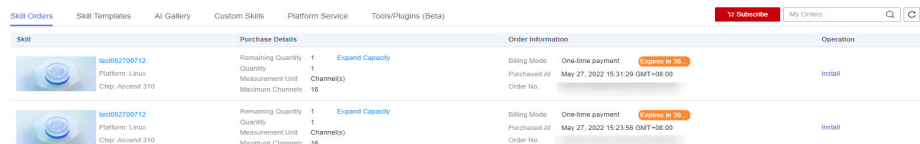
1. In the navigation pane of the HiLens console, choose **Device Management > Devices**. In the device list, click the target device card.
The **Skill Management** page is displayed by default.

Figure 6-2 Skill Management



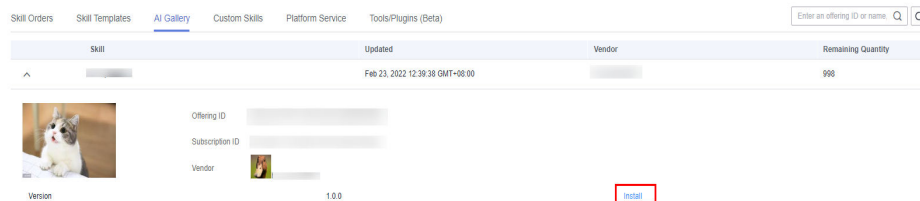
2. On the **Skill Management** page, click **Add Skill** in the upper right corner to switch to the **Subscribe > Order Management** page.
3. By default, the **Skill Orders** page is displayed, showing the list of skills purchased by your account from the HiLens Skill Market. You can also click **AI Gallery** to display the list of skills purchased by your account from the ModelArts AI Gallery.
4. Select the skill to be installed.
 - On the **Skill Orders** page: Click **Install** in the **Operation** column of the skill list.

Figure 6-3 Skill Orders-15



- On the **AI Gallery** page: Click **▼** on the left of the target skill name to view the skill information, and click **Install**.

Figure 6-4 Skill installation



NOTE

Only skills (with sufficient quota) **oriented to Ascend 310 chips** can be selected.

5. In the skill installation dialog box that is displayed, select the device to which the skill is to be installed. In the **Specifications** column, specify the number of channels used by the skill on each device. If you want to install the skill on multiple devices, select **Yes** in the **Batch Installation** column as required. Then, click **Install**.
6. Perform the following operations to check whether the skill is installed properly. If the skill status is **Installation succeeded**, the skill is successfully installed on each device.
 - a. In the navigation pane of the Huawei HiLens console, choose **Device Management > Devices**. The device list page is displayed.
 - b. Click **Skill Management** of the registered device. The status of the installed skill is **Stopped**. Click **Start** in the **Operation** column and click **OK** to start the skill on the device.

The skill status changes to **Running** after a short period of time, indicating that the skill is successfully running on the device.

Table 6-1 Skill installation statuses

Status	Description
Stopped	The skill stops running on the device.
Running	The skill is running on the device.


6.2.3 Adding Runtime Configurations

You may need to configure related parameters for some running skills. For example, upload a facial image library to the facial recognition skill.

Context

- You can add and modify configurations of skills whose developers have configured the runtime configuration function for them.
- **Runtime Configurations** are classified into **Global** and **Video** configurations.
 - The **Global** configuration takes effect for the entire device.
 - The **Video** configuration indicates a camera-channel configuration, requiring the skill to process multi-channel videos. For example, if a facial image library is configured for each camera, the facial recognition skill can perform recognition on multiple channels of videos.

Adding Runtime Configurations

1. In the navigation pane of the Huawei HiLens console, choose **Device Management > Devices**. In the device list, click the target device card. The **Skill Management** page is displayed by default.
2. On the **Skill Management** page, click the target skill card. The skill details page is displayed.
3. On the **Skill Management** page, click **Runtime Configurations** in the **Operation** column.
4. In the dialog box that is displayed, set the runtime configuration parameters. You can click  next to each parameter to view its details.

The parameters are classified into **Global** and **Video** configurations. The **Global** configuration takes effect for the entire device. The **Video** configuration indicates a camera-channel configuration, requiring the skill to process multi-channel videos. For example, if a facial image library is configured for each camera, the facial recognition skill can perform recognition on multiple channels of videos.

6.2.4 Subscribing to Skill Messages

After you subscribe to messages for a specified skill, the subscription object (recipient) will receive an email or SMS notification when the skill has any output.

For example, assume that a skill has the function of detecting strangers and sending messages, that is, the skill's **Product Description** on the skill details page contains "This skill supports notification messages". Then, after you install the skill and subscribe to a message topic (**Configuring Subscription Objects**), the configured subscription object will receive an email or SMS notification when the device detects a stranger.

Precautions

- When subscribing to a message for the first time, restart the skill so that the subscription object can receive the message. For details about how to restart the skill, see **Starting or Stopping Skills**.

Prerequisites

- The skill supports the message notification function. This function is configured during skill development. For details about the guide and APIs, see the HiLens Developer Guide.

If the message notification function is not configured for a skill, common users cannot receive notification messages from the skill.

- The skill has been deployed on a device and is in the **Running** state. Users cannot receive notification messages from a skill not in the **Running** state.

Configuring Subscription Objects

A subscription object is the recipient of notification messages. When the device detects a specified scenario (for example, a stranger detection skill installed on the device detects a stranger), the subscription object can receive a notification message.

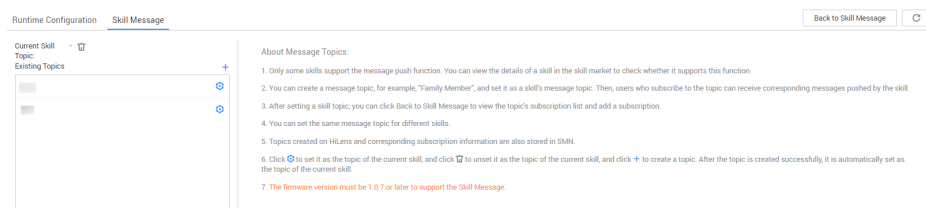
1. Log in to the Huawei HiLens console, choose **Device Management > Devices** in the navigation pane. The device list page is displayed.
2. Click the card of the device where the skill is installed. The **Skill Management** page is displayed by default.

Figure 6-5 Accessing the device details page



3. Click the name of the skill for which you want to configure a subscription object. The skill details page is displayed.
4. Click the **Skill Messages** tab.

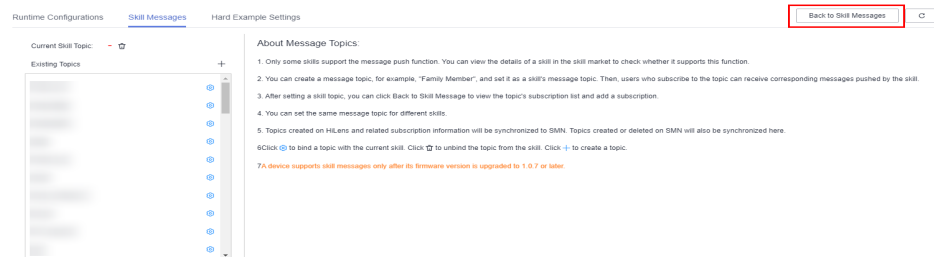
Figure 6-6 Skill message



5. Set the **Current Skill Topic**.
 - To add a new topic:
 - i. Click **+** on the right of **Existing Topics**. In the text box that is displayed, enter a topic name, which contains a maximum of 255 characters, starting with a letter or digit. Only letters, digits, underscores (**_**), and hyphens (**-**) are allowed. Then, click **✓**.
 - ii. Click **⊗** on the right of the topic. In the dialog box that is displayed, click **OK** to confirm the skill topic setting.
 - To add an existing topic:

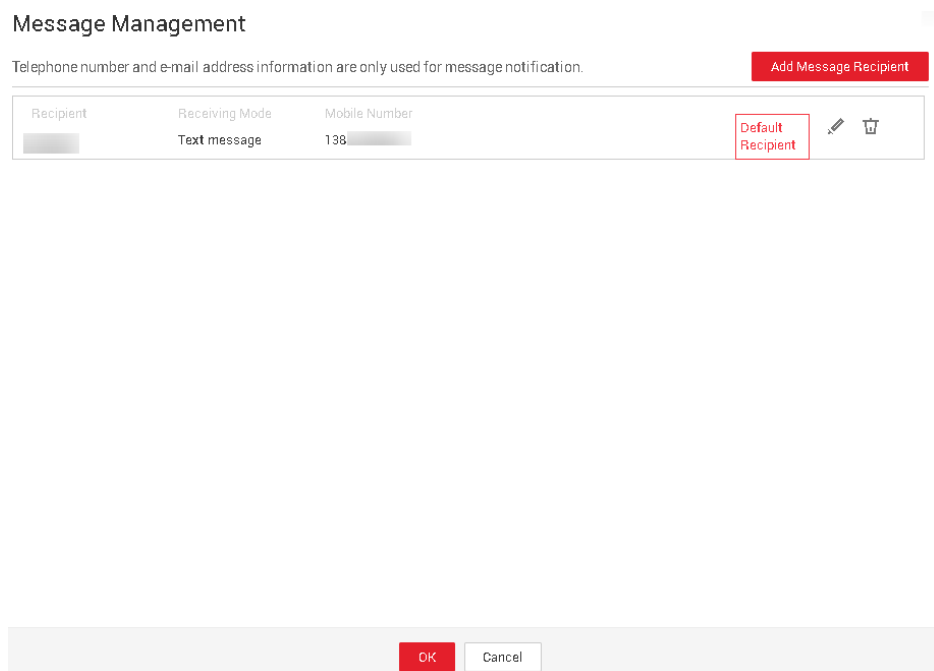
Click **⊗** on the right of an existing topic. In the dialog box that is displayed, click **OK** to confirm the skill topic setting.
6. Click **Back to Skill Messages** in the upper right corner of the **Skill Messages** tab to configure the message recipient of the current skill topic.

Figure 6-7 Back to Skill Messages



7. Click **+** on the right of the subscription list. The **Message Management** dialog box is displayed.

Figure 6-8 Message recipients



8. Edit the message recipients of the current topic as follows:

- To add a new recipient:
Click **Add Message Recipient** and enter recipient information in the text boxes that are displayed. For details about the parameters, see [Table 6-2](#). Confirm the information and click ✓.
- To edit an existing recipient:
Click ✎ on the right of a message recipient and re-enter the recipient information in the text boxes. For details about the parameters, see [Table 6-2](#). Confirm the information and click ✓.

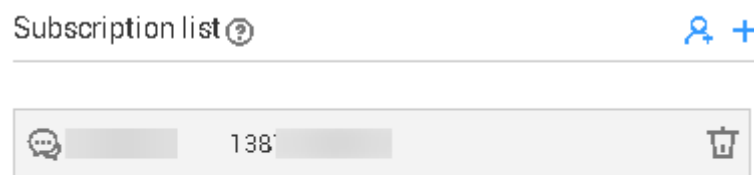
Table 6-2 Message recipient parameters

Parameter	Description
Recipient	Name of the subscription object
Receiving Mode	Message receiving mode. The options are Email and Text message .
Mobile Number/ Email Address	Mobile number or email address used for receiving messages

You can click **Set to Default** to set the default recipient for quick subscription. (You can click ⚙ on the right of the subscription list in step 4 to perform quick subscription.)

9. Click to select the recipient who subscribes to the skill, and click **OK**.
The subscription list is displayed. You can view the added recipient.
To add multiple message recipients, go to step 7.

Figure 6-9 Subscription list



10. After the message recipient is added, the recipient will receive a text message or email. Click **Confirm Subscription**.
The subscription is successful if you receive a feedback indicating successful subscription.
After the recipient confirms the subscription, you need to restart the skill, so that the recipient will receive an email or SMS notification when the skill has any output.
In the subscription list, the recipient icon changes from gray to orange 📧.

6.2.5 Configuring Hard Example Settings

Some skills can collect hard examples. You need to set the hard example parameters before running the skills.

Prerequisites

- The skill supports hard example upload.
- You have created a hard example dataset on ModelArts to train the hard example filtering model.

Setting Hard Examples

1. In the navigation pane of the Huawei HiLens console, choose **Device Management > Devices**. In the device list, click the target device card. The **Skill Management** page is displayed by default.
2. On the **Skill Management** page, locate the target skill and choose **Runtime Configurations** in the **Operation** column.

Figure 6-10 Runtime configuration-17



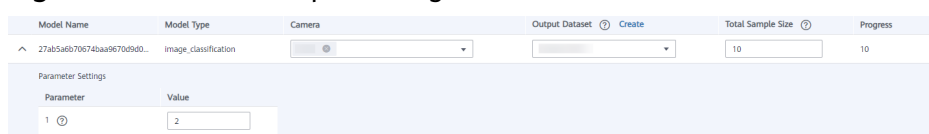
3. Click **Hard Example Settings** and enter related information.

Table 6-3 Hard example settings

Parameter	Description
Model Name	Name of a hard example filtering model. You do not need to set this parameter.
Model Type	Type of the hard example filtering model, such as image classification and object detection. You do not need to set this parameter.
Camera	Hard example data of the specified camera to be uploaded
Output Dataset	Dataset for training the hard example filtering model. You need to create a hard example dataset on ModelArts in advance.
Total Sample Size	Number of samples in the output dataset
Progress	Total volume of uploaded hard example data

Parameter	Description
Parameter Settings	Parameter values to be set for the hard example filtering model. Click ∨ on the left of the model name and set Value in Parameter Settings . You can click ? on the right of the parameter name to view the parameter description. See Figure 6-11 .

Figure 6-11 Hard example settings



4. After the configuration is complete, click **Execute Configuration** in the upper right corner.

6.2.6 Starting or Stopping Skills

You can start or stop skills on a device as required. After a skill is started successfully on a device, its status becomes **Running**, indicating that the skill has been used by the device. After a skill is stopped successfully on a device, its status becomes **Stopped**.

Before Your Start

- Before starting or stopping skills, upgrade the HiLens_Device_Agent firmware to version 1.0.7 or later. For details, see [Upgrading HiLens_Device_Agent Firmware](#).
- You can **Start** skills in the **Stopped** status.
- You can stop **Running** skills and uninstall **Faulty** or **Stopped** skills.
- After installing a skill, you need to start the skill so that it can run on the device. For details, see [Starting a Skill](#).

Starting a Skill

On the **Skill Management** tab page, select a skill whose status is **Stopped**, and click **Start** in the **Operation** column. Skill startup involves a command delivery process, so you need to wait for a period of time before you can view the skill status update on the page.

Figure 6-12 Starting a skill



Stopping a Skill

On the **Skill Management** tab page, select a skill whose status is **Running**, and click **Stop** in the **Operation** column. Skill stop involves a command delivery process, so you need to wait for a period of time before you can view the skill status update on the page.

Figure 6-13 Stopping a skill

Name	ID	Version	Status	Resource Constraints	Updated	Expiration Date	Operation
np-ml-ench		1.0.8	Running	16 Channel(s)	May 27, 2022	Permanent	Start Stop Uninstall Runtime Configurations Add Inst
np-ml-mgr		1.0.4	Running	16 Channel(s)	May 26, 2022	Permanent	Start Stop Uninstall Runtime Configurations Add Inst

6.2.7 Uninstalling Skills

If a skill is no longer needed, you can uninstall it on the Huawei HiLens console.

Before Your Start

- You can uninstall skills only from devices in the **Online (Running)** status. You cannot uninstall skills from offline devices.
- Uninstalling a skill will delete the skill and skill data on the device. Exercise caution when performing this operation.

Uninstalling a Skill

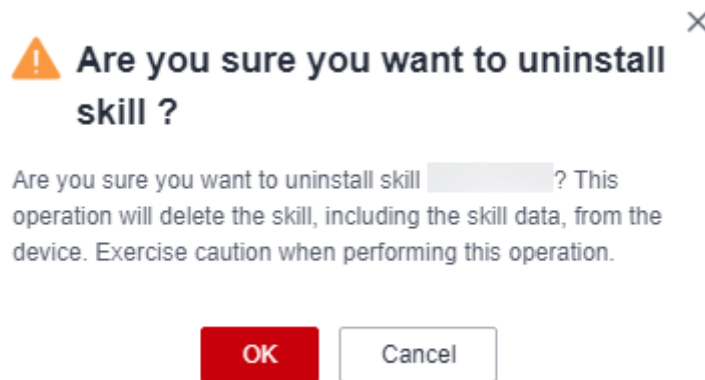
1. On the **Skill Management** tab page, select the target skill and click **Uninstall** in the **Operation** column.

Figure 6-14 Uninstalling a skill

Name	ID	Version	Status	Resource Constraints	Updated	Expiration Date	Operation
np-ml-ench		1.0.8	Running	16 Channel(s)	May 27, 2022	Permanent	Start Stop Uninstall Runtime Configurations Add Inst
np-ml-mgr		1.0.4	Running	16 Channel(s)	May 26, 2022	Permanent	Start Stop Uninstall Runtime Configurations Add Inst

2. In the dialog box that is displayed, confirm the information about the skill to be uninstalled and click **OK**.

Figure 6-15 Uninstalling a skill-18



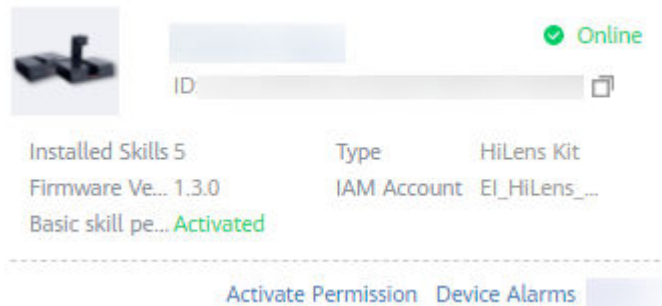
6.3 Viewing Device Information

On the device list page, you can view the basic information about or the upgrade status of a registered device.

Viewing Basic Device Information

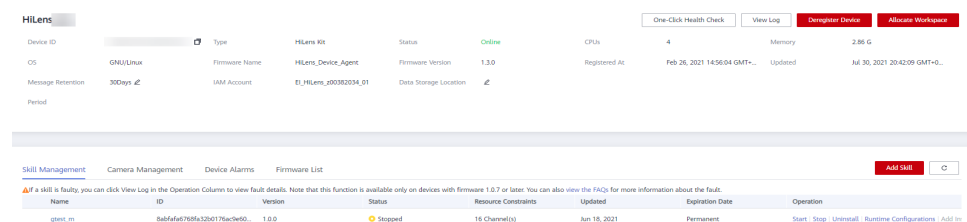
The **Device Management > HiLens Kit List** page displays basic information about each device, as shown in [Figure 6-16](#).

Figure 6-16 Device card



Click the name on a device card to enter the device details page, as shown in [Figure 6-17](#). You can view the device details and manage the device on this page, including managing skills and cameras, and upgrading firmware.

Figure 6-17 Device details page



6.4 Viewing Device Logs

After running a HiLens skill, you can view the device agent logs and skill logs.

Query Methods

- **Viewing Logs Using SSH:** Connect to the HiLens Kit device using SSH and view the logs.
- **Managing Data:** On the **Data Management (Beta)** page of the Huawei HiLens console, you can download and view skill running logs of each device.

Viewing Logs Using SSH

1. Connect to a HiLens Kit device using SSH. For details, see [Connecting a HiLens Kit Device to a PC](#).
2. Run corresponding commands to view logs based on the firmware version.
 - Firmware versions earlier than 1.0.4 (excluding 1.0.4):
 - Run the following command to view the device agent log:
tail -f /home/hilens/hda/log/hdad.log
 - Run the following command to view the skills log:
tail -f /home/hilens/hda/log/developer.skill name.skill ID.log
 - Firmware 1.0.4 to 1.1.1:
 - Run the following command to view the device agent log:
tail -f /home/log/alog/hilens/hda/hdad.log
 - Run the following command to view the skills log:
tail -f /home/log/alog/hilens/hda/developer.skill name.skill ID.log
 - Firmware 1.1.2 or later:
 - Run the following command to view the device agent log:
tail -f /home/log/alog/hilens/hda/hdad.log
 - Run the following command to view the skills log:
tail -f /home/log/alog/hilens/skills/developer.skill name.skill ID/developer.skill name.skill ID.log

NOTE

To ensure that your device can run newer and better skills, you are advised to upgrade its firmware in a timely manner. For details about how to upgrade the firmware, see [Upgrading HiLens_Device_Agent Firmware](#).

6.5 Upgrading HiLens_Device_Agent Firmware

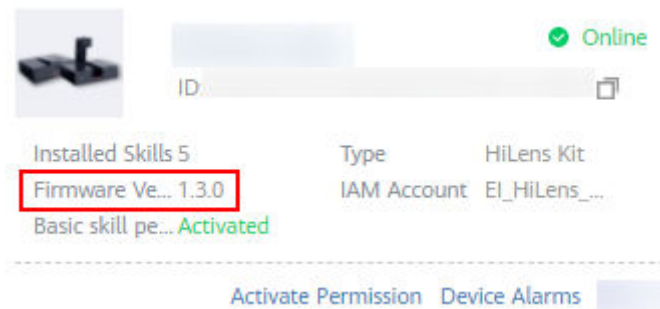
If the device manufacturer pushes a new firmware version, you can upgrade the HiLens Framework version on the Huawei HiLens console. As services keep evolving, engineers will continuously upgrade the HiLens Framework versions. To

ensure that your device can run newer and better skills, you are advised to upgrade its firmware in a timely manner.

Before You Start

- You can only upgrade firmware for **Online** devices.
- In the **HiLens Kit** list, if **Firmware Version** of a device is **Upgradable**, the firmware of a later version exists. You can upgrade it.
- When the firmware is being upgraded, you cannot perform other operations on the device.

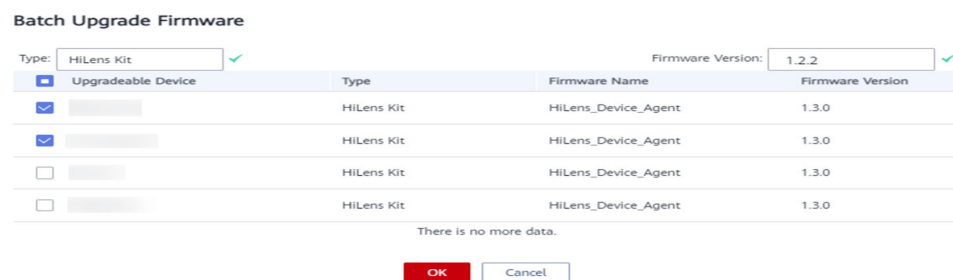
Figure 6-18 Prerequisites for upgrading firmware



Upgrading Firmware

1. In the navigation pane, choose **Device Management > Devices**.
2. Click **Batch Upgrade Firmware** in the upper right corner. The **Batch Upgrade Firmware** dialog box is displayed.
3. Select the firmware name and target firmware version. The devices that can be upgraded are displayed, including the device name, device type, firmware name, and firmware version.

Figure 6-19 Batch Upgrade Firmware



4. Select the device to be upgraded and click **OK**.
During the upgrade, you can view the firmware version and upgrade progress on the device details page. If the version **Status** displays **Upgrade succeeded**, the firmware is successfully upgraded.

6.6 Configuring Cameras

Each HiLens Kit device has a built-in camera, which can be enabled through **Video Collector**. A HiLens Kit device can connect and manage multiple IP cameras. (The

number of cameras per device cannot exceed the total number of channels supported by the skills installed on this device.) When connecting an external camera to a HiLens Kit device, you need to add and configure the camera on the console.

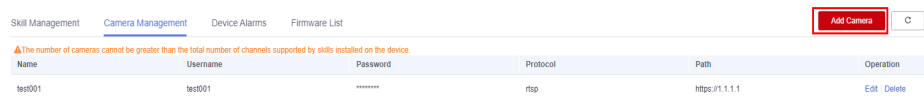
 **NOTE**

- Built-in HiLens Kit cameras do not support night vision.
- HiLens Kit can connect IP cameras of 4K or lower resolution, but cannot connect thermographic cameras.

Adding a Camera

1. In the navigation pane of the Huawei HiLens console, choose **Device Management > Devices**. In the device list, click the name of the target device to enter the device details page.
2. Click **Camera Management** to switch to the **Camera Management** tab on the device details page.

Figure 6-20 Camera Management



3. Click **Add Camera** in the upper right corner. In the displayed dialog box, enter related information.

Figure 6-21 Adding a camera

Add Camera

* Name

User Name

Password

* Protocol

* Path

Table 6-4 Camera configurations

Parameter	Description
Name	Name of a camera, which is used for identification. This parameter is user-defined.
User Name	Username for logging in to the IP camera. You can obtain it from the camera's user manual.
Password	Password for logging in to the IP camera.
Protocol	Protocol used by the camera to transmit videos. The default value is rtsp and cannot be changed.
Path	URL for accessing videos shot by the camera, for example, 192.168.0.1/root . Obtain it from the user guide of the camera.

4. After entering and confirming the information, click **OK** to add the camera and deliver the configurations to the device. A row of camera information is added on the **Camera Management** page.

Editing Camera Information

You can edit information for cameras that have been added to a device.

1. In the navigation pane of the Huawei HiLens console, choose **Device Management > Devices**. In the device list, click the name of the target device to enter the device details page.
2. Click **Camera Management** to switch to the **Camera Management** tab page on the device details page.
3. Locate the target camera and click **Edit** in the **Operation** column. Then modify the camera information and click **Execute Configuration**.

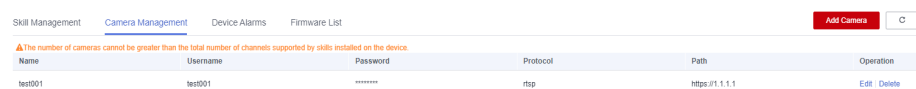
Only **User Name**, **Password**, **Protocol**, and **Path** can be modified. Once the **Name** is determined, it cannot be changed.

Deleting a Camera

You can delete unnecessary cameras that have been added to a device.

1. In the navigation pane of the Huawei HiLens console, choose **Device Management > Devices**. In the device list, click the name of the target device to enter the device details page.
2. Click **Camera Management** to switch to the **Camera Management** tab on the device details page.

Figure 6-22 Camera Management-19



3. Locate the target camera, click **Delete** in the **Operation** column, and click **OK**.

 NOTE

Deleted cameras cannot be recovered. Exercise caution when performing this operation.

6.7 Deregistering Devices

After a device is deregistered, it is unbound from the Huawei HiLens on the cloud. The HiLens console removes the corresponding device records and loses control over the device.

The device can be registered with Huawei HiLens under a new account only after it is deregistered from the current account.

Prerequisites

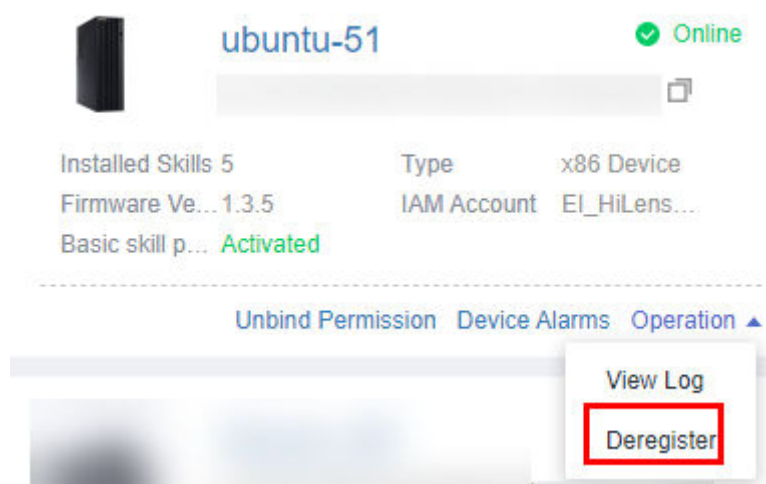
If the device to be deregistered has been activated, unbind the device before you deregister it.

In the navigation pane of the HiLens console, choose **Device Management > Devices**, click **Activate Permission** on the card of the target device, and unbind the device as prompted.

Deregistering a Device

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Device Management > Devices**. The device list page is displayed.
2. Select a device and choose **Operation > Deregister**.
You can also click the device name on a device card to go to the device details page and click **Deregister Device** in the upper right corner.

Figure 6-23 Deregistering a device



3. In the dialog box that is displayed, confirm the information about the device to be deregistered, and click **OK**.
After a device is deregistered, the skills deployed on it are uninstalled at the same time.

 NOTE

Before deregistering a device, you need to uninstall its skills purchased from the skill market. If you deregister a device without uninstalling the skills installed in the skill market, the skills and logs will be deleted when the device is registered again.

6.8 Viewing Device Alarms

When a device is faulty or works abnormally due to certain reasons, Huawei HiLens generates alarms based on the faults of different modules. You can view the device's hardware alarm information on the HiLens console and handle the alarm on the device.

Huawei HiLens can also monitor services. You can [configure alarm receiving settings](#). When skills on a device cannot run properly, Huawei HiLens sends service alarms to you in real time.

 NOTE

- The alarm function is available only on devices whose firmware version is 1.0.7 or later. The alarm notification function is available only on devices whose firmware version is 1.1.0 or later. For details about how to upgrade the firmware, see [Upgrading HiLens_Device_Agent Firmware](#).
- The alarm information of a device can be synchronized to the Huawei HiLens console only when the device is **Online**.

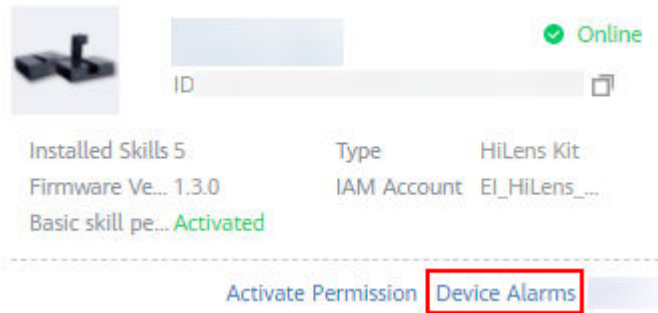
Alarm Severity

- **Minor**
A minor alarm has a minor impact on the system, but you need to take corrective actions as soon as possible to prevent a more severe alarm.
- **Major**
A major alarm has a major impact on the system. It affects the normal operating of the system or may cause service interruption.
- **Critical**
A critical alarm may power off the device, and even interrupt system services. You must take corrective actions immediately.

Procedure for Viewing Alarms

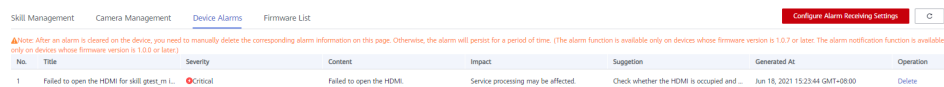
In the navigation pane of the Huawei HiLens console, choose **Device Management > Devices**. On the card of the device where an alarm is generated, click **Device Alarms** to enter the corresponding tab page.

Figure 6-24 Device alarms



You can view the alarm information in the device's alarm list and handle the alarm on the device. The alarm information includes the alarm **Title**, **Severity**, **Content**, **Impact**, **Suggestion**, **Generated At**, and **Operation**. Device alarms have three severity levels: **Minor**, **Major**, and **Critical**. For details, see [Alarm Severity](#).

Figure 6-25 Device alarm list

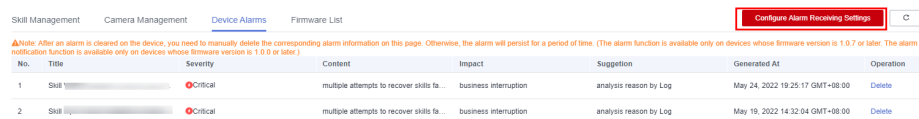


After you handle the alarm on the device, the alarm will be automatically cleared if no exception is detected on the device.

Configuring Alarm Receiving Settings

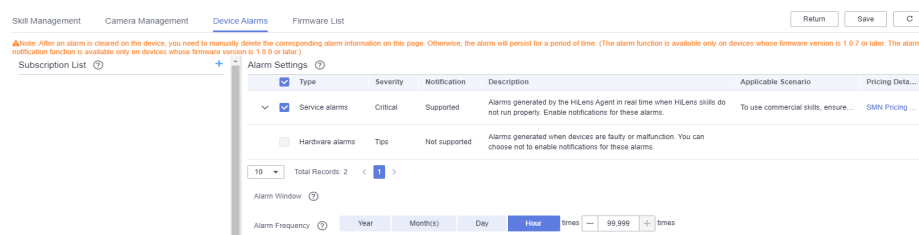
1. Log in to the HiLens console. In the navigation pane, choose **Device Management > Devices**. The device list page is displayed.
2. Click **Device Alarms**. On the right of the **Device Alarms** page, click **Configure Alarm Receiving Settings**.

Figure 6-26 Configure Alarm Receiving Settings




3. In the **Alarm Settings** area, select service alarms that can be received in real time, and specify **Alarm Window** and **Alarm Frequency**.


Figure 6-27 Device Alarms-20



- **Alarm Window** is the period during which alarms can be sent to subscribers.
- **Alarm Frequency** is the maximum number of times alarms can be sent to subscribers during the specified period. For example, if the alarm frequency is set to **3** times per **Day**, the maximum number of the alarms can be sent to you per day is 3.

Click  on the left of a service alarm to view the alarm description and impact.

Hardware alarms cannot be received in real time.

4. Click **Save** in the upper right corner of the **Device Alarms** tab page.
5. Click **+** on the right of **Subscription list**, select a subscription type (text message or email), enter the communication mode (mobile number or email address), and set the alarm recipient.
6. Confirm the information and click .
7. After the message recipient is added, the recipient will receive a text message or email. Click **Confirm Subscription**.

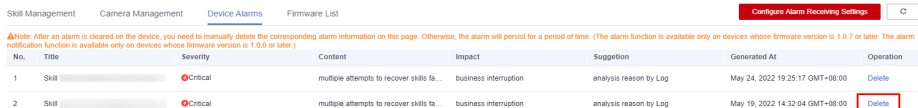
The service alarm subscription is successful if you receive a feedback indicating successful subscription.

Deleting an Alarm

You can delete alarms based on actual conditions.

1. In the navigation pane of the Huawei HiLens console, choose **Device Management > Devices**. On the card of the device where an alarm is generated, click **Device Alarms** to enter the corresponding tab page.
2. In the alarm list, click **Delete** in the **Operation** column of a processed alarm. The **Delete Alarm** dialog box is displayed.

Figure 6-28 Deleting an alarm



No.	Title	Severity	Content	Impact	Suggestion	Generated At	Operation
1	Skill [redacted]	Critical	multiple attempts to recover skills fa...	business interruption	analysis reason by Log	May 24, 2022 19:25:17 GMT+08:00	Delete
2	Skill [redacted]	Critical	multiple attempts to recover skills fa...	business interruption	analysis reason by Log	May 19, 2022 14:32:04 GMT+08:00	Delete

3. Click **OK**.

6.9 Performing One-Click Health Check

Due to time zone and network problems, devices may be displayed as offline on the console, skills fail to be started, or firmware fails to be upgraded. You can perform one-click health checks to quickly locate and resolve the problems.

Prerequisites

The cloud-side one-click health check function is available for devices whose firmware version is 1.1.0 or later. For details about how to upgrade the firmware, see [Upgrading HiLens_Device_Agent Firmware](#).

Cloud-Side Health Check Items

- **Process Information:** Check whether the basic process is running properly.
- **Dependency Library:** Check whether the dependency library of the device is deleted by mistake and whether it can be properly called.
- **Time:** Check whether the time zone and time are correct.

Cloud-Side Health Check Procedure

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Device Management > Devices**.
2. On the displayed device list, click the card of the target device.
The device details page is displayed.
3. Click **One-Click Health Check** in the upper right corner.

Wait for a period of time. After the system check is complete, the **Health Check** dialog box is displayed, showing the process, dependency library, and time statuses.

If a status is abnormal, the cause of the exception is displayed in the **Health Check** dialog box. You can rectify the fault based on the cause.

Figure 6-29 One-Click Health Check



Device-Side Health Check Items

You can also log in to the HiLens Kit device system through SSH to perform health checks on devices.

- Checking network connectivity of devices
- Checking whether the type of the wireless network connected to the device meets the requirements
- Checking whether a device is offline

Device-Side Health Check Procedure

1. [Connecting a HiLens Kit Device to a PC](#)
2. [Logging In to a HiLens Kit Device Using SSH](#)
3. Running `hdactl health_check`

7 Developing Skills on the Console

7.1 Skill Overview

A skill can be abstractly understood as "algorithm model + logic code". The algorithm model handles AI inference, and the logic code handles inference result processing.

This section describes how to use empty templates and basic skill templates to develop skills on the Huawei HiLens console.

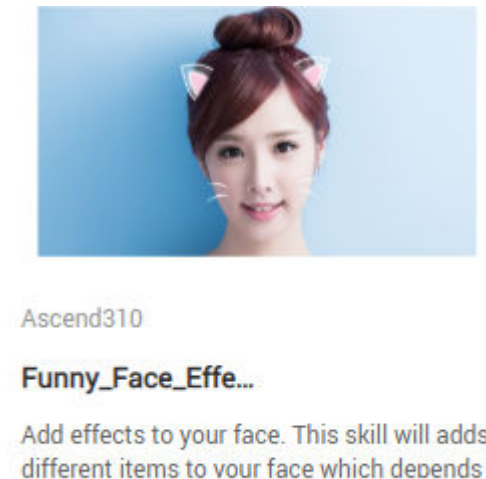
Skill

A skill is an AI application ready to run on a camera or another computing device. It consists of a model and logic code. The logic code is the skill's framework that governs how the skill behaves, including data reading, model import, model inference, and result output. The model is an AI algorithm trained using big data and handles inference while the skill is running.

The [Skill Market](#) of Huawei HiLens provides extensive skills for users.

- The application scenarios for HiLens skills include smart campus, smart home, smart vehicle-mounted terminals, smart shopping mall, and more.
- Skills are classified into two types by device: skills for Ascend chips and skills for HiSilicon Hi35xx series chips.

Figure 7-1 Skill card

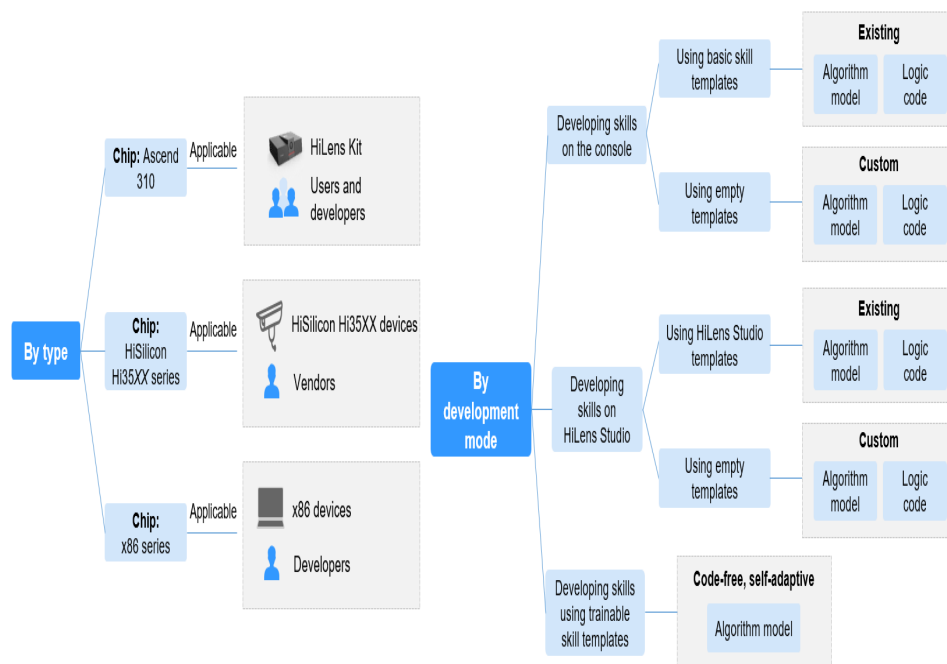


Skill Development Overview

Skills are classified into two types based on the chip types: skills used on HiLens Kit devices and skills used on devices that run HiSilicon Hi35xx series chips. The HiLens platform supports quick, convenient, and efficient skill development using skill templates. It also allows you to develop models and logic code without using templates, meeting your personalized requirements in more scenarios.

Skills that run on HiSilicon Hi35xx series chips require a large chip memory and high performance. You need to optimize the skill models. If you want to develop skills of this type, contact the support team of the Huawei HiLens platform.

Figure 7-2 Skill development



Developing New Skills on the Console

Before creating a skill, determine its type. Different skills have different models and logic code, but their creation process is the same.

 **NOTE**

Developing skills on the Huawei HiLens console may occupy OBS resources and incur fees. For details about OBS pricing, see [Product Pricing Details](#).

For details about the skill creation process, see [Figure 7-2](#).

- **Using basic skill templates:** Ensure that there are available basic skill templates on the platform.
- **Using empty templates:** To develop a skill using an empty template, you need to develop its model and logic code. If your model format does not meet our requirements, the Huawei HiLens platform will convert the model format during model import.

Figure 7-3 Skill development

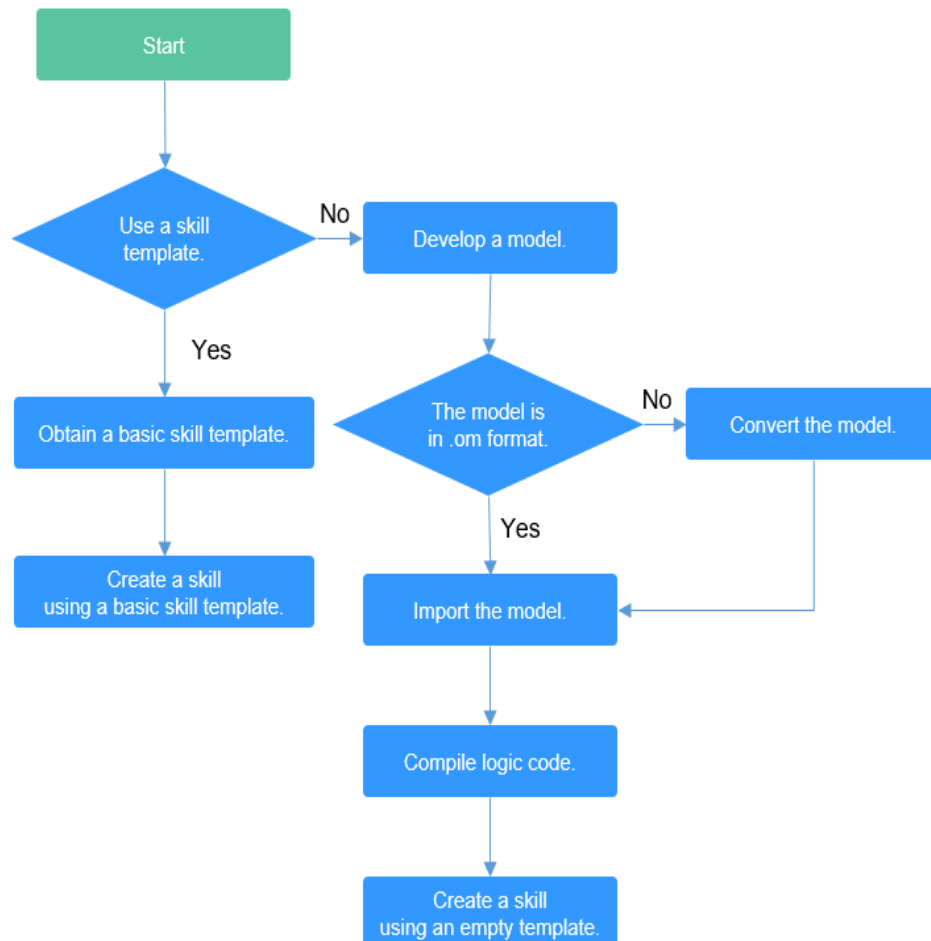


Table 7-1 Common operations for creating a skill

Category	Common Operation	Quick Link
Using a skill template	Obtain a skill template.	Obtaining Skill Templates
	Create a skill using the basic skill template.	Creating Skills
Using an empty template	Develop a model.	Developing an Algorithm Model
	Import (convert) the model.	Importing (Converting) Models
	Compile the logic code.	Writing the Code
	Create a skill using the empty template.	Creating Skills

Related Operations

On the Huawei HiLens platform, you cannot only develop skills, but also manage your developed skills. The detailed operations are as follows.

Table 7-2 Skill Related Operations

Operation	Description
Creating Skills	After determining the type of the skill to be developed, create the skill according to Developing New Skills on the Console .
Editing Skills	After a skill is created, you can edit and modify the skill, including its basic information, content, and runtime configurations.
Installing and Debugging Skills	After a skill is created, you can deploy it to devices, view the skill videos to check its effects, and debug it repeatedly.
Deleting Skills	You can delete skills that are no longer used to release resources.

7.2 Creating Skills

7.2.1 Creating a Skill Using a Skill Template

This section describes how to use a basic skill template to create a skill. When you use a basic skill template to create a skill, the default values in the skill template

can be directly used, including the logic code and algorithm model. You only need to enter basic information about the skill.

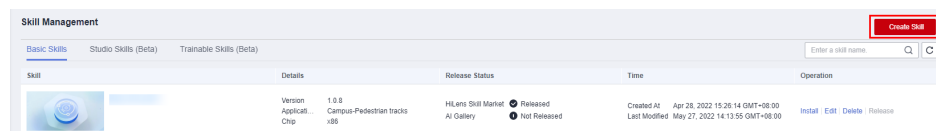
Prerequisites

- Ensure that the Huawei Cloud account is not in arrears. Developing skills on the Huawei HiLens console may occupy OBS resources and incur fees. For details about the OBS pricing, see [Product Pricing Details](#).
- You have searched and obtained the skill template that meets your service response requirements on the [Skill Templates](#) page. For details, see [Obtaining Skill Templates](#).

1. Starting to Create a Skill

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Skills**.
2. Click **Create Skill** in the upper right corner. The **Create Skill** page is displayed.

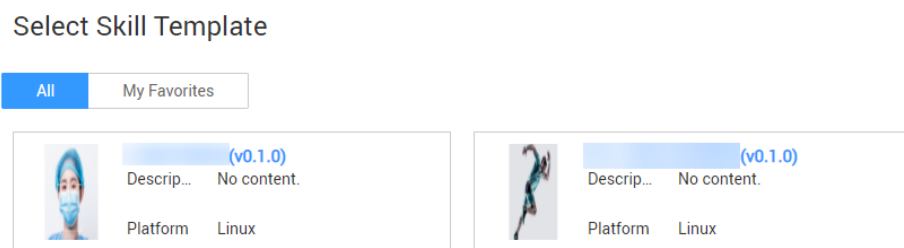
Figure 7-4 Creating a skill



2. Selecting a Template

1. Click **Select Template** for **Skill Templates**. The **Select Skill Template** dialog box is displayed.

Figure 7-5 Selecting a skill template



2. Select the template you want and click **OK**.
The area for editing the skill content on the **Create Skill** page is displayed. The default configuration of the selected template is automatically loaded. You can view the related information on the right pane of the page.

3. Filling in Related Information

After selecting an existing template, you can enter the skill information, including the **basic information**, **skill content**, and **(Optional) runtime configurations** by referring to [Creating a Skill Using an Empty Template](#).

The **Skill Code Value** is automatically initialized based on the selected template. You need to specify parameters such as the **Name** and **Skill Version**, which are

not contained in the skill template. You can modify the content of the skill template as required or retain all of the default values.

4. Confirming Information and Creating the Skill

After setting the preceding parameters, you can view the parameter values on the right pane. If a parameter is incorrectly set, a red cross is displayed on the right.

Confirm the information and click **OK** to create the skill.

After the skill is created, the **Skill Development > Skill Management** page is displayed.

After the skill is released, its status becomes **Reviewing**. Huawei HiLens personnel will complete the review within three working days. After the skill release is approved, the status changes **Released**.

7.2.2 Creating a Skill Using an Empty Template

This topic describes how to use an empty template to create a skill. If you choose to use an empty template, you need to prepare the algorithm model and logic code in advance.

Context

- Developing skills on the Huawei HiLens console may occupy OBS resources and incur fees. For details about OBS pricing, see [Product Pricing Details](#).
- The algorithm model must be in **.om** format and meet the requirements of Huawei HiLens. For details, see [Developing an Algorithm Model](#).
- Logic code can be handled (stored) in either of the following ways: **Online Compilation** and **Upload From OBS**.
 - If your code logic is simple, you are advised to compile the code online.
 - If your code structure is complex, you are advised to develop the code in a local integrated development environment (IDE) and upload it in **.zip** or **.tar.gz** format from OBS. For details about how to use OBS to upload files, see [Object Storage Service Getting Started](#). Uploading files to OBS is charged. For details about OBS pricing, see [Product Pricing Details](#).

NOTE

- Before uploading a file, ensure that the OBS bucket is in the same region as Huawei HiLens.
- If you upload code from OBS, compress all code files before uploading them. The uploaded files must be in **.zip** or **tar.gz** format, and the main file must be in the level-1 directory. As shown in the following example, the entry code (**main.py**) is in the level-1 directory. Other code needs to be designed as required. You can pack the model and code together to upload them.
- The file (for example, **main.py**) where the entry code is located is configured through the **Function Execution Entry** parameter, which will be described later.

The following is a code directory example:

```
skill/  
|--main.py      #Main file. The file name is the same as the value of parameter Code  
Configuration of the new skill.  
|--depends/     #Optional, used to store the model file.
```

```
|--workspace/ #Workspace, used to store data generated by the skill.
|--data/ # Used to store runtime configurations.
```

- If the skill is in the container image format, upload the image to SWR. For details, see [Uploading an Image Through a Container Engine Client](#).

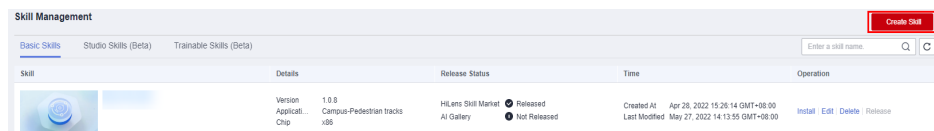
Prerequisites

- Ensure that your account is not in arrears. Developing skills on the Huawei HiLens console may occupy OBS resources and incur fees. For details about OBS pricing, see [Product Pricing Details](#).
- The model used for creating a skill has been [imported to HiLens](#).
- If you select **Upload From OBS**, upload the logic code to your OBS bucket in advance.

1. Starting to Create a Skill

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Skills**.
2. Click **Create Skill** in the upper right corner. The **Create Skill** page is displayed.

Figure 7-6 Creating a skill



2. Filling In Basic Information

On the **Create Skill** page, select **Create from Scratch** for **Skill Templates** and fill in the basic information. [Table 7-3](#) describes related parameters.

Figure 7-7 Filling in basic information

Basic Information

Skill Templates: **Create from Scratch** | Select Template

* Name:

* Skill Version:

* Chip:

* Skill Code Value:

* Application Scenario:

Icon:

* OS Platform: Linux | Android | iOS | LiteOS | Windows

Description:

B I H

The description consists of a maximum of 2,048 characters and cannot contain the following special characters: ~^\$%&.

Table 7-3 Basic information

Parameter	Description
Skill Templates	Whether to use a template to develop a skill. If you select an existing template, the model and code of the template are used. That is, all parameters in the Skill Content module use the values specified by the template.
Name	<p>Name of a skill. The name is also used as the root directory name when the skill is delivered to a device.</p> <p>The name is a string of 3 to 60 characters that can contain only letters, digits, hyphens (-), and underscores (_). It must start with a letter and end with a letter or digit.</p> <p>NOTE Skills in the skill market cannot share the same name. If you want to release a skill to the skill market, use a globally unique name to name the skill.</p>
Skill Version	<p>Skill version. The version number is in the format of A.A.A, where A is a natural number consisting of a maximum of three digits, for example, 1.0.0.</p> <p>If A contains more than one digit, it cannot start with 0. For example, 01.0.0 is not allowed.</p>
Chip	Chip supported by the skill. For skills running on HiLens Kit devices, select the Ascend 310 chip.
Skill Code Value	Used for skill verification to prevent fake skills. The value must be the same as that on the initialization interface in the code. For details about the usage, see Initializing the HiLens Framework in the <i>HiLens Developer Guide</i> .
Applicable Scenario	Application scenario of the skill, such as sub-scenarios of Smart Campus, Smart Home, Smart In Vehicle Device, and Smart Mall and Supermarket, and Others
Icon	Icon of the skill
OS Platform	Operating system platform where the skill runs, including Linux, Android, iOS, LiteOS, and Windows. The HiLens Kit device uses the Linux distribution EulerOS developed based on CentOS. Therefore, if the skill is to run on HiLens Kit devices, select Linux .
Description	<p>Detailed description about the skill.</p> <p>You can enter a maximum of 2048 characters. Special characters ~^\$%& are not allowed.</p> <p>NOTE If messages need to be sent to users' mobile phones or email addresses, (for example, when a stranger is detected, a message needs to be sent to notify the user), you need to add "This skill supports notification messages." in the description. Specify the message to be sent in Writing the Code. For details, see the <i>HiLens Developer Guide</i>.</p>

3. Setting the Skill Content

Enter the skill content based on your model and logic code. For details about the parameters, see [Table 7-4](#).

Figure 7-8 Setting the skill content

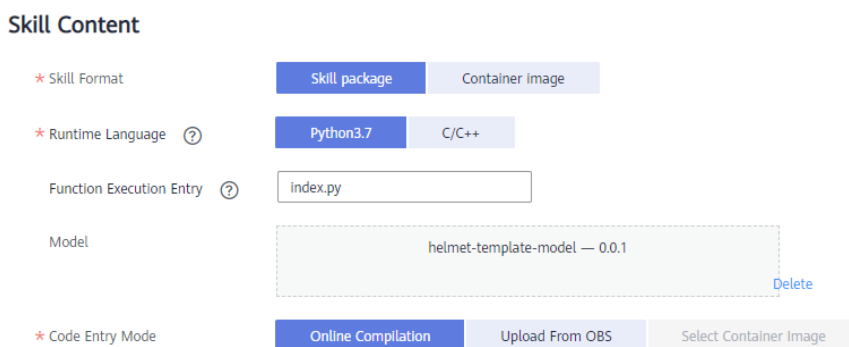


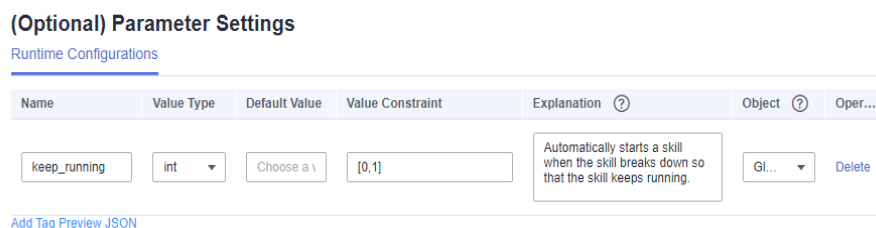
Table 7-4 Skill content parameters

Parameter	Description
Skill Format	Format of the skill. The value can be Skill package or Container image .
Runtime Language	Runtime language of the logic code. Currently, the value can be Python3.7 or C/C++ . If you select C/C++ for development, online code compilation is not supported. You need to develop the code offline, compile and pack the code in the Linux environment, and upload it to OBS.
Function Execution Entry	<p>The startup of a skill uses the code execution file as the entry, which is similar to the main function of the C language. The main file must be in the level-1 directory.</p> <ul style="list-style-type: none"> For skills of the Skill package format, the main file must be in the root directory of the skill package. For example, a Python skill file must end with .py. For skills of the Container image format, if ENTRYPOINT is configured in dockerfile, the description can be left blank. Otherwise, the path of the execution file in the image must be entered. <p>The file name contains a maximum of 1024 characters. Only letters, digits, and underscores (_) are allowed. Special characters #~^ are not allowed.</p>

Parameter	Description
Model	Core algorithms of the skill. Train the model locally or in ModelArts and import it to the HiLens platform . Click the plus sign (+). In the dialog box that is displayed, select a model in the model management list on the Huawei HiLens platform.
Code Entry Mode	<p>Upload mode of the logic code.</p> <p>NOTE Firmware 1.1.2 or later supports configuring Python dependencies for skills. During skill development, you can configure Python dependencies for a skill as required. For details, see How Do I Configure Python Dependencies for a Skill.</p> <ul style="list-style-type: none"> Online Compilation When you use the logic code for online skill development, a .zip package is automatically generated and uploaded to OBS. HiLens automatically creates an OBS bucket for storing skill packages. The naming rule of the bucket is <project_id>-hilens-skill. You can go to OBS, find the OBS bucket with the corresponding naming rule, and obtain the code package file edited online. You can create a file or folder in the editing area to edit the logic code. The created file named index.py by default. You can rename it or create a file or folder that is not in .py format as required. Upload From OBS You can develop complex project code offline, manually generate a .zip or .tar.gz package, upload it to OBS, and select the .zip or .tar.gz package from OBS here. Before uploading the code to OBS, you can package the model together. For details about the model and code structures, see the specifications defined in Context. Select Container Image If the skill is in the container image format, upload the image to SWR. For details, see Uploading Images Through the Docker Client.

4. (Optional) Setting Runtime Configurations

Figure 7-9 Runtime configuration



You need to set parameters for some running skills. For example, import facial image libraries when using facial recognition skills. If users configure runtime configurations when running a skill, HiLens will record the configurations in the code for other developers to use. You can click **Add Tag** on the page and enter your configuration by referring to [Table 7-5](#).

You can also click **Preview JSON** to view the runtime configuration's format. Developers can obtain the skill configurations in JSON format through the **get_skill_config** interface provided by the HiLens Framework and read the field values to use the user configurations.

The runtime configurations in JSON format are as follows:

```
{
  //Global configuration
  "Language": "enum",
  //Channel-based video configuration
  "multi_camera": [ //The key value multi_camera is fixed, indicating channel-based camera configuration.
    {
      "camera_names": [ //The key value camera_names is fixed, indicating the cameras included in
the group configuration.
        "cameraX",
        "cameraY"
      ],
      "FaceLib": { //Configuration name
        "from": "file source", //File type value, indicating the service from which the file comes, for
example, OBS.
        "path": "file path" //File type value, indicating the file path.
      }
    }
  ],
}
```

Table 7-5 Runtime configuration parameters

Parameter	Description
Name	Configuration name. This field can be used to obtain the configuration value in the logic code. A runtime configuration refers to the content configured by users when a skill is running. For example, facial recognition skills require users to upload facial image libraries. Note that the runtime configuration is optional. If a configuration is added, the configuration Name and Description are mandatory.
Value Type	Type of the value of a configuration item. The value can be int , float , enum , string , or file .
Constrain	Value range of the configuration. This parameter is related to the value type. <ul style="list-style-type: none"> ● Numeral type (int and float): The value range is an open or closed interval. For example, [1, 100) indicates that the value is greater than or equal to 1 and less than 100. ● string and file: You do not need to set this parameter. ● enum: The value range is a set, for example, {a, b}.


Parameter	Description
Explanation	Describes the function of the configuration item and how to configure it. This field is important. Users can set a configuration based on the description here. Therefore, if a skill has a runtime configuration, developers must explain it clearly.
Object	<p>A HiLens Kit device can connect to multiple external cameras (the number of cameras cannot be greater than the total number of channels supported by the skills installed on the device). Therefore, you can develop skills that support multiple video channels.</p> <ul style="list-style-type: none"> • If you select Video, the configuration takes effect for videos of each camera on the device. For example, camera 1 and camera 2 use different facial image libraries. • If you select Global, all cameras on the device use the same configuration value.

6. Confirming Information and Creating the Skill


After setting the preceding parameters, you can view the parameter values on the right pane. If a parameter is incorrectly set, a red cross is displayed on the right.

Confirm the information and click **OK** to create the skill.


Figure 7-10 Confirming information and creating the skill

Basic Information 


Skill Templ...	Create from ...
* Name	test
* Skill Version	1.0.0
* Chip	Ascend310
* Skill Code V...	helmet
* Application ...	Others- <input type="text"/>
Icon	
* OS Platform	Linux
Description	

Skill Content 

* Skill Format	file
* Runtime La...	Python3.7
Function Ex...	index.py
Model	[{"model_id":...]
Model Key	
Code Entry ...	Online Comp...

(Optional) Runtime Configuration 

server_url	<input type="text"/>
IPC_address	<input type="text"/>

(Optional) Elastic Skill Parameters 

Web Requ...

After the skill is created, the **Skill Development > Skills** page is displayed.

7.3 Obtaining Skill Templates

Skill templates can be used to quickly create skills to improve the development efficiency.

This section describes how to obtain and use skill templates.

Skill Template Introduction

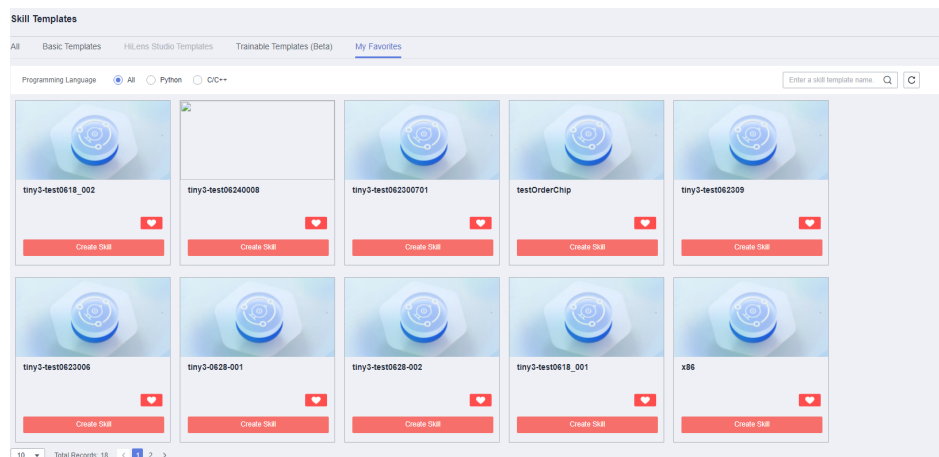
You can also perform the following operations on the basic templates:

- [Viewing a Skill Template](#)
- [Downloading a Skill Template](#)
- [Adding a Skill Template to Favorites](#)
- [Creating a Skill Using a Skill Template](#)

Viewing a Skill Template

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Skill Templates**. The **All** tab page is displayed.
2. Click a skill template card. The skill template's details page is displayed, on which you can view the basic information and runtime parameters of the skill. You can click the **My Favorites** tab to view the list of skills you have added to favorites, and click a skill card to go to the details page.

Figure 7-11 Skill Templates

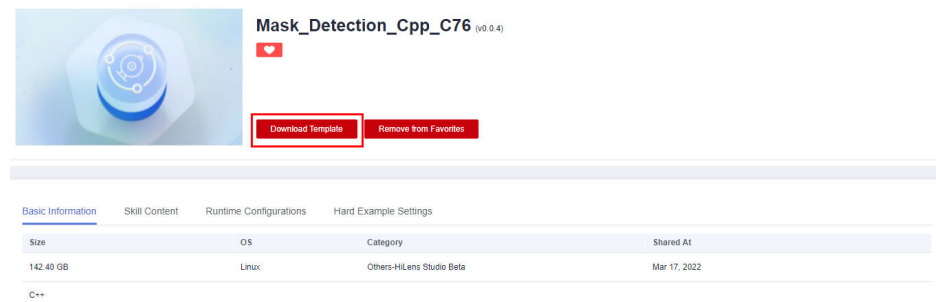


Downloading a Skill Template

1. On the **All**, **Basic Templates**, or **My Favorites** tab page, click a skill template card. The details page is displayed.
2. Click **Download Template** in the upper right corner to download the skill template package to your local PC.

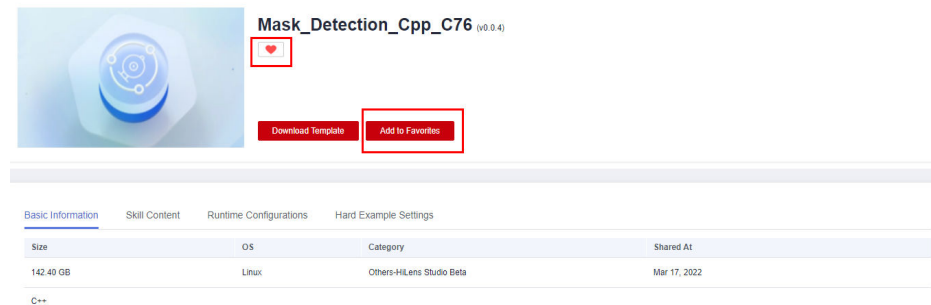
The package is in the **.zip** format. A downloaded package contains the algorithm model and logic code of the skill template. Some skill templates may not have algorithm models. In this case, the package contains only the logic code file.


Figure 7-12 Downloading a skill template




Adding a Skill Template to Favorites

Figure 7-13 Adding a skill template to favorites



- On the **All** tab page, click a skill template card. The details page is displayed. Click **Add to Favorites** in the upper right corner.
- On the **All** tab page, click the  icon on a skill template card.

After you add a skill template to favorites, it will be displayed on the **My Favorites** page. Click the  icon on the card again or click the **Remove from Favorites** button in the upper right corner of the skill details page to remove the template from your favorites.

Creating a Skill Using a Skill Template

1. On the **All**, **Basic Templates**, or **My Favorites** tab page, click a skill template card. The details page is displayed.
2. Click **Create Skill Using Template** in the upper right corner. The **Create Skill** page is displayed. For details about the skill creation parameters, see [Creating a Skill Using a Skill Template](#).

7.4 Managing Algorithm Models

7.4.1 Developing an Algorithm Model

Prerequisites

Currently, the HiLens Kit AI chip supports only **.om** models, which can be converted from TensorFlow or Caffe models. However, **.om** models do not support all operators of the TensorFlow and Caffe models. Therefore, you need to use operators supported by **.om** models during TensorFlow/Caffe model development. In this case, the TensorFlow and Caffe models can be converted into **.om** models. For details about the TensorFlow and Caffe operator boundaries supported by **.om** models, see [Caffe Operator Boundaries](#) and [TensorFlow Operator Boundaries](#).

NOTE

- .om** models are not fully compatible with the built-in Keras APIs of TensorFlow.
- .om** models do not support Caffe2.

Developing Models Using ModelArts

ModelArts is a one-stop development platform for AI developers. You can use ModelArts to develop algorithm models used on the Huawei HiLens platform.

After a model is developed or trained using ModelArts, it is stored in the model list of ModelArts. . Currently, Huawei HiLens supports only the algorithm models developed using TensorFlow and Caffe engines. Therefore, you are advised to select a correct engine type and model storage format when using ModelArts for development.

Models generated by ExeML of ModelArts cannot be used on the Huawei HiLens platform.

Developing Models Offline

You can use a familiar algorithm model development tool to develop an algorithm model locally.

Currently, only the algorithm models developed by TensorFlow or Caffe engines are supported, and the models you developed must be saved in **.pb** or **.caffemodel** format. Then, use the model import/conversion function of Huawei HiLens to convert the models to the **.om** format supported by Ascend 310 chips.

Follow-up Operations

After models are developed, you need to import them into the Huawei HiLens console. Then, use the model import/conversion function of Huawei HiLens to convert models developed using TensorFlow and Caffe engines to the **.om** format supported by Ascend 310 chips. For details, see [Importing \(Converting\) Models](#).

7.4.2 Importing (Converting) Models

A skill can be abstractly understood as "algorithm model + logic code". The algorithm model handles AI inference, and the logic code handles inference result

processing. Therefore, you need to import models to Huawei HiLens before developing skills.

Model Requirements

The models to be imported can be in **.om**, **.pb**, or **.caffemodel** format. Only models in **.om** format can run on HiLens Kit devices. If you import a model in **.pb** or **.caffemodel** format, the Huawei HiLens platform automatically converts it into the **.om** format.

Not all models can be successfully converted. Before importing and converting a model, check whether it uses the TensorFlow and Caffe operator boundaries supported by **.om** models. For details, see [Caffe Operator Boundaries](#) and [TensorFlow Operator Boundaries](#).

Prerequisites

You can train a model in ModelArts or locally.

- Importing a model trained locally

Before importing a custom model, upload it to OBS. A non-om model package contains Caffe model files **.caffemodel** and **.prototxt** and configuration file **.cfg**, or TensorFlow model file **.pb** and configuration file **.cfg**.

For details about how to upload model files to OBS, see [Object Storage Service Getting Started](#). The directory to which the model files are uploaded must meet certain specifications. For details, see [Model Input Directory Specifications](#).

Uploading files to OBS is charged. For details about the OBS pricing, see [Product Pricing Details](#).

When uploading files to OBS, ensure that the OBS bucket and Huawei HiLens are in the same region and the OBS folder name meets the following requirements:

- Cannot contain special characters: \:*?"<>|
- Cannot start or end with a period (.) or slash (/).
- The absolute path of the folder can contain a maximum of 1,023 characters.
- Cannot contain two or more adjacent slashes (/).

Importing (Converting) a Model

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Models**. The **Models** page is displayed.
2. Click **Import (Convert) Model** in the upper right corner.
3. On the **Import Model** page, set parameters by referring to [Table 7-6](#), confirm the information, and click **OK**.

Figure 7-14 Importing a model

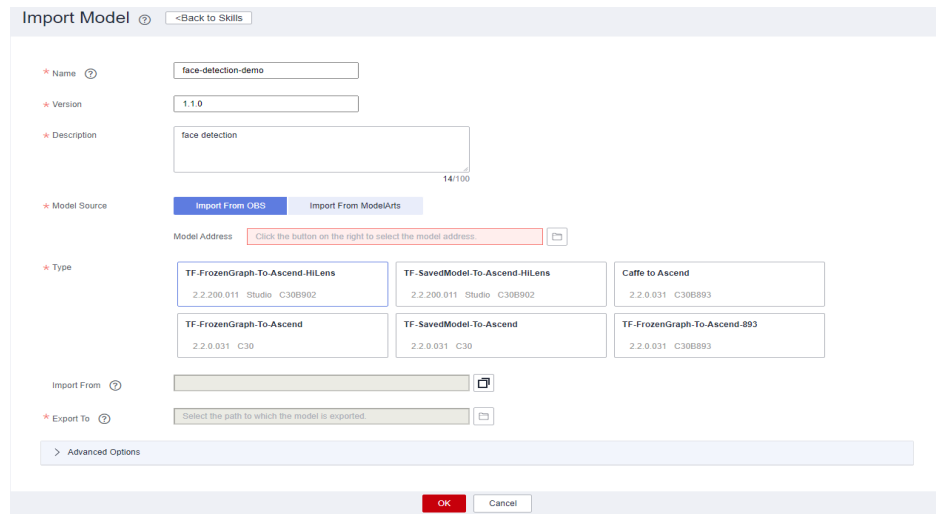



Table 7-6 Parameters for importing a custom model

Parameter	Description
Name	Name of the model to be imported. A model name contains 2 to 24 characters. Only lowercase letters, digits, and hyphens (-) are allowed. It must start with a lowercase letter and end with a lowercase letter or digit.
Version	Version of the model to be imported. The value is in the <i>number.number.number</i> format, for example, 2.1.3 . Each number ranges from 0 to 999 . When it has two or more digits, it cannot start with 0 .
Description	Description of the model to be imported. The value contains 1 to 100 characters. Special characters &!""\<>= are not allowed.
Model Source	Select the source of the model to be imported. You can choose Import From OBS or Import From ModelArts . <ul style="list-style-type: none"> Importing a model from OBS Click Import From OBS, select the bucket and folder where the custom model is stored on OBS, and click OK. Importing a model from ModelArts <ol style="list-style-type: none"> Click Import From ModelArts and select a model framework from the drop-down list on the right, including TensorFlow, Caffe, and OM (obtained from the conversion task). After you import a TensorFlow or Caffe model, the Huawei HiLens platform converts it into the .om format and then imports it. The .om model is obtained from the previous conversion task. Select the model to be imported from the model list.

Parameter	Description
Type	<p>If you select a model (non-om model) whose format needs to be converted for Model Source, you need to select a model conversion template, which can be TF-FrozenGraph-To-Ascend-HiLens, TF-SavedModel-To-Ascend-HiLens, TF-FrozenGraph-To-Ascend, TF-SavedModel-To-Ascend or Caffe to Ascend.</p> <ul style="list-style-type: none"> ● TF-FrozenGraph-To-Ascend-HiLens This template converts TensorFlow frozen_graph models into those run on Ascend chips. If the firmware version of your HiLens Kit system is 2.2.200.011, you are advised to use this template for model conversion. ● TF-SavedModel-To-Ascend-HiLens This template converts TensorFlow saved_model models into those run on Ascend chips. If the firmware version of your HiLens Kit system is 2.2.200.011, you are advised to use this template for model conversion. ● TF-FrozenGraph-To-Ascend This template converts TensorFlow frozen_graph models into those run on Ascend chips. If the firmware version of your HiLens Kit system is earlier than 2.2.200.011, you are advised to use this template for model conversion. ● TF-SavedModel-To-Ascend This template converts TensorFlow saved_model models into those run on Ascend chips. If the firmware version of your HiLens Kit system is earlier than 2.2.200.011, you are advised to use this template for model conversion. ● Caffe to Ascend Caffe models can be converted into models that can run on Ascend chips. ● TF-FrozenGraph-To-Ascend-893 This template converts TensorFlow frozen_graph models into those run on Ascend chips. If the firmware version of your HiLens Kit system is earlier than 2.2.200.011, you are advised to use this template for model conversion.
Import From	<p>If you select a model (non-om model) that needs to be converted in Model Source, this parameter is automatically initialized based on the model source. The directory must contain the model file and configuration file required for conversion. For details about the input directory specifications of model files, see Model Input Directory Specifications.</p>
Export To	<p>If you select a model (non-om model) that needs to be converted in Model Source, click  to select the OBS directory to which the converted model is exported. Ensure that no .om model exists in the directory. For details about the output directory specifications, see Model Output Directory Specifications.</p>

Parameter	Description
Advanced Options	Includes the Input Tensor Shape and out_nodes parameters. For details, see Table 7-7 .

Table 7-7 Advanced Options

Parameter	Description
Input Tensor Shape	<p>If you select a model (non-OM model) whose format needs to be converted for Model Source and the conversion type is TF-FrozenGraph-To-Ascend-HiLens or TF-SavedModel-To-Ascend-HiLens, you must enter the tensor shape.</p> <p>Input Tensor Shape indicates the shape of the input data of the model. The input data format is NHWC, for example, input_name:1,224,224,3. input_name must be the node name in the network model before model conversion. This parameter is mandatory when the model has dynamic shape input. For example, in input_name1?:h,w,c, the question mark (?) indicates the batch size, that is, the number of images processed at a time. It is used to convert the original model with a dynamic shape into an offline model with a fixed shape.</p> <p>Use commas (,) to separate multiple inputs.</p>
out_nodes	<p>Specifies the output node, for example, node_name1:0;node_name1:1;node_name2:0. node_name must be the node name in the network model before model conversion. The digit after each colon (:) indicates the sequence number of the output. For example, node_name1:0 indicates the 0th output of node_name1.</p>
input_format	<p>The default input data format is NHWC. If the actual format is NCHW, you need to set this parameter to NCHW.</p>
net_format	<p>Specifies the preferred data format for network operators. Possible values are ND (N cannot be more than 4) and 5D. This parameter only takes effect if the input data of operators on the network supports both ND and 5D formats. ND indicates that operators in the model are converted into the NCHW format. 5D indicates that operators in the model are converted into the Huawei-developed 5D format. 5D is the default value.</p>

Parameter	Description
fp16_high_prec	Specifies whether to generate a high-precision FP16 Davinci model. <ul style="list-style-type: none"> The default value is 0, indicating that a common FP16 Davinci model is generated, which has better inference performance. The value 1 indicates that a high-precision FP16 Davinci model with better inference precision is generated.
output_type	FP32 is the default value and is recommended for classification and detection networks. For image super-resolution networks, UINT8 is recommended for better inference performance.

After the model is imported, the **Models** page is displayed. You can view the imported model in the list.

Viewing Details About a Model

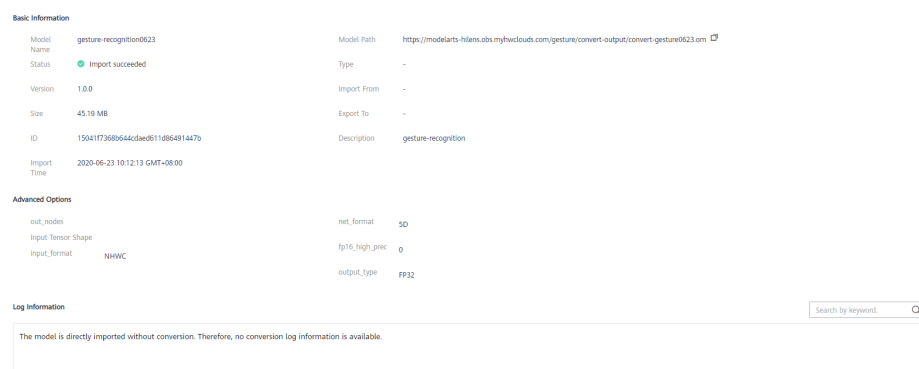
- Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Models**. The **Models** page is displayed.
You can view the **Model Name**, **Version**, **Model Size**, **Import Time**, **Status**, **Description**, and **Operation** in the list. The model status can be **Converting**, **Conversion failed**, **Conversion succeeded**, **Import failed**, and **Import succeeded**.
- Click **Details** in the **Operation** column of a model. The **Model Details** page is displayed.

You can view the **Basic Information** and **Log Information** of the model, as shown in [Figure 7-15](#).

[Table 7-7](#) describes the **Basic Information** parameters.

For the model that needs to be converted, you can enter keywords in the upper right corner of the **Log Information** area to quickly locate key information in a log.

Figure 7-15 Model details



Reconverting a Model



If the imported model is not in the **.om** format and is in **Conversion failed** status, you can modify the model parameters and convert it again.

You can view the model status on the **Models** page of the Huawei HiLens console.

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Models**. The **Models** page is displayed.
2. Click **Details** in the **Operation** column of a model. The **Model Details** page is displayed.

You can view the **Basic Information** and **Log Information** of the model.

For the model that needs to be converted, you can enter keywords in the upper right corner of the **Log Information** area to quickly locate key information in a log.

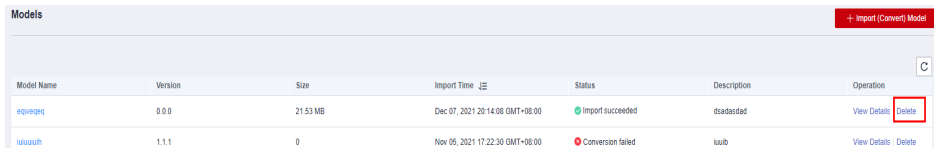
3. On the **Basic Information** tab page, click  to modify the **Type**, **Advanced Options**, **Import From**, **Export To**, and **Description** parameters. After the modification is complete, click  to save the modification. [Table 7-7](#) describes related parameters.
4. After modifying the model parameters, click **Reconvert** in the upper right corner to convert the model again.

Deleting a Model

You can delete imported (converted) models based on service requirements. Models in the **Converting** state cannot be deleted.

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Models**. The **Models** page is displayed.
2. Locate a model and click **Delete** in the **Operation** column to delete it. You can also click **Details** in the **Operation** column and click **Delete Model** in the upper right corner of the model details page to delete the model.

Figure 7-16 Deleting a model



Model Name	Version	Size	Import Time	Status	Description	Operation
dsadasd	0.0.0	21.53 MB	Dec 07, 2021 20:14:08 GMT+08:00	Import succeeded	dsadasd	View Details Delete
luub	1.1.1	0	Nov 05, 2021 17:22:30 GMT+08:00	Conversion failed	luub	View Details Delete

7.4.3 Model Input Directory Specifications

The input directory of a model (oriented to Ascend chips) developed locally or developed in ModelArts and converted on Huawei HiLens must meet certain specifications. Huawei HiLens has the following requirements on the model input directory:

- Input directory specifications for Caffe models:

---xxxx.caffemodel	Model parameter file, mandatory. Only one model parameter file can exist in the input directory.
---xxxx.prototxt	Model network file, mandatory. Only one model network file can exist in the input directory.
---insert_op_conf.cfg	Insertion operator configuration file, optional. Only one insertion operator

configuration file can exist in the input path.

|---plugin Custom operator directory, optional The input directory can contain only one plugin folder. Only custom operators developed based on Tensor Engine (TE) are supported.

- Input directory specifications for TensorFlow models (in **frozen_graph** or **saved_model** format):

frozen_graph format

|
|---xxx.pb Model network file, mandatory. Only one model network file can exist in the input path. The model must be in **frozen_graph** or **saved_model** format.

|---insert_op_conf.cfg Insertion operator configuration file, optional. Only one insertion operator configuration file can exist in the input path.

|---plugin Custom operator directory, optional The input directory can contain only one plugin folder. Only custom operators developed based on Tensor Engine (TE) are supported.

saved_model format

|
|---saved_model.pb Model network file, mandatory. Only one model network file can exist in the input path. The model must be in **frozen_graph** or **saved_model** format.

|---variables Fixed subdirectory name, including the model weight deviation, mandatory

|---variables.index Mandatory

|---variables.data-00000-of-00001 Mandatory

|---insert_op_conf.cfg Insertion operator configuration file, optional. Only one insertion operator configuration file can exist in the input path.

|---plugin Custom operator directory, optional The input directory can contain only one plugin folder. Only custom operators developed based on Tensor Engine (TE) are supported.

7.4.4 Model Output Directory Specifications

After a model import (conversion) task is executed, Huawei HiLens outputs the converted model to a specified OBS path. The output directories of models from different conversion tasks and oriented to Ascend chips must meet certain specifications. Huawei HiLens has the following requirements on the model output directory:

- Output directory specifications for Caffe models:

|
|---xxx.om Converted model to run on the Ascend chip. The model file name extension is **.om**.
|---job_log.txt Conversion log file

- Output directory specifications for TensorFlow models:

|
|---xxx.om Converted model to run on the Ascend chip. The model file name extension is **.om**.
|---job_log.txt Conversion log file

7.5 Writing the Code

Huawei HiLens provides a HiLens Framework, which encapsulates bottom-layer APIs to implement common management functions, enabling you to easily develop skills and the AI ecosystem. For details about the guide and APIs, see the HiLens Developer Guide.

HiLens Framework Sample Code

HiLens Kit devices run on the HiLens Framework. Developers can directly call APIs of the HiLens Framework when compiling logic code for skill development.

The following is an example of using the HiLens Framework to obtain the camera content and perform simple processing.

NOTE

- You need to fill in the code execution file during skill development.
- Firmware 1.1.2 or later supports configuring Python dependencies for skills. During skill development, you can configure Python dependencies for a skill as required. For details, see [How Do I Configure Python Dependencies for a Skill](#).

```
#!/usr/bin/python3
# skillframework 1.0.0 python demo

import hilens
import cv2

def main():
    hilens.init("hello")
    model = hilens.Model(hilens.get_model_dir() +
        "faceDetection.om")
    disp = hilens.Display(hilens.HDMI)
    cap = hilens.VideoCapture()
    proc = hilens.Preprocessor()

    hilens.set_log_level(hilens.DEBUG)
    hilens.Info("This is a skillframework python demo")

    for i in range(10):
        frame = cap.read()
        rframe = proc.resize(frame, 480, 480, 1)
        inputs = [rframe.flatten()]
        outputs = model.infer(inputs)
        hilens.Info("outputs len: %d" % len(outputs))
        for o in outputs:
            hilens.Info("output shape: " + str(o.shape))
        # post process is ignored
        disp.show(frame)

    hilens.terminate()

if __name__ == "__main__":
    main()
```

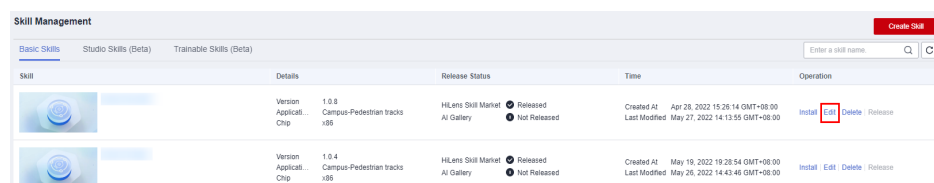
7.6 Editing Skills

During skill creation and use, if you find incorrect information, you can edit the skill's basic information, content, or runtime configurations to correct it.

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Skills**.
2. On the **Skills** page, locate the skill to be edited and click **Edit** in the **Operation** column to enter the **Edit Skill** page.

Modify the skill content based on parameters described in [Creating a Skill Using an Empty Template](#).

Figure 7-17 Editing skills



7.7 Installing and Debugging Skills

After a skill is created, you can install it on a device, and check its running effects by viewing the skill videos or original videos of the device, to determine whether the skill can meet service requirements.

You can also locate faults and debug the skill according to the device log. For detailed operations, see [Viewing Device Logs](#).

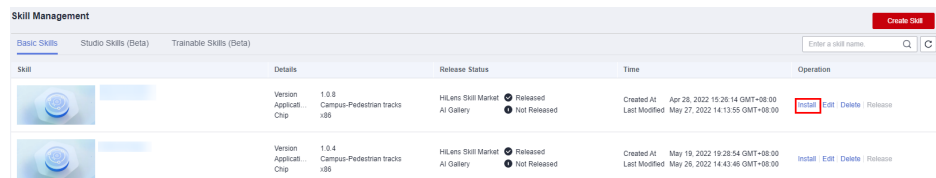
Prerequisites

- A HiLens Kit device has been registered with Huawei HiLens and is **Online**. For details about device management, see [Device Management Overview](#).
- A skill has been developed and **created** on Huawei HiLens.

Installing a Skill

1. Log in to the Huawei HiLens console. In the navigation pane on the left, choose **Skill Development > Skills**.
The **Basic Skills** tab page is displayed.
2. Select the skill to be installed and click **Install** in the **Operation** column.

Figure 7-18 Installing a skill



3. In the dialog box that is displayed, select the devices to which the skill is installed and click **OK**.

If the information similar to the following is displayed, the skill is installed successfully.

Figure 7-19 Successful installation

Install Skill Gesture_Recognition to Device

Device Name	Device status	Resource con...	Firmware Name	Firmware ...	Progress
<input checked="" type="radio"/> test	● Online	<input checked="" type="radio"/> Number of ch. <input type="radio"/> qps	HiLens_Device_Agent	1.0.9	✔ Deployment succeed...
<input type="radio"/> test1	● Offline	<input checked="" type="radio"/> Number of ch. <input type="radio"/> qps	HiLens_Device_Agent	1.0.7	-

5 Total Records: 2 < 1 >

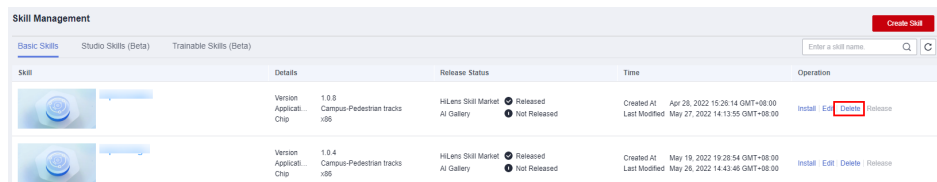
4. After the installation is successful, you can go to the **Data Management** page to view the skill effect.

7.8 Deleting Skills

You can delete skills that are no longer used to release resources.

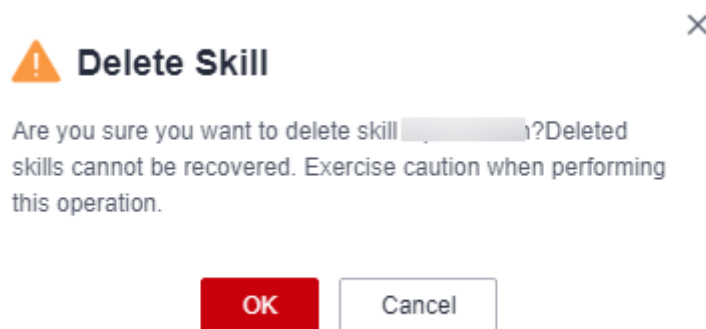
1. Log in to the Huawei HiLens console. In the navigation pane, choose **Skill Development > Skills**.
2. On the **Skills** page, choose **More > Delete** in the **Operation** column. The **Delete Skill** dialog box is displayed.

Figure 7-20 Deleting a skill



3. Confirm the information and click **OK**.

Figure 7-21 Deleting a skill-23



NOTE

Deleted skills cannot be recovered. Exercise caution when performing this operation.

8 Using the Skill Market

8.1 Skill Market Overview

After you register a device, you can purchase skills from the skill market of Huawei HiLens and install them to your device to enable AI capabilities. For example, if you install a facial recognition skill on your device, your device can recognize faces.

- You can search for and filter skills in the **Skill Market** to [find the skills you need](#).
- If the **Skill Market** has the skills you need, you can directly purchase the skills. For details, see [Purchasing Skills](#).
- You can manage the skills you purchased or released to the market on the **Subscribe > All Skills** page. For details, see [Managing Orders](#).

Skill

A skill is an AI application ready to run on a camera or another computing device. It consists of a model and logic code. The logic code is the skill's framework that governs how the skill behaves, including data reading, model import, model inference, and result output. The model is an AI algorithm trained using big data and handles inference while the skill is running.

The **Skill Market** of Huawei HiLens provides extensive skills for users.

- The application scenarios for HiLens skills include smart campus, smart home, smart vehicle-mounted terminals, smart shopping mall, and more.
- Skills are classified into two types by device: skills for Ascend chips and skills for HiSilicon Hi35xx series chips.

Figure 8-1 Skill card



Ascend310

Funny_Face_Effe...

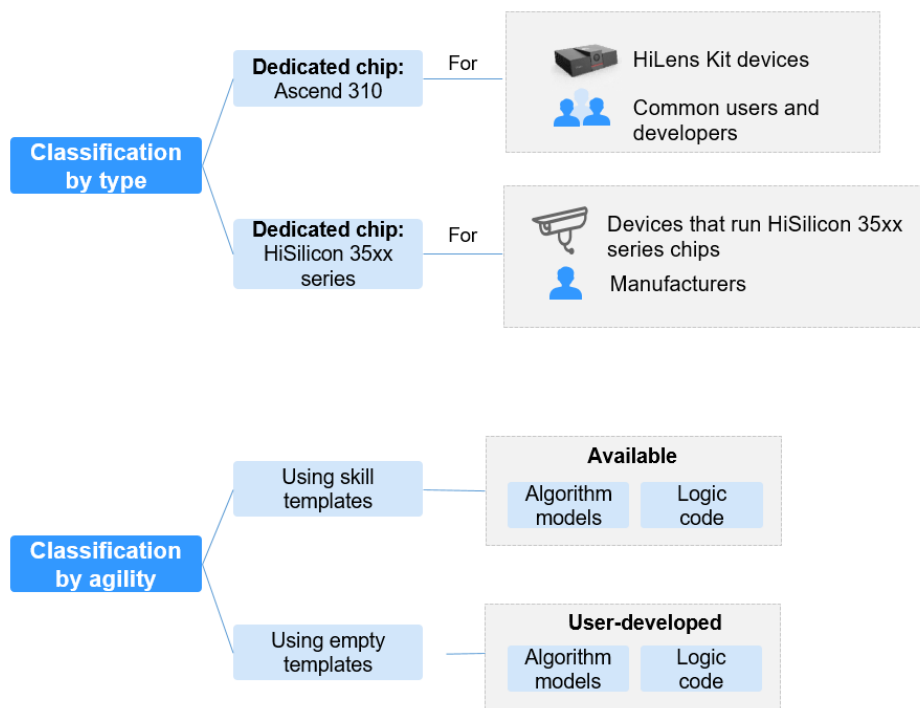
Add effects to your face. This skill will adds different items to your face which depends

Skill Classification

As shown in the following figure, Huawei HiLens skills are classified into two types based on the chip types: skills used on HiLens Kit devices and skills used on devices that run HiSilicon Hi35xx series chips. The Huawei HiLens platform supports quick, convenient, and efficient skill development with skill templates, and also allows you to develop models and logic code to meet your specific requirements in more scenarios.

HiSilicon Hi35xx series chips have low memory and performance. Therefore, you need to optimize models of skills oriented to this type of chips. If you want to develop skills of this type, contact Huawei HiLens platform personnel for support.

Figure 8-2 Classification of skill development scenarios



8.2 Searching for Skills in the Skill Market

The skill market is an open market that has preset skills, skills you released, and skills shared by other developers. You can search for skills in the following ways.

By default, skills in the skill market are sorted by a combination of various factors.

Searching for a Skill

In the upper left corner of the **Skill Market** page, you can enter a keyword of the skill name in the **Custom Filtering** text box to search for a skill.

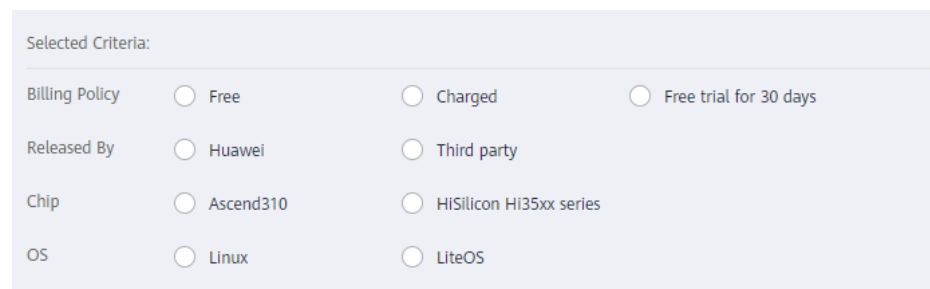
Figure 8-3 Search



Filtering Skills

In the **Selected Criteria** area, you can select **Billing Policy**, **Released By**, **Chip**, and **OS** options to filter skills. **Released By** can be **Huawei** and **Third party**. **Huawei** refers to the official skills released by HiLens. **Third party** refers to the skills developed by users on the Huawei HiLens console and released to the skill market.

Figure 8-4 Filtering skills



8.3 Purchasing Skills

Whether for device management or product management, you need to purchase an available skill in the skill market.

Context

- Skill purchase incurs certain fees. Before using Huawei HiLens, check your account status, which cannot be in arrears or frozen.
- Skills are classified into those **oriented to Ascend chips** and those **oriented to HiSilicon Hi35xx series chips**. Check the skill types carefully and purchase them based on site requirements.

Purchasing Skills from the Huawei HiLens Skill Market

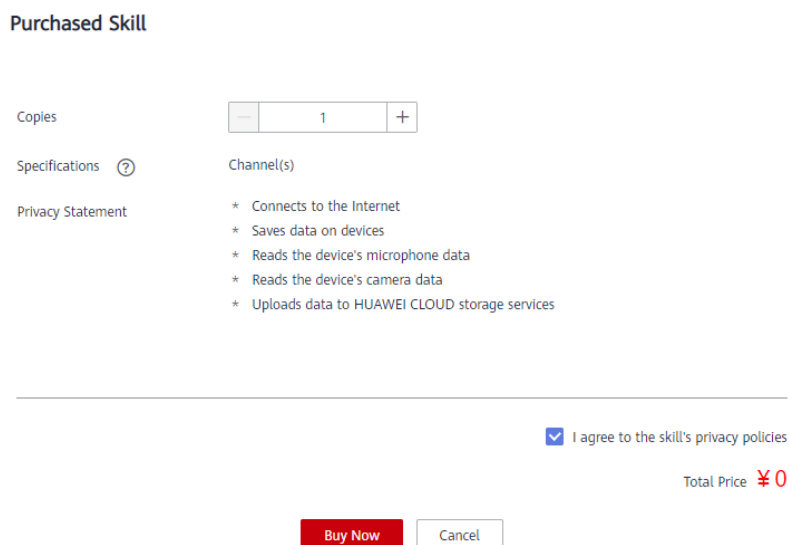
1. Log in to the Huawei HiLens console. In the navigation pane, choose **Subscribe > All Skills**. The **Skill Market** page is displayed.
2. Filter and search for the skill you need and click the skill card to access the skill details page.
3. On the skill details page, click **Buy Now**.
 - i. Specify **Copies** based on the **Specifications**, read the privacy policies, and select **I agree to the skill's privacy policies**. Confirm the price and click **Buy Now**.

The **Specifications** is **Channel(s)**, indicating the number of video channels that can be used by each license on the device. The measurement unit can be the **Number of channels** or **QPS**. If you select **Number of channels**, one license can be installed on a device to use one channel of video.

For some skills, you do not need to select **I agree to the skill's privacy policies**. In this case, directly click **Buy Now**.

- ii. After the dialog displays a message indicating that the purchase is complete, you can click **Purchased** to return to the **Skill Market** page. Alternatively, you can choose **Subscribe > Order Management** in the navigation pane to access the **Skill Orders** tab page, and view the purchased skill.

Figure 8-5 Purchasing a skill



Follow-up Operations

- Common users or developers (skill users): You can install purchased skills on your devices. For details, see [Installing Skills](#).
- Manufacturers (product managers): You can distribute purchased skills to your products. For details, see [Distributing Skills](#).

8.4 Managing Orders

You can manage skills you purchased from the Huawei HiLens skill market.

Skill Orders

Log in to the Huawei HiLens console. In the navigation pane, choose **Subscribe > Order Management**. The **Skill Orders** tab page is displayed by default. The **Skill Orders** tab page displays all the skills you purchased and the order details.

On the **Skill Orders** tab page, you can perform the following operations:

- **Install a skill**

Click **Install** in the **Operation** column to install the skill on a registered device. For detailed operations, see [Installing a Skill](#).

- **Expand the capacity of an order**

- a. Click **Expand Capacity** in the **Purchase Details** column to increase the number of copies.
- b. On the **Purchase Skill** page, specify **Quantity**, select **I agree to the Privacy Statement and the Declaration**, and click **Buy Now** in the lower right corner.
- c. On the **Confirm** page, confirm the order details and click **Submit** in the lower right corner.
- d. On the **Pay** page, select a payment method and click **Pay** in the lower right corner.

Capacity expansion is supported only for orders that use the yearly/monthly billing mode.

- **Distribute a skill**

Click **Operation** in the **Operation** column to expand the drop-down list, and click **Distribute** to distribute the skill to products. For detailed operations, see [Distributing Skills](#).

- **Download a skill**

Click **Operation** in the **Operation** column to expand the drop-down list, and click **Download**. The browser automatically downloads the SDK package of the selected skill.

9 Managing Data

After a skill is delivered to a device, you can download the device data on the Huawei HiLens console for analysis purposes, for example, to check how the skill is performing on the device where it is currently deployed.


NOTE

- Only the data of HiLens Kit devices can be managed on the **Data Management** page. The data of devices running HiSilicon Hi35xx series chips cannot be managed on the Huawei HiLens console.
- You can only use the data management function to view data of skills that are exported to OBS as files.

Prerequisites

- A common user successfully installs a purchased skill to a device. For details, see [Installing a Skill](#).
- A developer successfully installs a skill to a device. For details, see [Installing and Debugging Skills](#).
- Only when the output module of a skill defines the scenario that data is output to OBS, you can view the skill data on the **Data Management** page.
 - If you are a skill developer and need to view the running effect of your skill on devices, you need to specify that the skill data is stored on OBS, so that you can download the data from the HiLens console.
 - If you are a skill user, you need to check whether the data of a skill is stored on OBS on the skill details page in the skill market.

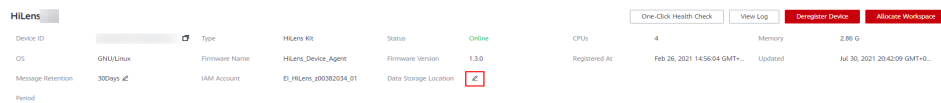
Configuring the Data Storage Location (OBS Storage Path)

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Device Management > Devices**. In the device list, click the name of the device to be managed to enter the device details page.
2. Click  on the right of **Data Storage Location**. In the dialog box that is displayed, select an OBS directory to store the video data of the current device.

NOTE

- Storing device data consumes storage resources of Huawei Cloud. Using OBS for storage will incur fees. For detailed operations and specifications, see the [Object Storage Service Console Operation Guide](#).
- When a URL is used to access a bucket, the bucket name is a part of the URL. According to DNS standards, a URL does not support uppercase letters and cannot be used to access a bucket whose name contains uppercase letters. Therefore, a bucket name can contain only lowercase letters, digits, hyphens (-), and periods (.). It cannot contain two consecutive periods (.) or contain a period (.) and a hyphen (-) adjacent to each other. For details about the naming rules, see [Creating a Bucket](#).

Figure 9-1 Configuring the data storage location



Viewing Data

You can view the device data and skill running effect based on the data output mode. For details about the skill output modes, see [Table 9-1](#).

NOTE

- Video data of most skills is output to a display through the HDMI port of a device, and little video data is output to a server for viewing. Go to the **Skill Market** and view the output settings of a skill under **Product Description** on the skill details page. For example, if the description of a celebrity recognition skill is "View via HDMI", you can view the skill running effect on a display after the skill is deployed.
- Skills of other output modes (excluding **HDMI**) are stored on OBS. For details about how to view data, see [Configuring the Data Storage Location \(OBS Storage Path\)](#).

Table 9-1 Skill output modes

Mode	Data Viewing Description
HDMI	The video data is directly output to a display through the HDMI port of the device.
RTMP	The video data is output to a server in real time for users to view. For details about the server address, see the skill's Product Description .
H264_FILE	The video data is output to OBS as files for users to view. For details about the OBS path, see Configuring the Data Storage Location (OBS Storage Path) .

Manually Collecting Logs

If the firmware version of the current device is 1.3.3 or later, you need to manually collect logs.

1. Log in to the Huawei HiLens management console. In the navigation pane, choose **Data Management (Beta)**. The **Data Management** page is displayed.
2. Click the device that failed to upload logs and click **Save All Logs** in the upper right corner.
A dialog box for uploading logs is displayed.
3. Select the device logs you want to upload to OBS and click **OK**.
Wait for log collection.

 **NOTE**

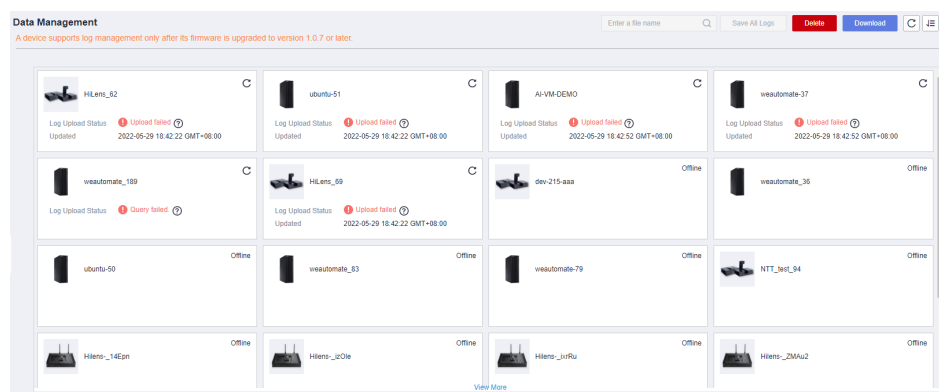
The system will collect the device logs you selected. This typically takes 5 minutes.

4. After the log collection is complete, on the **Data Management** page, click the device card to go to the corresponding folder and download the log file package to the local PC to view the logs.

Downloading Data

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Data Management (Beta)**. The **Data Management** page is displayed.
2. Double-click the specified device name to access the device data folder to be downloaded.

Figure 9-2 Data Management

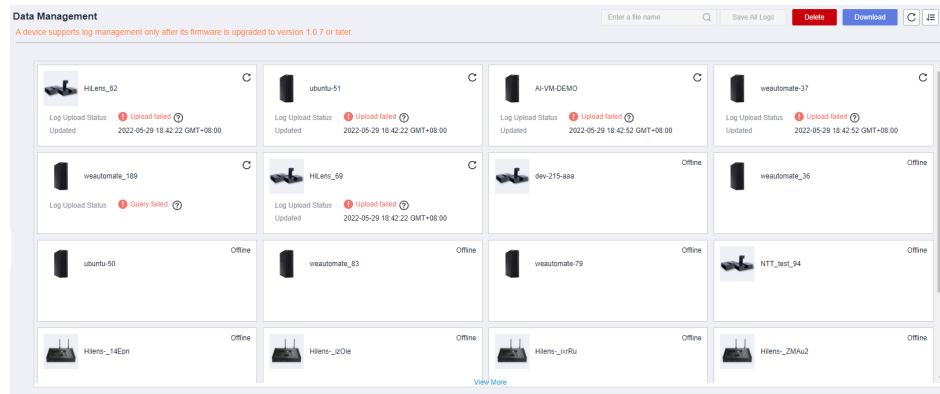


3. Select the data folder and click **Download** to download the data to your local PC.

Deleting Data

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Data Management (Beta)**. The **Data Management** page is displayed.
2. Double-click the specified device name to access the device data folder to be deleted.

Figure 9-3 Data Management



3. Select the data folder and click **Delete**.

10 Managing Products

10.1 Overview

As the manufacturers of cameras running HiSilicon Hi35xx series chips, you can use Huawei HiLens to manage products in batches. You can also purchase skills from the skill market, and distribute the skills to your products to enable your products with AI capabilities.

- [Managing Products](#)
- [Purchasing Skills \(for Devices Running Hi35xx Series Chips\)](#)
- [Distributing Skills](#)
- [Adding or Deleting Skills](#)

10.2 Managing Products

After a manufacturer creates a product on the Huawei HiLens console, the manufacturer can associate devices with the product. Once a user who purchases a device registers the device, the manufacturer can manage the registered device and its skills on the Huawei HiLens console.

Creating a Product

1. Log in to the Huawei HiLens console, and choose **Device Management** > **Products** in the navigation pane. The **Products** page is displayed.
2. Click **Create Product** in the upper right corner and set product parameters, as shown in [Table 10-1](#).

Figure 10-1 Creating a product

Table 10-1 Product parameters

Parameter	Description
Name	Product name. Enter 3 to 60 characters, starting with a letter and ending with a letter or digit. Only letters, digits, underscores (_), and hyphens (-) are allowed.
Platform	Device OS type. Android, Linux, iOS, LiteOS, and Windows are supported.
Chip	Device chip model: HiSilicon Hi35xx series, including 3516CV500, 3519AV100, 3519V101, 3516DV300, 3516EV200, 3516EV300, 3518EV300, and Arm. When the chip model is 3518EV300, 3516EV200, or 3516EV300, you can select 3516CV500 .
Description	Product description. The value can contain a maximum of 512 characters.

3. Confirm the product information and click **OK**.
After the product is created, the **Products** page is automatically displayed.

Downloading a License

After a product is created, you need to download the corresponding license and save it to the specified path on the device.

1. Choose **Device Management > Products** in the navigation pane. The **Products** page is displayed.
2. Select the target product and click **Download** in the **Operation** column to download the license file.

Figure 10-2 Downloading a license

Name	ID	Chip	Description	Platform	Operation
		3516CV500	hilens	Linux	Edit Download Add Skill Delete
		3516CV500	hilens	Linux	Edit Download Add Skill Delete
		3516CV500	hilens	Linux	Edit Download Add Skill Delete
		3516CV500	hilens	Linux	Edit Download Add Skill Delete

3. Place the license file in the specified directory on the device.
The location for storing the license varies according to the device manufacturer. After the cooperation relationship is established between you and the manufacturer, you can submit a service ticket, and the Huawei HiLens team will provide guidance. You can also submit a service ticket to learn more about the skills for HiSilicon Hi35xx series chips.

Editing a Product

1. Log in to the Huawei HiLens console, and choose **Device Management > Products** in the navigation pane. The **Products** page is displayed.
2. Select the target product and click **Edit** in the **Operation** column. The product editing dialog box is displayed.

Figure 10-3 Editing a product

Name ×

Name

* Platform ?

* Chip ?

* Description

3. In the displayed dialog box, edit the **Name**, **Platform**, **Chip**, and **Description** based on [Table 10-1](#).
4. Confirm the product information and click **OK**.

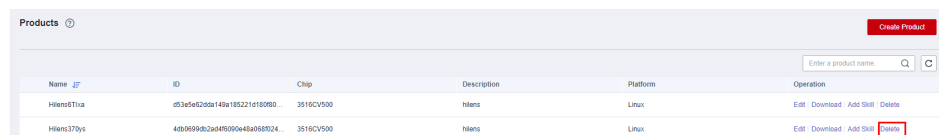
Deleting a Product

NOTICE

Before deleting a product, ensure that the product is not in use. Exercise caution when performing this operation. If the product is associated with devices, the devices cannot be managed after the product is deleted. If users have registered the devices, the manufacturer cannot distribute skills to the devices.

1. Log in to the Huawei HiLens console, and choose **Device Management > Products** in the navigation pane. The **Products** page is displayed.
2. Select the target product and click **Delete** in the **Operation** column.

Figure 10-4 Deleting a product



Name	ID	Chip	Description	Platform	Operation
HiLens370a	ed3e4e6206a149a185221c110089...	3516CV500	HiLens	Linux	Edit Download Add Skill Delete
HiLens370b	4db0999db2a4f5090a4ba088924...	3516CV500	HiLens	Linux	Edit Download Add Skill Delete

3. Confirm the product information and click **OK**.

10.3 Purchasing Skills (for Devices Running Hi35xx Series Chips)

You can purchase required skills in the skill market of Huawei HiLens to develop devices running HiSilicon Hi35xx series chips. The applicable chips include 3516CV500, 3519AV100, 3519V101, 3516DV300, 3516EV200, 3516EV300, 3518EV300, and Arm.

If the **Skill Market** has the skills you need, you can directly purchase the skills.

- Purchasing skills: In the skill market, select the skills oriented to **HiSilicon Hi35xx series chips**. For details, see [Purchasing Skills](#).

10.4 Distributing Skills

Skills oriented to HiSilicon Hi35xx series chips can be used only after being distributed to devices. For a device that runs HiSilicon Hi35xx series chips, a maximum of five skills can be distributed to it due to performance restrictions.

Prerequisites

- Devices have been associated with a product. For details, see [Managing Products](#).
- Skills have been purchased. For details, see [Purchasing Skills \(for Devices Running Hi35xx Series Chips\)](#).

Distributing a Skill

1. Log in to the Huawei HiLens console. In the navigation pane, choose **Subscribe > Order Management**. The **Skill Orders** tab page is displayed by default.
2. Select the skill you need and distribute the skill license to a product.
In the skill information, only the skills whose **Chip** is not **Ascend 310** can be distributed to products.
 - a. Select the skill to be distributed and click **Distribute** in the **Operation** column.

Figure 10-5 Distributing a skill

Skill	Platform	Quantity	Chip	Operation
...	Ascend 310	2	Ascend 310	...
...	Ascend 310	10	Ascend 310	...
...	Ascend 310	10	Ascend 310	...
...	Ascend 310	4	Ascend 310	...
...	Ascend 310	4	Ascend 310	...
...	Ascend 310	1	Ascend 310	...
...	Ascend 310	1	Ascend 310	...

- b. In the **Distribute to Product** dialog box, select the product to which the license is to be distributed, select or not select **Yes** under **Automatic Upgrade** as required, and click **Distribute to Product**.

If you select **Automatic Upgrade**, the skill in the product will be automatically upgraded to the latest version if a new version is available for the skill.

Figure 10-6 Distribute to Thin Product

X

Distribute to Thin Product

Name	Description	Platform	Chip	Automatic Upgrade
No data available				








5 Total Records: 0 < 1 >

Distribute to Product
Cancel

- c. The skill distribution information is displayed in the dialog box. Confirm that the installation status is successful and click **OK**.
3. Download the skill.

On the **Skill Orders** tab page, select the skill you want to use and click **Download** in the **Operation** column. The browser automatically downloads the SDK package of the skill.

Figure 10-7 Download a skill

Skill	Purchase Details	Order Information	Operation
 Platform: Linux Chip: Ascend 310	Remaining Quantity: 2 Capacity: 2 Measurement Unit: CHANNEL1 Maximum Channels: 8	Order ID: 2021-14-06-47-04M1+00-00 Order Time: 2021-14-06-14:06:47 Order Fee: 0.00 Expiration Time: 2021-14-06-14:06:47	Yearly/Monthly Change Type: Change Purchase At: 0.00 Order Fee: 0.00 Expiration Time: 2021-14-06-14:06:47
 Platform: Linux Chip: Ascend 310	Remaining Quantity: 10 Capacity: 10 Measurement Unit: CHANNEL1 Maximum Channels: 10	Order ID: 2021-14-07-38-04M1+00-00 Order Time: 2021-14-07-18:38:39 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39	One-time Pay Change Type: Buy Purchase At: 0.00 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39
 Platform: Linux Chip: Ascend 310	Remaining Quantity: 10 Capacity: 10 Measurement Unit: CHANNEL1 Maximum Channels: 10	Order ID: 2021-14-07-38-04M1+00-00 Order Time: 2021-14-07-18:38:39 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39	One-time Pay Change Type: Buy Purchase At: 0.00 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39
 Platform: Linux Chip: Ascend 310	Remaining Quantity: 4 Capacity: 4 Measurement Unit: CHANNEL1 Maximum Channels: 10	Order ID: 2021-14-07-38-04M1+00-00 Order Time: 2021-14-07-18:38:39 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39	Yearly/Monthly Change Type: Change Purchase At: 0.00 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39
 Platform: Linux Chip: Ascend 310	Remaining Quantity: 1 Capacity: 1 Measurement Unit: CHANNEL1 Maximum Channels: 10	Order ID: 2021-14-07-38-04M1+00-00 Order Time: 2021-14-07-18:38:39 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39	Yearly/Monthly Change Type: Change Purchase At: 0.00 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39
 Platform: Linux Chip: Ascend 310	Remaining Quantity: 1 Capacity: 1 Measurement Unit: CHANNEL1 Maximum Channels: 10,000	Order ID: 2021-14-07-38-04M1+00-00 Order Time: 2021-14-07-18:38:39 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39	One-time Pay Change Type: Buy Purchase At: 0.00 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39
 Platform: ARMV8 Chip: Ascend 310	Remaining Quantity: 1 Capacity: 1 Measurement Unit: CHANNEL1 Maximum Channels: 99	Order ID: 2021-14-07-38-04M1+00-00 Order Time: 2021-14-07-18:38:39 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39	One-time Pay Change Type: Buy Purchase At: 0.00 Order Fee: 0.00 Expiration Time: 2021-14-07-18:38:39

4. Deploy the skill. Integrate the SDK package downloaded in 3 to a device. The device location where the skill is integrated varies according to the device. For details about the operation guide, contact Huawei technical engineers.

10.5 Adding or Deleting Skills

Distributing skills refers to distributing skills from the skill market to products. Adding skills refers to adding skills for specified products on the **Products** page. The final results of the two operations are the same.

You can delete skills that are no longer used from the current product.

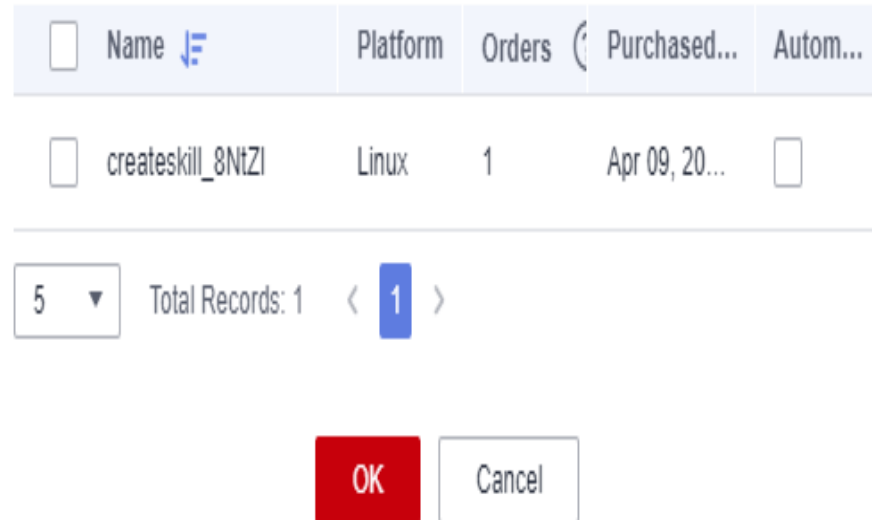
Adding a Skill

1. Log in to the Huawei HiLens console, and choose **Device Management** > **Products** in the navigation pane. The **Products** page is displayed.
2. Select a product and click **Add Skill** in the **Operation** column.
3. The **Available Skills** dialog box displays a list of purchased skills by default. Select the skills to be added, select **Automatic Upgrade** as required, and click **OK**.

If you select **Automatic Upgrade**, the skill will be automatically upgraded to the latest version if a new version is available for the skill.

Figure 10-8 Adding a skill

Available skills



Deleting a Skill

Deleting a skill is to remove the skill from a device.

NOTE

- After devices that support automatic skill update (devices running the 3516CV500 and 3519AV100 chips) detect that the skills on the cloud are deleted, the devices also delete the device-side skills.
- For devices that do not support automatic skill update, after the skills are deleted from the cloud, the devices are not affected and the device-side skills can still be used. If you want to delete the device-side skills, submit service tickets.


1. Log in to the Huawei HiLens console, and choose **Device Management** > **Products** in the navigation pane. The **Products** page is displayed.
2. Select a product and click the arrow  on the left to view all skills under the product.
3. Select the skill to be deleted and click **Delete**.

Figure 10-9 Deleting a skill

Name	ID	Chip	Description	Platform	Operation
mmmm	8a430895204a3aa03054600e0...	Arm	mmmm	Linux	Edit Download Add Skill Delete
ff	7cc2621f044a4e0069fce692385...	Arm	ed	Linux	Edit Download Add Skill Delete
wwwwwww	bc9d76e9ee04c3da47a59c4e21...	3518EV300	wwwwwww	LiteOS	Edit Download Add Skill Delete
ssssss	445c4483e3364463815c08a3909...	3518EV300	www	Android	Edit Download Add Skill Delete
hhhhh	4a89fc34d0504e3987302e66310...	Arm	ff	Android	Edit Download Add Skill Delete

4. Confirm the information of the skill to be deleted and click **OK**.

A Caffe Operator Boundaries

For the Caffe framework, if the input dimension of each operator is not 4 and the **axis** parameter exists, negative numbers cannot be used.

Table A-1 shows the boundaries of Caffe operators supported by **.om** models.

Table A-1 Caffe operator boundaries

No.	Operator	Definition	Boundary
1	Absval	Computes the absolute value of the input.	[Input] One input [Parameter] engine: (optional) enum, default to 0 , CAFFE = 1, CUDNN = 2 [Constraint] None [Quantitative tool support] Yes
2	Argmax	Returns the index number corresponding to the maximum input value.	[Input] One input [Parameter] <ul style="list-style-type: none"> • out_max_val: (optional) bool, default to false • top_k: (optional) unit32, default to 1 • axis: (optional) int32 [Constraint] None [Quantitative tool support] No

No.	Operator	Definition	Boundary
3	BatchNorm	Normalizes the input: variance of $[(x - \text{avg}(x))/x]$	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • use_global_stats: bool, must be true • moving_average_fraction: (optional) float, default to 0.999 • eps: (optional) float, default to 1e-5 <p>[Constraint] Only the C dimension can be normalized.</p> <p>[Quantitative tool support] Yes</p>
4	Concat	Concatenates the input along the given dimension.	<p>[Input] Multiple inputs</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • concat_dim: (optional) uint32, default to 1, greater than 0 • axis: (optional) int32, default to 1, exclusive with concat_dim. When axis is -1, four input dimensions are required. Otherwise, the result may be incorrect. <p>[Constraint]</p> <ul style="list-style-type: none"> • For the input tensor, the sizes of its dimensions must be the same except the dimension for concatenation. • The range of the input tensor count is [1, 1,000]. <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
5	DepthwiseConvolution	Depthwise convolution	<p>[Input] One 4D input, with a constant filter</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • num_output: (optional) uint32 • bias_term: (optional) bool, default to true • pad: uint32, default to 0, array • kernel_size: uint32, array • stride: uint32, default to 1, array • dilation: uint32, default to 1, array • pad_h: (optional) uint32, default to 0 (2D only) • pad_w: (optional) uint32, default to 0 (2D only) • kernel_h: (optional) uint32 (2D only) • kernel_w: (optional) uint32 (2D only) • stride_h: (optional) uint32 (2D only) • stride_w: (optional) uint32 (2D only) • group: (optional) uint32, default to 1 • weight_filler: (optional) FillerParameter • bias_filler: (optional) FillerParameter • engine: (optional) enum, default to 0, CAFFE = 1, CUDNN = 2 • force_nd_im2col: (optional) bool, default to false • axis: (optional) int32, default to 1 <p>[Constraint] filterN=inputC=group</p> <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
6	Convolution	Convolution	<p>[Input] One 4D input, with a constant filter</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • num_output: (optional) uint32 • bias_term: (optional) bool, default to true • pad: uint32, default to 0, array • kernel_size: uint32, array • stride: uint32, default to 1, array • dilation: uint32, default to 1, array • pad_h: (optional) uint32, default to 0 (2D only) • pad_w: (optional) uint32, default to 0 (2D only) • kernel_h: (optional) uint32 (2D only) • kernel_w: (optional) uint32 (2D only) • stride_h: (optional) uint32 (2D only) • stride_w: (optional) uint32 (2D only) • group: (optional) uint32, default to 1 • weight_filler: (optional) FillerParameter • bias_filler: (optional) FillerParameter • engine: (optional) enum, default to 0, CAFFE = 1, CUDNN = 2 • force_nd_im2col: (optional) bool, default to false • axis: (optional) int32, default to 1 <p>[Constraint]</p> <ul style="list-style-type: none"> • $(inputW + padWHead + padWTail) \geq (((FilterW-1) \times dilationW) + 1)$ • $(inputW + padWHead + padWTail)/StrideW + 1 \leq 2147483647$ • $(inputH + padHHead + padHTail) \geq (((FilterH-1) \times dilationH) + 1)$

No.	Operator	Definition	Boundary
			<ul style="list-style-type: none"> • $(inputH + padHHead + padHTail) / StrideH + 1 \leq 2147483647$ • $0 \leq Pad < 256, 0 < FilterSize < 256, 0 < Stride < 64, 1 \leq dilationsize < 256$ [Quantitative tool support] Yes
7	Crop	Crops the input.	[Input] Two inputs [Parameter] <ul style="list-style-type: none"> • axis: (optional) int32, default to 2. When axis is -1, four input dimensions are required. • offset: uint32, array [Constraint] None [Quantitative tool support] No

No.	Operator	Definition	Boundary
8	Deconvolution	Deconvolution	<p>[Input]</p> <p>One 4D input, with a constant filter</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • num_output: (optional) uint32 • bias_term: (optional) bool, default to true • pad: uint32, default to 0, array • kernel_size: uint32, array • stride: uint32, default to 1, array • dilation: uint32, default to 1, array • pad_h: (optional) uint32, default to 0 (2D only) • pad_w: (optional) uint32, default to 0 (2D only) • kernel_h: (optional) uint32 (2D only) • kernel_w: (optional) uint32 (2D only) • stride_h: (optional) uint32 (2D only) • stride_w: (optional) uint32 (2D only) • group: (optional) uint32, default to 1 • weight_filler: (optional) FillerParameter • bias_filler: (optional) FillerParameter • engine: (optional) enum, default to 0, CAFFE = 1, CUDNN = 2 • force_nd_im2col: (optional) bool, default to false • axis: (optional) int32, default to 1 <p>[Constraint]</p> <ul style="list-style-type: none"> • group = 1 • dilation = 1 • filterH - padHHead - 1 ≥ 0 • filterW - padWHead - 1 ≥ 0 <p>Restrictions involving intermediate variables:</p>

No.	Operator	Definition	Boundary
			<ol style="list-style-type: none"> 1. $a = \text{ALIGN}(\text{filter_num}, 16) \times \text{ALIGN}(\text{filter_c}, 16) \times \text{filter_h} \times \text{filter_w} \times 2$ 2. If $\text{ALIGN}(\text{filter_c}, 16) \% 32 = 0$, $a = a/2$ 3. $\text{conv_input_width} = (\text{deconvolution input } W - 1) \times \text{strideW} + 1$ 4. $b = (\text{conv_input_width}) \times \text{filter_h} \times \text{ALIGN}(\text{filter_num}, 16) \times 2 \times 2$ 5. $a + b \leq 1024 \times 1024$ <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Operator	Definition	Boundary
9	Detection Output	Generates detection results and outputs FSR.	<p>[Input] Three inputs</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● num_classes: (mandatory) int32, indicating the number of classes to be predicted ● share_location: (optional) bool, default to true, indicating that classes share one bounding box ● background_label_id: (optional) int32, default to 0 ● nms_param: (optional) indicating non-maximum suppression (NMS) ● save_output_param: (optional) indicating whether to save the detection result ● code_type: (optional) default to CENTER_SIZE ● variance_encoded_in_target: (optional) bool, default to true. The value true indicates that the variance is encoded in the target, otherwise the prediction offset needs to be adjusted accordingly. ● keep_top_k: (optional) int32, indicating the total number of BBoxes to be reserved for each image after NMS ● confidence_threshold: (optional) float, indicating that only the detection whose confidence is above the threshold is considered. If this parameter is not set, all boxes are considered. ● nms_threshold: (optional) float ● top_k: (optional) int32 ● boxes: (optional) int32, default to 1 ● relative: (optional) bool, default to true ● objectness_threshold: (optional) float, default to 0.5 ● class_threshold: (optional) float, default to 0.5 ● biases: array

No.	Operator	Definition	Boundary
			<ul style="list-style-type: none"> • general_nms_param: optional [Constraint] • Used for Faster R-CNN • Non-maximum suppression (NMS) ratio nmsThreshold is within (0, 1) • Probability threshold postConfThreshold is within (0, 1) • Classes ≥ 2 • Input box count $\leq 1,024$ • Output W dimension = 16 [Quantitative tool support] Yes
10	Eltwise	Computes element-wise operations (PROD, MAX, and SUM).	[Input] At least two inputs [Parameter] <ul style="list-style-type: none"> • operation: (optional) enum, PROD = 0, SUM = 1, MAX = 2; default to SUM • coeff: float array • stable_prod_grad: (optional) bool, default to true [Constraint] <ul style="list-style-type: none"> • Up to four inputs • Compared with the native operator, this operator does not support the stable_prod_grad parameter. • PROD, MAX, and SUM operations are supported. [Quantitative tool support] Yes
11	Elu	Activation function	[Input] One input [Parameter] alpha : (optional) float, default to 1 [Constraint] None [Quantitative tool support] No

No.	Operator	Definition	Boundary
12	Exp	Applies e as the base and x as the exponent.	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • base: (optional) float, default to -1.0 • scale: (optional) float, default to 1.0 • shift: (optional) float, default to 0.0 <p>[Constraint] None</p> <p>[Quantitative tool support] No</p>
13	Flatten	Converts an input of shape $N * C * H * W$ to a vector output of shape $N * (C * H * W)$.	<p>[Input] One input (top_size \neq bottom_size \neq 1. When axis is -1, four input dimensions are required.)</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • axis: (optional) int32, default to 1 • end_axis: (optional) int32, default to -1 <p>[Constraint] axis < end axis</p> <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
14	FullConnection	Computes an inner product.	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● num_output: (optional) uint32 ● bias_term: (optional) bool, default to true ● weight_filler: (optional) FillerParameter, 2D ● bias_filler: (optional) FillerParameter, 1D ● axis: (optional) int32, default to 1 ● transpose: (optional) bool, default to false <p>[Constraint]</p> <ul style="list-style-type: none"> ● transpose = false, axis = 1 ● Bias_C ≤ 56832 ● To quantify the model, the following dimension restrictions must be satisfied: <ul style="list-style-type: none"> – When N = 1, then 2 x CEIL(C, 16) x 16 x xH x xW ≤ 1024 x 1024 – When N > 1, then 2 x 16 x CEIL(C, 16) x 16 x xH x xW ≤ 1024 x 1024 <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
15	Interp	Interpolation layer	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • height: (optional) int32, default to 0 • width: (optional) int32, default to 0 • zoom_factor: (optional) int32, default to 1 • shrink_factor: (optional) int32, default to 1 • pad_beg: (optional) int32, default to 0 • pad_end: (optional) int32, default to 0 <p>NOTE zoom_factor and shrink_factor are exclusive. height and zoom_factor are exclusive. height and shrink_factor are exclusive.</p> <p>[Constraint] $(\text{outputH} \times \text{outputW}) / (\text{inputH} \times \text{inputW}) > 1/30$</p> <p>[Quantitative tool support] No</p>
16	Log	Performs logarithmic operation on the input.	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • base: (optional) float, default to -1.0 • scale: (optional) float, default to 1.0 • shift: (optional) float, default to 0.0 <p>[Constraint] None</p> <p>[Quantitative tool support] No</p>

No.	Operator	Definition	Boundary
17	LRN	Normalizes the input in a local region.	<p>[Input] One non-constant input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● local_size: (optional) uint32, default to 5 ● alpha: (optional) float, default to 1 ● beta: (optional) float, default to 0.75 ● norm_region: (optional) enum, default to ACROSS_CHANNELS (ACROSS_CHANNELS = 0, WITHIN_CHANNEL = 1) ● lrnk: (optional) float, default to 1 ● engine: (optional) enum, default to 0, CAFFE = 1, CUDNN = 2 <p>[Constraint]</p> <ul style="list-style-type: none"> ● local_size is an odd number greater than 0. ● Inter-channel: If local_size is within [1, 15]: $\lnK > 0.00001$ and $\beta > 0.01$; Otherwise, lrnk and beta are any values. lrnk and alpha are not 0 at the same time. When the C dimension is greater than 1,776, $\text{local_size} < 1728$. ● Intra-channel: $\lnK = 1$, local_size is within [1, 15], $\beta > 0.01$. <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
18	LSTM	Long and short term memory network (LSTM)	<p>[Input]</p> <p>Two or three inputs</p> <ul style="list-style-type: none"> • X: time sequence data ($T \times B \times X_t$), which is in the NCHW 4D format, • where, N corresponds to the time sequence length T, C corresponds to the batch size B, H corresponds to the input data X_t at time point t, and W is fixed at 1. • Cont: sequence continuity flag ($T \times B$) • Xs: (optional) static data ($B \times X_t$) <p>[Parameter]</p> <ul style="list-style-type: none"> • num_output: (optional) uint32, default to 0 • weight_filler: (optional) FillerParameter • bias_filler: (optional) FillerParameter • debug_info: (optional) bool, default to false • expose_hidden: (optional) bool, default to false <p>[Constraint]</p> <p>Restrictions involving intermediate variables:</p> $a = (\text{ALIGN}(x_t, 16) + \text{ALIGN}(\text{output}, 16)) \times 16 \times 2 \times 2$ $b = (\text{ALIGN}(x_t, 16) + \text{ALIGN}(\text{output}, 16)) \times 16 \times 4 \times 2 \times 2$ $c = \text{use_projection} ? \text{ALIGN}(h_t, 16) \times \text{ALIGN}(\text{output}, 16) \times 2 : 0$ $d = 16 \times \text{ALIGN}(h_t, 16) \times 2$ $e = \text{batchNum} \times 4$ <p>The constraints are as follows:</p> $a + b + c \leq 1024 \times 1024$ $d \leq 256 \times 1024/8$ $e \leq 256 \times 1024/32$ <p>[Quantitative tool support]</p> <p>No</p>

No.	Operator	Definition	Boundary
19	Normalize	Normalization layer	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • across_spatial: (optional) bool, default to true • scale_filler: (optional) default to 1.0 • channel_shared: (optional) bool, default to true • eps: (optional) float, default to 1e-10 <p>[Constraint]</p> <ul style="list-style-type: none"> • $1e - 7 < eps \leq 0.1 + (1e - 6)$ • across_spatial can only be true for Caffe, indicating normalization by channel. <p>[Quantitative tool support] Yes</p>
20	Permute	Permutates the input dimensions according to a given mode.	<p>[Input] One input</p> <p>[Parameter] order: uint32, array</p> <p>[Constraint] None</p> <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
21	Pooling	Pools the input.	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● pool: (optional) enum, indicating the pooling method, MAX = 0, AVE = 1, and STOCHASTIC = 2, default to MAX ● pad: (optional) uint32, default to 0 ● pad_h: (optional) uint32, default to 0 ● pad_w: (optional) uint32, default to 0 ● kernel_size: (optional) uint32, exclusive with kernel_h/kernel_w ● kernel_h: (optional) uint32 ● kernel_w: (optional) uint32, used in pair with kernel_h ● stride: (optional) uint32, default to 1 ● stride_h: (optional) uint32 ● stride_w: (optional) uint32 ● engine: (optional) enum, default to 0, CAFFE = 1, CUDNN = 2 ● global_pooling: (optional) bool, default to false ● ceil_mode: (optional) bool, default to true ● round_mode: (optional) enum, CEIL = 0, FLOOR = 1, default to CEIL <p>[Constraint]</p> <ul style="list-style-type: none"> ● $\text{kernelH} \leq \text{inputH} + \text{padTop} + \text{padBottom}$ ● $\text{kernelW} \leq \text{inputW} + \text{padLeft} + \text{padRight}$ ● $\text{padTop} < \text{windowH}$ ● $\text{padBottom} < \text{windowH}$ ● $\text{padLeft} < \text{windowW}$ ● $\text{padRight} < \text{windowW}$ <p>In addition to common restrictions, the following restrictions must be satisfied.</p>

No.	Operator	Definition	Boundary
			<p>The global pool mode supports only the following ranges:</p> <ol style="list-style-type: none"> outputH==1 && outputW==1 && kernelH>=inputH && kernelW>=inputW inputH*inputW ≤ 10,000 <p>[Quantitative tool support] Yes</p>
22	Power	Computes the output y as (scale * x + shift)^power.	<p>[Input] One input [Parameter]</p> <ul style="list-style-type: none"> power: (optional) float, default to 1.0 scale: (optional) float, default to 1.0 shift: (optional) float, default to 0.0 <p>[Constraint]</p> <ul style="list-style-type: none"> power! = 1 scale * x + shift > 0 <p>[Quantitative tool support] Yes</p>
23	Prelu	Activation function	<p>[Input] One input [Parameter]</p> <ul style="list-style-type: none"> filler: optional channel_shared: (optional) bool, indicating whether to share slope parameters across channels, default to false <p>[Constraint] None</p> <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
24	PriorBox	Obtains the real location of the target from the box proposals.	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● min_size: (mandatory) indicating the minimum frame size (in pixels) ● max_size: (mandatory) indicating the maximum frame size (in pixels) ● aspect_ratio: array, float. A repeated ratio is ignored. If no aspect ratio is provided, the default ratio 1 is used. ● flip: (optional) bool, default to true. The value true indicates that each aspect ratio is reversed. For example, for aspect ratio r, the aspect ratio 1.0/r is generated. ● clip: (optional) bool, default to false. The value true indicates that the previous value is clipped to the range [0, 1]. ● variance: array, used to adjust the variance of the BBoxes ● img_size: (optional) uint32, exclusive with img_h or img_w ● img_h: (optional) uint32 ● img_w: (optional) uint32 ● step: (optional) float, exclusive with step_h or step_w ● step_h: (optional) float ● step_w: (optional) float ● offset: float, default to 0.5 <p>[Constraint] Used for the SSD network only Output dimensions: [n, 2, detected boxes x 4, 1]</p> <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
25	Proposal	Sorts the box proposals by (proposal, score) and obtains the top N proposals by using the NMS.	<p>[Input] Three inputs (scores, bbox_pred, im_info)</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • feat_stride: (optional) float • base_size: (optional) float • min_size: (optional) float • ratio: float array • scale: float array • pre_nms_topn: (optional) int32 • post_nms_topn: (optional) int32 • nms_thresh: (optional) float <p>[Constraint]</p> <ul style="list-style-type: none"> • Used only for Faster R-CNN • ProposalParameter and PythonParameter are exclusive. • Value range of preTopK: 1-6,144 • Value range of postTopK: 1-1,024 • $scaleCnt \times ratioCnt \leq 64$ • nmsTresh: threshold for Intersection-over-Union (IoU) box filtering, $0 < nmsTresh \leq 1$ • minSize: minimum edge length of a box. A value less than this parameter is filtered out. • featStride: H/W stride between the two adjacent boxes used in default box generation • baseSize: default box size used in default box generation • ratio and scale: used in default box generation • imgH and imgW: height and width of the image input to the network. The values must be greater than 0. • Restrictions on the input dimensions: clsProb: $C = 2 \times scaleCnt \times ratioCnt$ bboxPred: $C = 4 \times scaleCnt \times ratioCnt$ bboxPrior: $N = clsProb.N, C = 4 \times scaleCnt \times ratioCnt$

No.	Operator	Definition	Boundary
			imInfo: N = clsProb.N, C = 3 [Quantitative tool support] Yes

No.	Operator	Definition	Boundary
26	PSROI Pooling	Position-sensitive region-of-interest pooling (PSROI Pooling)	<p>[Input]</p> <p>Two inputs</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • spatial_scale: (mandatory) float • output_dim: (mandatory) int32, indicating the number of output channels • group_size: (mandatory) int32, indicating the number of groups to encode position-sensitive score maps <p>[Constraint]</p> <p>Used for the Region-based Fully Convolutional Network (R-FCN)</p> <ul style="list-style-type: none"> • ROI coordinates [roiN, roiC, roiH, roiW]: $1 \leq \text{roiN} \leq 65535$, $\text{roiC} == 5$, $\text{roiH} == 1$, $\text{roiW} == 1$ • Dimensions of the input feature map: [xN, xC, xH, xW] $\text{pooledH} == \text{pooledW} == \text{groupSize} \leq 128$ pooledH and pooledW indicate the length and width of the pooled ROI. • Output format: y [yN, yC, yH, yW] • poolingMode == avg pooling, $\text{pooledH} == \text{pooledW} == \text{groupSize}$, $\text{pooledH} \leq 128$, $\text{spatialScale} > 0$, $\text{groupSize} > 0$, $\text{outputDim} > 0$ • $1 \leq \text{xN} \leq 65535$, $\text{roisN} \% \text{xN} == 0$ • HW_LIMIT defines the limits of xH and xW. $\text{xHW} = \text{xH} * \text{xW}$ $\text{pooledHW} = \text{pooledH} * \text{pooledW}$ $\text{HW_LIMIT} = (64 * 1024 - 8 * 1024) / 32$ $\text{xH} \geq \text{pooledH}$, $\text{xW} \geq \text{pooledW}$ $\text{xHW} \geq \text{pooledHW}$ $\text{xHW} / \text{pooledHW} \leq \text{HW_LIMIT}$ • In multi-batch scenarios, the ROIs are allocated equally to the batches. In addition, the batch

No.	Operator	Definition	Boundary
			sequence of the ROIs is the same as the feature. [Quantitative tool support] Yes
27	Relu	Activation function, including common ReLU and Leaky ReLU, which can be specified by parameters	[Input] One input [Parameter] <ul style="list-style-type: none"> ● negative_slope: (optional) float, default to 0 ● engine: (optional) enum, default to 0, CAFFE = 1, CUDNN = 2 [Constraint] None [Quantitative tool support] Yes
28	Reshape	Reshapes the input.	[Input] One input [Parameter] <ul style="list-style-type: none"> ● shape: constant, int64 or int ● axis: (optional) int32, default to 0 ● num_axes: (optional) int32, default to -1 [Constraint] None [Quantitative tool support] Yes

No.	Operator	Definition	Boundary
29	ROIAlign	Aggregates features using ROIs.	<p>[Input] At least two inputs</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● pooled_h: (optional) uint32, default to 0 ● pooled_w: (optional) uint32, default to 0 ● spatial_scale: (optional) float, default to 1 ● sampling_ratio: (optional) int32, default to -1 <p>[Constraint]</p> <p>Mainly used for Mask R-CNN</p> <ul style="list-style-type: none"> ● Restrictions on the feature map: <ol style="list-style-type: none"> 1) $H \times W \leq 5,248$ ($N > 1$) or $W \times C < 40,960$ ($N = 1$) 2) $C \leq 1280$ 3) $((C - 1)/128 + 1) \times \text{pooledW} \leq 216$ ● Restrictions on the ROI: <ol style="list-style-type: none"> 1) $C = 5$ (caffe), $H = 1$, $W = 1$ 2) $\text{samplingRatio} \times \text{pooledW} \leq 128$, $\text{samplingRatio} \times \text{pooledH} \leq 128$ 3) $H \geq \text{pooledH}$, $W \geq \text{pooledW}$ <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
30	ROI Pooling	Maps ROI proposals to a feature map.	<p>[Input] At least two inputs</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● pooled_h: (mandatory) uint32, default to 0 ● pooled_w: (mandatory) uint32, default to 0 ● spatial_scale: (mandatory) float, default to 1. The multiplication spatial scale factor is used to convert ROI coordinates from the input scale to the pool scale. <p>[Constraint]</p> <p>Mainly used for Faster R-CNN</p> <ul style="list-style-type: none"> ● Input dimensions: $H \times W \leq 8,160$, $H \leq 120$, $W \leq 120$ ● Output dimensions: $pooledH \leq 20$, $pooledW \leq 20$ <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
31	Scale	out = alpha x Input + beta	<p>[Input] Two inputs, each with four dimensions</p> <p>[Parameter]</p> <ul style="list-style-type: none"> • axis: (optional) int32, default to 1. Only 1 or -3 is supported. • num_axes: (optional) int32, default to 1 • filler: (optional) ignored unless only one bottom is given and scale is a learned parameter • bias_term: (optional) bool, default to false, indicating whether to learn a bias (equivalent to ScaleLayer + BiasLayer, but may be more efficient). Initialized with bias_filler. • bias_filler: (optional) default to 0 <p>[Constraint] shape of scale and bias: (n, c, 1, 1), with the C dimension equal to that of the input</p> <p>[Quantitative tool support] Yes</p>
32	ShuffleChannel	Shuffles information across the feature channels.	<p>[Input] One input</p> <p>[Parameter] group: (optional) uint32, default to 1</p> <p>[Constraint] None</p> <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
33	Sigmoid	Activation function	[Input] One input [Parameter] engine: (optional) enum, default to 0 , CAFFE = 1, CUDNN = 2 [Constraint] None [Quantitative tool support] Yes
34	Slice	Slices an input into multiple outputs.	[Input] One input [Parameter] <ul style="list-style-type: none"> • slice_dim: (optional) uint32, default to 1, exclusive with axis • slice_point: array, uint32 • axis: (optional) int32, default to 1, indicating concatenation along the channel dimension [Constraint] None [Output] None [Quantitative tool support] Yes

No.	Operator	Definition	Boundary
35	Softmax	Normalized logic function	<p>[Input] One input</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● engine: (optional) default to 0, CAFFE = 1, CUDNN = 2 ● axis: (optional) int32, default to 1, indicating the axis along which softmax is performed <p>[Constraint] Softmax can be performed on each of the four input dimensions. According to axis:</p> <ul style="list-style-type: none"> ● When axis = 1: $C \leq ((256 \times 1024/4) - 8 \times 1024 - 256)/2$ ● When axis = 0: $n \leq (56 \times 1024 - 256)/2$ ● When axis = 2: $W = 1, 0 < h < (1024 \times 1024/32)$ ● When axis = 3: $0 < W < (1024 \times 1024/32)$ <p>If the input contains fewer than four dimensions, softmax is performed only on the last dimension, with the last dimension $\leq 46,080$.</p> <p>[Quantitative tool support] Yes</p>
36	Tanh	Activation function	<p>[Input] One input</p> <p>[Parameter]</p> <p>engine: (optional) enum, default to 0, CAFFE = 1, CUDNN = 2</p> <p>[Constraint] The number of tensor elements cannot exceed INT32_MAX.</p> <p>[Quantitative tool support] Yes</p>

No.	Operator	Definition	Boundary
37	Upsample	Backward propagation of max pooling	[Input] Two inputs [Parameter] scale: (optional) int32, default to 1 [Constraint] None [Quantitative tool support] Yes

No.	Operator	Definition	Boundary
38	SSDDetectionOutput	SSD network detection output	<p>[Input] Three inputs</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● num_classes: (mandatory) int32, indicating the number of classes to be predicted ● share_location: (optional) bool, default to true, indicating that classes share one bounding box ● background_label_id: (optional) int32, default to 0 ● nms_param: (optional) indicating non-maximum suppression (NMS) ● save_output_param: (optional) indicating whether to save the detection result ● code_type: (optional) default to CENTER_SIZE ● variance_encoded_in_target: (optional) bool, default to true. The value true indicates that the variance is encoded in the target, otherwise the prediction offset needs to be adjusted accordingly. ● keep_top_k: (optional) int32, indicating the total number of BBoxes to be reserved for each image after NMS ● confidence_threshold: (optional) float, indicating that only the detection whose confidence is above the threshold is considered. If this parameter is not set, all boxes are considered. ● nms_threshold: (optional) float ● top_k: (optional) int32 ● boxes: (optional) int32, default to 1 ● relative: (optional) bool, default to true ● objectness_threshold: (optional) float, default to 0.5 ● class_threshold: (optional) float, default to 0.5 ● biases: array

No.	Operator	Definition	Boundary
			<ul style="list-style-type: none"> • general_nms_param: optional [Constraint] • Used for the SSD network only • Value range of preTopK and postTopK: 1-1024 • shareLocation = true • nmsEta = 1 • Value range of numClasses: 1-2048 • code_type = CENTER_SIZE • Value range of nms_threshold and confidence_threshold: 0.0-1.0 [Quantitative tool support] Yes
39	Reorg	Real-time object detection	[Input] One input [Parameter] <ul style="list-style-type: none"> • stride: (optional) uint32, default to 2 • reverse: (optional) bool, default to false [Constraint] sed only for YOLOv2 [Quantitative tool support] No
40	Reverse	Reversion	[Input] One input [Parameter] <p>axis: (optional) int32, default to 1. Controls the axis to be reversed. The content layout will not be reversed.</p> [Constraint] None [Quantitative tool support] No

No.	Operator	Definition	Boundary
41	LeakyRelu	LeakyRelu activation function	[Input] One input [Parameter] Same as ReLU [Constraint] None [Quantitative tool support] Yes

No.	Operator	Definition	Boundary
42	YOLODetectionOutput	YOLO network detection output	<p>[Input] Four inputs</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● num_classes: (mandatory) int32, indicating the number of classes to be predicted ● share_location: (optional) bool, default to true, indicating that classes share one bounding box ● background_label_id: (optional) int32, default to 0 ● nms_param: (optional) indicating non-maximum suppression (NMS) ● save_output_param: (optional) indicating whether to save the detection result ● code_type: (optional) default to CENTER_SIZE ● variance_encoded_in_target: (optional) bool, default to true. The value true indicates that the variance is encoded in the target, otherwise the prediction offset needs to be adjusted accordingly. ● keep_top_k: (optional) int32, indicating the total number of BBoxes to be reserved for each image after NMS ● confidence_threshold: (optional) float, indicating that only the detection whose confidence is above the threshold is considered. If this parameter is not set, all boxes are considered. ● nms_threshold: (optional) float ● top_k: (optional) int32 ● boxes: (optional) int32, default to 1 ● relative: (optional) bool, default to true ● objectness_threshold: (optional) float, default to 0.5 ● class_threshold: (optional) float, default to 0.5 ● biases: array

No.	Operator	Definition	Boundary
			<ul style="list-style-type: none"> • general_nms_param: optional [Constraint] • sed only for YOLOv2 • classNUm < 10,240; anchorBox ≤ 8 • $W \leq 1,536$ • The upper layer of yolodetectionoutput must be the yoloregion operator. [Quantitative tool support] No

B TensorFlow Operator Boundaries

Table B-1 shows the boundaries of TensorFlow operators supported by **.om** models.

Table B-1 TensorFlow operator boundaries

No.	Python API	C++ API	Boundary
1	tf.nn.avg_pool	AvgPool Type: Mean	<p>[Parameter]</p> <ul style="list-style-type: none"> ● value: 4D tensor of type float32, with shape [batch, height, width, channels] ● ksize: list or tuple of four integers, each value corresponding to the window size for each dimension of the input tensor. ● strides: list or tuple of four integers. Each value corresponding to the stride of the sliding window for each dimension of the input tensor. ● padding: string, either VALID or SAME ● data_format: string, either NHWC (default) or NCHW ● name: (optional) string <p>[Constraint]</p> <ul style="list-style-type: none"> ● $\text{kernelH} \leq \text{inputH} + \text{padTop} + \text{padBottom}$ ● $\text{kernelW} \leq \text{inputW} + \text{padLeft} + \text{padRight}$ ● $\text{padTop} < \text{windowH}$ ● $\text{padBottom} < \text{windowH}$ ● $\text{padLeft} < \text{windowW}$ ● $\text{padRight} < \text{windowW}$ <p>In addition to common restrictions, the following restrictions must be satisfied.</p> <p>The global pooling mode supports only the following scenarios:</p> <ul style="list-style-type: none"> ● $\text{outputH} == 1 \ \&\& \ \text{outputW} == 1 \ \&\& \ \text{kernelH} \geq \text{inputH} \ \&\& \ \text{kernelW} \geq \text{inputW}$ ● $\text{inputH} * \text{inputW} \leq 10,000$ <p>[Output] Tensor of the identical data type as value</p> <p>[Quantitative tool support] Yes</p>
2	tf.nn.max_pool	MaxPool	Same as tf.nn.avg_pool

No.	Python API	C++ API	Boundary
3	tf.nn.conv2d	Conv2D	<p>[Parameter]</p> <ul style="list-style-type: none"> • value: 4D tensor, with shape [batch, height, width, channels] • Data type: float32 • filter: constant tensor, with same data type and dimensions as value, with shape [filter_height, filter_width, in_channels, out_channels] • strides: non-null list or tuple of four integers, each value corresponding to the stride of the sliding window for each dimension of the input tensor • padding: non-null string, either VALID or SAME • data_format: non-null string, either NHWC (default) or NCHW • dilations: (optional) list of four integers, default to [1,1,1,1], each value corresponding to a dimension. If $k > 1$, $k - 1$ units are skipped at the corresponding dimension in filtering. The dimension sequence is determined by data_format. The values of batch and depth of dilations must be 1. • name: (optional) string <p>[Constraint]</p> <ul style="list-style-type: none"> • $(inputW + padWHead + padWTail) \geq (((FilterW-1) \times dilationW) + 1)$ • $(inputW + padWHead + padWTail) / StrideW + 1 \leq INT32_MAX$ • $(inputH + padHHead + padHTail) \geq (((FilterH-1) \times dilationH) + 1)$ • $(inputH + padHHead + padHTail) / StrideH + 1 \leq INT32_MAX$ • $0 \leq Pad < 256, 0 < FilterSize < 256, 0 < Stride < 64, 1 \leq dilationsize < 256$ <p>[Output] Tensor of the identical data type as value</p> <p>[Quantitative tool support] Yes</p>

No.	Python API	C++ API	Boundary
4	tf.concat	Concat	<p>[Parameter]</p> <ul style="list-style-type: none"> • values: list of tensor objects or a single tensor. The values of dimensions must be the same except the dimensions to be concatenated. • axis: 0D tensor of type int32, specifying the dimension to be concatenated. The value range is [-rank(values), rank(values)]. As in Python, indexing for axis is 0-based. Positive axis in the range [0, rank(values)) refers to axis-<i>th</i> dimension, while negative axis refers to [axis + rank(values)]-<i>th</i> dimension. <p>[Constraint]</p> <p>For the input tensor, the sizes of its dimensions must be the same except the dimension for concatenation.</p> <p>The range of the input tensor count is [1, 1000].</p> <p>[Output]</p> <p>Tensor, resulting from concatenation of the input tensors</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
5	tf.matmul	MatMul	<p>[Parameter]</p> <ul style="list-style-type: none"> • a: non-constant tensor of type float32, rank ≥ 2 • b: constant tensor with the same data type and rank as a • transpose_a: The value true indicates that a is transposed before multiplication. • transpose_b: The value true indicates that b is transposed before multiplication. If transpose_a is false, transpose_b is also false. • adjoint_a: The value true indicates that a is conjugated and transposed before multiplication. • adjoint_b: The value true indicates that b is conjugated and transposed before multiplication. • a_is_sparse: The value true indicates that a is treated as a sparse matrix. • b_is_sparse: The value true indicates that b is treated as a sparse matrix. • name: optional <p>[Constraint]</p> <ul style="list-style-type: none"> • The transposing property of weight is false. • The multiplication of two tensors is not supported. Only one tensor by one constant is supported. <p>[Output] Tensor of the identical data type as a and b</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
6	tf.nn.fused_batch_norm	FusedBatchNorm	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: 4D tensor of type float32 • scale: 1D tensor for scaling • offset: 1D tensor for bias • mean: 1D tensor for population mean used for inference • variance: 1D tensor for population variance used for inference • epsilon: small float number added to the variance of x • data_format: data format for x, either NHWC (default) or NCHW • is_training: bool, specifying whether the operation is used for training or inference • name: (optional) operation name <p>[Constraint]</p> <p>shape of scale, bias, mean, and var: (1, C, 1, 1), with the C dimensions equal to that of the input</p> <p>[Output]</p> <ul style="list-style-type: none"> • y: 4D tensor for the normalized, scaled, offset x • batch_mean: 1D tensor for the mean of x • batch_var: 1D Tensor for the variance of x <p>[Quantitative tool support]</p> <p>No</p>
7	tf.abs	Abs	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: tensor or sparse tensor of type float32 • name: (optional) operation name <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Returns the absolute value of x, tensor or sparse tensor. The size and type are the same as those of x.</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
8	tf.image.resize_nearest_neighbor	ResizeNearestNeighbor	<p>[Parameter]</p> <ul style="list-style-type: none"> • images: 4D tensor of type float32, with shape [batch, height, width, channels] or 3D tensor of type float32 with shape [height, width, channels] • size: 1D 2-element constant tensor, indicating the new size for the images • method: ResizeMethod.NEARESTNEIGHBOR • align_corners: bool, default to False. The value True indicates that the centers of the 4 corner pixels of the input and output tensors are aligned and the values at the corner pixels are preserved. <p>[Constraint] None</p> <p>[Output] Tensor of type float, with the identical shape as the input</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
9	tf.image.resize_bilinear	ResizeBilinear	<p>[Parameter]</p> <ul style="list-style-type: none"> ● images: 4D non-constant tensor of type float32, with shape [batch, height, width, channels] ● size: 1D 2-element constant tensor, indicating the new size for the images ● method: ResizeMethod.BILINEAR ● align_corners: bool, default to False. The value True indicates that the centers of the 4 corner pixels of the input and output tensors are aligned and the values at the corner pixels are preserved. <p>[Constraint] $(\text{outputH} \times \text{outputW}) / (\text{inputH} \times \text{inputW}) > 1/7$</p> <p>[Output] Tensor of type float, with the identical shape as the input</p> <p>[Quantitative tool support] No</p>
10	tf.cast	Cast	<p>[Input] Type: float32, int32, bool, int64, int16, int8, uint8, uint16, double</p> <p>[Parameter]</p> <ul style="list-style-type: none"> ● x: Tensor, SparseTensor, or IndexedSlices ● dtype: destination type, same as the data type of x ● name: optional <p>[Constraint] None</p> <p>[Output] Tensor, SparseTensor, or IndexedSlices with the same dtype and shape as the input</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
11	tf.nn.depthwise_conv2d	DepthwiseConv2dNative	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: 4D • filter: 4D constant, with shape [filter_height, filter_width, in_channels, channel_multiplier] • strides: non-null list of four integers, each value corresponding to the stride of the sliding window for each dimension of the input tensor • padding: string, either VALID or SAME • rate: 1D of size 2. The dilation rate in which we sample input values across the height and width dimensions in atrous convolution. If it is greater than 1, then all values of strides must be 1. • data_format: data format for input, either NHWC (default) or NCHW • name: optional <p>[Constraint] filterN = inputC = group</p> <p>[Output] 4D tensor, with shape according to data_format. For example, for format NHWC, shape = [batch, out_height, out_width, in_channels * channel_multiplier]</p> <p>[Quantitative tool support] Yes</p>

No.	Python API	C++ API	Boundary
12	tf.reshape	Reshape	<p>[Parameter]</p> <ul style="list-style-type: none"> • tensor: 1 tensor input • shape: constant tensor of type int64 or int, indicating the output shape • name: (optional) operation name <p>[Constraint] None</p> <p>[Output] Tensor of the identical data type as input</p> <p>[Quantitative tool support] Yes</p>
13	tf.squeeze	Squeeze	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: non-constant tensor • axis: list of ints, specifying the dimensions to be squeezed, default to []. It is an error to squeeze a dimension that is not 1. • name: (optional) operation name • squeeze_dims: (deprecated) exclusive with axis <p>[Constraint] None</p> <p>[Output] Tensor, with the same data and type as input, but has one or more dimensions of size 1 removed.</p> <p>[Quantitative tool support] Yes</p>

No.	Python API	C++ API	Boundary
14	tf.expand_dims	ExpandDims	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: 1 tensor input • axis: 0D (scalar), specifying the dimension index of the extended input shape • name: name of the output tensor • dim: 0D (scalar), equivalent to axis (deprecated) <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor with the same data as input, but its shape has an additional dimension of size 1 added</p> <p>[Quantitative tool support]</p> <p>Yes</p>
15	tf.greater	Greater	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: 1 tensor input • y: scalar • name: (optional) operation name <p>[Constraint]</p> <p>Broadcasting is supported, so the shape of x and shape of y are compared. For a right-aligned dimension, if the values of xdim[i] and ydim[i] are not the same, one of them must be 1 or missing.</p> <p>[Output]</p> <p>Tensor of type bool</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
16	tf.nn.relu	Relu	[Parameter] <ul style="list-style-type: none"> • features: non-constant tensor • name: (optional) operation name [Constraint] None [Output] Tensor of the identical data type as features [Quantitative tool support] Yes
17	tf.nn.relu6	Relu6	[Parameter] <ul style="list-style-type: none"> • features: non-constant tensor • name: (optional) operation name [Constraint] None [Output] Tensor of the identical data type as features [Quantitative tool support] Yes
18	tf.nn.leaky_relu	/	[Parameter] <ul style="list-style-type: none"> • features: non-constant tensor representing pre-activation values • alpha: slope of the activation function at $x < 0$ • name: (optional) operation name [Constraint] None [Output] Activation value [Quantitative tool support] Yes

No.	Python API	C++ API	Boundary
19	tf.exp	exp	[Parameter] <ul style="list-style-type: none"> • x: tensor of type float32 or double • name: (optional) operation name [Constraint] <p>None</p> [Output] <p>Tensor of the identical data type as x</p> [Quantitative tool support] <p>No</p>

No.	Python API	C++ API	Boundary
20	tf.nn.conv2d_transpose	Conv2DBackpropInput	<p>[Parameter]</p> <ul style="list-style-type: none"> ● value: 4D tensor with shape [batch, height, width, in_channels] for NHWC data format or [batch, in_channel, height, width] for NCHW data format ● filter: 4D constant tensor with shape [height, width, output_channels, in_channels] ● output_shape: 1D tensor, indicating the output shape ● strides: non-null list of integers, each value corresponding to the stride of the sliding window for each dimension of the input tensor ● padding: non-null string, either VALID or SAME ● data_format: non-null string, either NHWC or NCHW ● name: (optional) output name <p>[Constraint]</p> <ul style="list-style-type: none"> ● group = 1 ● dilation = 1 ● filterH - padHHead - 1 ≥ 0 ● filterW - padWHead - 1 ≥ 0 <p>Restrictions involving intermediate variables:</p> <ol style="list-style-type: none"> 1. $a = \text{ALIGN}(\text{filter_num}, 16) \times \text{ALIGN}(\text{filter_c}, 16) \times \text{filter_h} \times \text{filter_w} \times 2$ 2. If $\text{ALIGN}(\text{filter_c}, 16) \% 32 = 0$, $a = a/2$ 3. $\text{conv_input_width} = (\text{deconvolution input } W - 1) \times \text{strideW} + 1$ 4. $b = (\text{conv_input_width}) \times \text{filter_h} \times \text{ALIGN}(\text{filter_num}, 16) \times 2 \times 2$ 5. $a + b \leq 1024 \times 1024$ <p>[Output]</p> <p>Tensor of the identical data type as value</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
21	tf.sigmoid	Sigmoid	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: 1 tensor input • name: (optional) operation name <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as value</p> <p>[Quantitative tool support]</p> <p>Yes</p>
22	tf.add	Add	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 or int32 • y: Tensor of the identical data type as x. For two constant inputs, one of them is a scalar. • name: (optional) operation name <p>[Constraint]</p> <p>If the two inputs have inconsistent dimensions, broadcasting (that is, dimension padding) is performed.</p> <p>Broadcasting is supported in the following scenarios:</p> <ul style="list-style-type: none"> • NHWC + NHWC, NHWC + scalar • NHWC + 1 1 1 1 • NHWC + W, HWC + W, HW + W (W-based broadcasting) • NCHW + NH1C, HWC + H1C, HW + H1 • HWC + 1 WC (H-based broadcasting) <p>NOTE The input sequence of the two tensors is not fixed.</p> <p>[Output]</p> <p>Tensor of the identical data type as y</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
23	tf.multiply	Multiply Type: Mul	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 or int32 • y: Tensor of the identical data type as x. If two constants are input, their dimensions must be identical (scalar or 1D tensor) • name: (optional) operation name <p>[Constraint]</p> <p>If the two inputs have inconsistent dimensions, broadcasting (that is, dimension padding) is performed. Broadcasting is supported in the following scenarios:</p> <ul style="list-style-type: none"> • NHWC + NHWC, NHWC + scalar • NHWC + 1 1 1 1 • NHWC + W, HWC + W, HW + W (W-based broadcasting) • NCHW + NH1C, HWC + H1C, HW + H1 • HWC + 1 WC (H-based broadcasting) <p>NOTE The input sequence of the two tensors is not fixed.</p> <p>[Output]</p> <p>1 tensor</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
24	tf.subtract	Subtract Type: Sub	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: 1 tensor input • y: Tensor of the identical data type as x, constant or non-constant • name: (optional) operation name <p>[Constraint]</p> <p>If the two inputs have inconsistent dimensions, broadcasting (that is, dimension padding) is performed.</p> <p>Broadcasting is supported in the following scenarios:</p> <ul style="list-style-type: none"> • NHWC + NHWC, NHWC + scalar • NHWC + 1 1 1 1 • NHWC + W, HWC + W, HW + W (W-based broadcasting) • NCHW + NH1C, HWC + H1C, HW + H1 • HWC + 1 WC (H-based broadcasting) <p>NOTE The input sequence of the two tensors is not fixed.</p> <p>[Output]</p> <p>1 tensor</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
25	tf.nn.bias_add	BiasAdd	<p>[Parameter]</p> <ul style="list-style-type: none"> • value: non-constant tensor • bias: 1D constant tensor, with size matching the last dimension of value, of the same type as value unless value is a quantized type • data_format: string, either NHWC or NCHW • name: (optional) operation name <p>[Constraint]</p> <ul style="list-style-type: none"> • $C < 10,000$ • input and bias must have the same data layout. • When bias is added to the C dimension, the C dimensions of input and bias must of the same size. <p>[Output]</p> <p>Tensor of the identical data type as value</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
26	tf.nn.lrn	LRN	<p>[Parameter]</p> <ul style="list-style-type: none"> ● input: 4D tensor of type float32 ● depth_radius: 0D of type int, default to 5, indicating the half-width of the 1D normalization window ● bias: (optional) float, default to 1, indicating the offset (usually positive to avoid dividing by 0) ● alpha: (optional) float, default to 1, indicating the scale factor, usually positive ● beta: (optional) float, default to 0.5, indicating an exponent ● name: (optional) operation name <p>[Constraint]</p> <ul style="list-style-type: none"> ● depth_radius is an odd number greater than 0. ● Inter-channel: When depth_radius is within [1,15], alpha > 0.00001 and beta > 0.01; Otherwise, alpha and beta are of any values. When C > 1,776, depth_radius < 1,728. <p>[Output]</p> <p>Tensor of the identical data type as input</p> <p>[Quantitative tool support]</p> <p>Yes</p>
27	tf.nn.elu	Elu	<p>[Parameter]</p> <ul style="list-style-type: none"> ● features: non-constant tensor ● name: optional <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as features</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
28	tf.rsqrt	Rsqrt	[Parameter] <ul style="list-style-type: none"> • x: tensor input • name: optional [Constraint] <p>None</p> [Output] <p>Tensor of the identical data type as x</p> [Quantitative tool support] <p>Yes</p>
29	tf.log	Log	[Parameter] <ul style="list-style-type: none"> • x: Tensor of type float32 • name: optional [Constraint] <p>None</p> [Output] <p>Tensor of the identical data type as x</p> [Quantitative tool support] <p>No</p>
30	tf.tanh	Tanh	[Parameter] <ul style="list-style-type: none"> • x: non-constant tensor • name: optional [Constraint] <p>None</p> [Output] <p>Tensor of the identical data type as x</p> [Quantitative tool support] <p>Yes</p>

No.	Python API	C++ API	Boundary
31	tf.slice	Slice	<p>[Parameter]</p> <ul style="list-style-type: none"> • input_: tensor input • begin: tensor of type int32 or int64 • size: Tensor of type int32 or int64 • name: optional <p>[Constraint]</p> <p>The number of tensor elements cannot exceed INT32_MAX.</p> <p>[Output]</p> <p>Tensor of the identical data type as input_</p> <p>[Quantitative tool support]</p> <p>Yes</p>
32	tf.split	Split	<p>[Parameter]</p> <ul style="list-style-type: none"> • value: tensor input • num_or_size_splits: not supported • axis: integer, specifying the dimension along which to split • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>List of tensor objects resulting from splitting</p> <p>[Quantitative tool support]</p> <p>Yes</p>
33	tf.nn.softplus	Softplus	<p>[Parameter]</p> <ul style="list-style-type: none"> • features: Tensor of type float32 • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as features</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
34	tf.nn.softsign	Softsign	<p>[Parameter]</p> <ul style="list-style-type: none"> • features: Tensor of type float32 • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as features</p> <p>[Quantitative tool support]</p> <p>No</p>
35	tf.pad	Excessive CPU usage usually means insufficient idle CPUs in the system.	<p>[Parameter]</p> <ul style="list-style-type: none"> • tensor: 4D tensor of type float32 and int32 • paddings: constant tensor of type int32 • mode: string, CONSTANT, REFLECT, or SYMMETRIC • name: (optional), string • constant_values: scalar pad value to use, of the identical data type as tensor. <p>[Constraint]</p> <p>In CONSTANT mode: $0 \leq \text{PAD} \leq 128$, $0 < W \leq 3000$</p> <p>[Output]</p> <p>Tensor of the identical data type as tensor</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
36	tf.fake_quant_with_min_max_vars	FakeQuantWithMinMaxVars	<p>[Parameter]</p> <ul style="list-style-type: none"> • inputs: Tensor of type float32 • min: tensor of type float32 • max: tensor of type float32 • num_bits: scalar of type int, default to 8 • narrow_range: (optional) bool, default to False • name: (optional), string <p>[Constraint]</p> <p>$-65,504 \leq \text{min} \leq 65,504, -65,504 \leq \text{max} \leq 65,504$</p> <p>[Output]</p> <p>Tensor of type float32</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
37	tf.reduce_max	Max	<p>[Parameter]</p> <ul style="list-style-type: none"> ● input_tensor: Tensor, must be one of the following types: float32, int64, int32, uint8, uint16, int16, int8 ● axis: 1D list or scalar of type integer ● keepdims: bool, indicating whether to retain reduced dimensions with length 1 ● name: (optional), string ● reduction_indices: (deprecated) equivalent to axis ● keep_dims: (deprecated) equivalent to keepdims <p>[Constraint]</p> <ul style="list-style-type: none"> ● When the input tensor has four dimensions: input axis = {3,{1,2,3}}, keepDims = true, $H * W * 16 * 2 \leq 16 * 1024$ ● When the input tensor has two dimensions: input axis = {1,{1}}, keepDims = true, $H * W * \text{CEIL}(C, 16) * 16 * 2 \leq 16 * 1024$ <p>[Output]</p> <p>Reduced Tensor of the identical data type as input_tensor</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
38	tf.strided_slice	StridedSlice	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: 1 tensor • begin: 1D tensor of type int32 • end: 1D tensor of type int32 • strides: 1D tensor of type int32 • begin_mask: scalar of type int32 • end_mask: scalar of type int32 • ellipsis_mask: scalar of type int32 • new_axis_mask: scalar of type int32 • new_axis_mask: scalar of type int32 • var: variable corresponding to input_ or None • name: (optional), string <p>[Constraint] strides ≠ 0</p> <p>[Output] Tensor of the identical data type as input_</p> <p>[Quantitative tool support] No</p>
39	tf.reverse	Reverse	<p>[Parameter]</p> <ul style="list-style-type: none"> • tensor: list of tensor objects • axis: int32 or int64, indicating the dimensions to reverse • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of the identical data type as tensor</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
40	tf.realdiv	RealDiv	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 • y: Tensor of type float32 • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of the identical data type as x</p> <p>[Quantitative tool support] Yes</p>
41	tf.stack	Stack	<p>[Parameter]</p> <ul style="list-style-type: none"> • values: list of tensor objects with the same shape and type (float32 or int32) • axis: (mandatory) integer, indicating the axis to stack along, default to the first dimension • name: optional <p>[Constraint] None</p> <p>[Output] Stacked tensor of the identical data type as values</p> <p>Mandatory attributes: T, N, and axis</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
42	tf.unstack	Unpack	<p>[Parameter]</p> <ul style="list-style-type: none"> • value: tensor (rank > 0) to be unstacked, of type float32 or int32 • num: integer, indicating the length of the dimension axis, default to None • axis: (mandatory) integer, indicating the axis to unstack along, default to the first dimension • name: optional <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>List of tensor objects unstacked from value</p> <p>[Quantitative tool support]</p> <p>No</p>
43	tf.transpose	Transpose	<p>[Parameter]</p> <ul style="list-style-type: none"> • a: tensor input • perm: permutation of the dimensions of a • name: optional • conjugate: (optional) bool, default to False. Setting it to True is mathematically equivalent to <code>tf.conj(tf.transpose(input))</code> <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Resulting tensor after transposition</p> <p>[Quantitative tool support]</p> <p>Yes</p>

No.	Python API	C++ API	Boundary
44	tf.space_to_batch_nd	SpaceToBatch ND	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: ND tensor with shape $\text{input_shape} = [\text{batch}] + \text{spatial_shape} + \text{remaining_shape}$, where spatial_shape has M dimensions. The following data types are supported: uint8, int8, int16, uint16, int32, int64, and float32 • block_shape: 1D Tensor of type int32 or int64, with shape [M]. All values must be ≥ 1. • paddings: 2D tensor of type int32 or int64, with shape [M, 2]. All values must be ≥ 0. <p>[Constraint]</p> <p>When the tensor rank is 4: The length of blockShape must be 2, and the length of paddings must be 4.</p> <ul style="list-style-type: none"> • Element value of blockShape ≥ 1; Element value of paddings ≥ 0 • The padded H dimension is a multiple of blockShape[0], and the padded W dimension is a multiple of blockShape[1]. <p>[Output]</p> <p>Tensor of the identical data type as input</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
45	tf.batch_to_space_nd	BatchToSpace ND	<p>[Parameter]</p> <ul style="list-style-type: none"> ● input: ND tensor with shape $input_shape = [batch] + spatial_shape + remaining_shape$, where $spatial_shape$ has M dimensions. The following data types are supported: uint8, int8, int16, uint16, int32, int64, and float32 ● block_shape: 1D Tensor of type int32 or int64, with shape [M]. All values must be ≥ 1. ● crops: 2D tensor of type int32 or int64, with shape [M, 2]. All values must be ≥ 0. <p>[Constraint]</p> <ul style="list-style-type: none"> ● The element data type of blockShape and crops must be int32. When the dimension count of the tensor is 4, the length of blockShape must be 2, and the length of crops must be 4. ● Element value of blockShape ≥ 1; Element value of crops ≥ 0; for the crops array: $crop_start[i] + crop_end[i] < block_shape[i] * input_shape[i+1]$ <p>[Output]</p> <p>Tensor of the identical data type as images</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
46	tf.extract_image_patches	ExtractImagePatches	<p>[Parameter]</p> <ul style="list-style-type: none"> • images: 4D tensor with shape [batch, in_rows, in_cols, depth], must be one of the following types: float32, int32, int64, uint8, int8, uint16, and int16 • ksizes: list of ints with length ≥ 4 • strides: list of ints, must be [1, stride_rows, stride_cols, 1] • rate: list of ints, must be [1, rate_rows, rate_cols, 1] • padding: string, either VALID or SAME. VALID indicates that the selected patch area must be completely included in the source image. SAME indicates that the part that exceeds the source image is padded with 0. • name: optional <p>[Constraint] None</p> <p>[Output] Tensor of the identical data type as images</p> <p>[Quantitative tool support] No</p>
47	tf.floormod	FloorMod	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 or int32 • y: Tensor of the identical data type as x • name: optional <p>[Constraint] Broadcasting is supported, so the shape of x and shape of y are compared. For a right-aligned dimension, if the values of xdim[i] and ydim[i] are not the same, one of them must be 1 or missing.</p> <p>[Output] Tensor of the identical data type as x</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
48	tf.nn.softmax	Softmax	<p>[Parameter]</p> <ul style="list-style-type: none"> ● logits: non-null Tensor of type float32 ● axis: dimension softmax to be performed on, default to -1, indicating the last dimension. The value cannot be greater than the rank of logits. ● name: optional ● dim: (deprecated) equivalent to axis <p>[Constraint]</p> <ul style="list-style-type: none"> ● Softmax can be performed on each of the four input dimensions. According to axis: When axis = 1: $C \leq ((256 \times 1024/4) - 8 \times 1024 - 256)/2$ When axis = 0: $n \leq (56 \times 1024 - 256)/2$ When axis = 2: $W = 1, 0 < h < (1024 \times 1024/32)$ When axis = 3: $0 < W < (1024 \times 1024/32)$ ● If the input contains fewer than four dimensions, softmax is performed only on the last dimension, with the last dimension $\leq 46,080$. <p>[Output] Tensor of the identical type and shape as logits</p> <p>[Quantitative tool support] Yes</p>

No.	Python API	C++ API	Boundary
49	tf.math.pow	Power	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 • y: Tensor of type float32 • name: optional <p>[Constraint]</p> <p>power! = 1 scale*x + shift > 0</p> <p>[Output]</p> <p>1 tensor</p> <p>[Quantitative tool support]</p> <p>Yes</p>
50	tf.placeholder	-	<p>[Parameter]</p> <ul style="list-style-type: none"> • dtype: (mandatory) data type • shape: (mandatory) shape of the tensor to be fed <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>1 tensor</p> <p>[Quantitative tool support]</p> <p>No</p>
51	tf.shape	Shape	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: Tensor or SparseTensor • name: (optional), string • out_type: data type for the output tensor, either int32 (default) or int64 <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the data type specified by out_type</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
52	tf.math.argmax x	ArgMax	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: Tensor, must be one of the following types: int8, uint8, int16, uint16, int32, int64, float32 • axis: 1 tensor of type: int32 or int64 • out_type: data type for the output tensor, either int32 or int64 (default) • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of the data type specified by out_type</p> <p>[Quantitative tool support] No</p>
53	tf.gather	Gather GatherV2	<p>[Parameter]</p> <ul style="list-style-type: none"> • params: 1 tensor, must be at least rank axis + 1 • indices: tensor of type int32 or int64, must be in range [0, params.shape[axis]) • axis: output tensor of type int32 or int64, specifying the axis in params to gather indices from, rank = 0 • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of the identical data type as params</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
54	tf.gather_nd	GatherNd	<p>[Parameter]</p> <ul style="list-style-type: none"> • params: 1 tensor, must be at least rank axis + 1 • indices: 1 tensor of type: int32 or int64 • name: (optional), string <p>[Constraint]</p> <ul style="list-style-type: none"> • indices: The last dimension of indices can be at most the rank of params. • The elements in the last dimension of indices correspond to the coordinates along a dimension of params. Therefore, the coordinate rules must be met. • The coordinates of the corresponding dimension in the indices cannot exceed the dimension size. <p>[Output]</p> <p>Tensor of the identical data type as params</p> <p>[Quantitative tool support]</p> <p>Yes</p>
55	tf.math.floorDiv	FloorDiv	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 or int32 • y: Tensor, denominator of type float32 or int32 • name: (optional), string <p>[Constraint]</p> <p>Broadcasting is supported, so the shape of x and shape of y are compared. For a right-aligned dimension, if the values of xdim[i] and ydim[i] are not the same, one of them must be 1 or missing.</p> <p>[Output]</p> <p>Tensor, floor (x/y)</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
56	tf.range	Range	<p>[Parameter]</p> <ul style="list-style-type: none"> • start: start constant scalar of type float32 or int32 • limit: end constant scalar of type float32 or int32 • delta: stride constant scalar of type float32 or int32 • dtype: data type of the resulting tensor • name: (optional), string <p>[Constraint] None</p> <p>[Output] 1 1D tensor</p> <p>[Quantitative tool support] No</p>
57	tf.tile	Tile	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: Tensor, must be one of the following types: int8, uint8, int16, uint16, int32, int64, float32 • multiples: 1D constant tensor of type int32. The length must be the same as that of input. • name: (optional), string <p>[Constraint] None</p> <p>[Output] 1 tensor</p> <p>[Quantitative tool support] Yes</p>

No.	Python API	C++ API	Boundary
58	tf.size	Size	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: tensor of type float32 • name: (optional), string • out_type: data type for the output tensor, default to int32 <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the data type specified by out_type</p> <p>[Quantitative tool support]</p> <p>No</p>
59	tf.fill	Fill	<p>[Parameter]</p> <ul style="list-style-type: none"> • dims: 1D tensor of type int32 • value: variable of type int32 or float32 • name: (optional), string <p>[Constraint]</p> <p>The following padding modes are supported: Constant, GivenTensor, Range, Diagonal, Gaussian, MSRA, Uniform, UniformInt, UniqueUniform, and XavierFill. When the Uniform, UniformInt, UniqueUniform, and xavier padding modes are used, the value range of the generated value is [min, max).</p> <p>[Output]</p> <p>Tensor of the identical data type as value</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
60	tf.concat	Concat	<p>[Parameter]</p> <ul style="list-style-type: none"> • value: list of tensor objects of type int32 or float32 • axis: int32, indicating the dimensions to concatenate • name: (optional), string <p>[Constraint]</p> <p>For the input tensor, the sizes of its dimensions must be the same except the dimension for concatenation.</p> <p>The range of the input tensor count is [1, 1000].</p> <p>[Output]</p> <p>1 tensor</p> <p>[Quantitative tool support]</p> <p>Yes</p>
61	tf.reverse	Reverse	<p>[Parameter]</p> <ul style="list-style-type: none"> • tensor: list of tensor objects • axis: int32, indicating the dimensions to reverse • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as tensor</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
62	tf.reduce_sum	sum	<p>[Parameter]</p> <ul style="list-style-type: none"> • input_tensor: tensor input • axis: int32, indicating the dimensions to sum up • keepdims: bool, indicating whether to retain reduced dimensions • name: (optional), string • reduction_indices: (deprecated) string, equivalent to axis • keep_dims: deprecated alias for keepdims <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as tensor</p> <p>[Quantitative tool support]</p> <p>No</p>
63	tf.math.maximum	Maximum	<p>[Parameter]</p> <ul style="list-style-type: none"> • max: tensor, must be one of the following types: int32, int64, float32 • y: tensor of the identical data type as x • name: (optional), string <p>[Constraint]</p> <p>Broadcasting is supported, so the shape of x and shape of y are compared. For a right-aligned dimension, if the values of xdim[i] and ydim[i] are not the same, one of them must be 1 or missing.</p> <p>[Output]</p> <p>Tensor. Returns the max of x and y (x < y ? x:y)</p> <p>Identical data type as x</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
64	tf.math.minimum	Minimum	<p>[Parameter]</p> <ul style="list-style-type: none"> • max: tensor, must be one of the following types: int32, int64, float32 • y: tensor of the identical data type as x • name: optional <p>[Constraint]</p> <p>Broadcasting is supported in the following two scenarios: NHWC + scaler, NHWC + NHWC</p> <p>[Output]</p> <p>Tensor. Returns the max of x and y ($x < y ? x:y$)</p> <p>Identical data type as x</p> <p>[Quantitative tool support]</p> <p>No</p>
65	tf.clip_by_value	ClipByValue	<p>[Parameter]</p> <ul style="list-style-type: none"> • t: Tensor • clip_value_min: minimum value to clip by • clip_value_max: maximum value to clip by • name: (optional), string <p>[Constraint]</p> <p>The minimum value must be less than or equal to the maximum value.</p> <p>[Output]</p> <p>Clipped tensor. The return value range is [clip_value_min, clip_value_max].</p> <p>[Quantitative tool support]</p> <p>No</p>
66	tf.math.logical_not	LogicalNot	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: tensor of type bool • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of type bool</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
67	tf.math.logical_and	LogicalAnd	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: tensor of type bool • y: tensor of type bool • name: (optional), string <p>[Constraint]</p> <p>Broadcasting is supported in the following dimension scenarios: NHWC and [1,1,1,1], [N,H,W,C], [N,H,W,1], [1,H,W,C], [N,1,1,C]</p> <p>[Output]</p> <p>Tensor of type bool</p> <p>[Quantitative tool support]</p> <p>No</p>
68	tf.equal	Equal	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor • y: tensor of the identical data type as x • name: (optional), string <p>[Constraint]</p> <p>Broadcasting is supported, so shape of x and shape of y are compared. For a right-aligned dimension, the values of xdim[i] and ydim[i] are the same. If not, one of them must be 1 or missing.</p> <p>[Output]</p> <p>Tensor of type bool</p> <p>[Quantitative tool support]</p> <p>No</p>
69	tf.square	Square	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as x</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
70	tf.image.crop_and_resize	CropAndResize	<p>[Parameter]</p> <ul style="list-style-type: none"> • image: 4D tensor, must be one of the following types: float32, int8, int32, int64, with shape [num_boxes, 4] • boxes: 2D tensor of type float32, with shape [num_boxes] • box_ind: 1D tensor of type int32 • crop_size: 1D 2-element tensor of type int32 • method: string, indicating the interpolation method, either bilinear (default) or nearest • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of type float32</p> <p>[Quantitative tool support] No</p>
71	tf.math.top_k	TopKV2	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: 1D tensor or higher with the last dimension at least k, of type float32 (k: scalar of type int32, ≥ 1) • sorted: bool • name: (optional), string <p>[Constraint] k must be a constant.</p> <p>[Output]</p> <ul style="list-style-type: none"> • values: tensor, indicating k largest elements along each last dimensional slice • indices: tensor, indicating the indices of values of input <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
72	tf.invert_permutation	InvertPermutation	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: 1D tensor of type int32 or int64 • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as x</p> <p>[Quantitative tool support]</p> <p>No</p>
73	tf.multinomial	Multinomial	<p>[Parameter]</p> <ul style="list-style-type: none"> • logits: 2D tensor, with shape [batch_size, num_classes] • num_samples: scalar, indicating the number of samples to draw • seed: int32 or int64, used to create a random seed • name: (optional), string • output_dtype: integer, data type for the output tensor, default to int64 <p>[Constraint]</p> <p>When seed is 0, the generated random is dynamic.</p> <p>Number of rows in the output data = Number of rows in the output data; Number of columns in the output data = num_samples</p> <p>[Output]</p> <p>Tensor of the data type specified by output_dtype</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
74	tf.reverse_sequence	ReverseSequence	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: tensor input • seq_lengths: 1D tensor of type int32 or int64 • seq_axis: scalar of type integer • batch_axis: scalar of type integer • name: (optional), string <p>[Constraint]</p> <ul style="list-style-type: none"> • The length of seq_lengths must be equal to the number of elements of input in batchAxis. • The maximum element in seq_lengths must be less than or equal to the number of elements in seq_dim. • seqAxis, batchAxis, seqDim, and batchDim must be of type int64. • seqAxis and seqDim are exclusive. batchAxis and batchDim are exclusive. • batchAxis and batchDim are optional. The default values are 0. <p>[Output]</p> <p>Tensor of the data type specified by input</p> <p>[Quantitative tool support]</p> <p>No</p>
75	tf.math.reciprocal	Reciprocal	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 • name: (optional), string <p>[Constraint]</p> <p>The input data cannot contain 0.</p> <p>[Output]</p> <p>Tensor of the identical data type as x</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
76	tf.nn.selu	Selu	[Parameter] <ul style="list-style-type: none"> • features: Tensor of type float32 • name: (optional), string [Constraint] <p>None</p> [Output] <p>Tensor of the identical data type as features</p> [Quantitative tool support] <p>No</p>
77	tf.math.acosh	Acosh	[Parameter] <ul style="list-style-type: none"> • x: Tensor of type float32 • name: (optional), string [Constraint] <p>None</p> [Output] <p>Tensor of the identical data type as x</p> [Quantitative tool support] <p>No</p>
78	tf.math.asinh	Asinh	[Parameter] <ul style="list-style-type: none"> • x: Tensor of type float32 • name: (optional), string [Constraint] <p>None</p> [Output] <p>Tensor of the identical data type as x</p> [Quantitative tool support] <p>No</p>

No.	Python API	C++ API	Boundary
79	tf.math.reduce_prod	Prod	<p>[Parameter]</p> <ul style="list-style-type: none"> • input_tensor: tensor input • axis: dimension to reduce • keepdims: bool, indicating whether to retain reduced dimensions • name: (optional), string • reduction_indices: (deprecated) equivalent to axis • keep_dims: (deprecated) equivalent to keepdims <p>[Constraint] None</p> <p>[Output] Tensor and reduced tensor</p> <p>[Quantitative tool support] No</p>
80	tf.math.sqrt	Sqrt	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of the identical data type as x</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
81	tf.math.reduce_all	All	<p>[Parameter]</p> <ul style="list-style-type: none"> • input_tensor: tensor of type bool • axis: dimension to reduce • keepdims: bool • name: (optional), string • reduction_indices: (deprecated) equivalent to axis • keep_dims: (deprecated) equivalent to keepdims <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as input_tensor</p> <p>[Quantitative tool support]</p> <p>No</p>
82	tf.nn.l2_normalize	L2Normalize	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type boolean • axis: dimension along which to normalize <ul style="list-style-type: none"> - For format NCHW, axis must be set to 1. - For format NHWC, axis must be set to 3. • epsilon: lower bound value for the norm. If $\text{norm} < \sqrt{\text{epsilon}}$, sqrt(epsilon) is used as the divisor. • name: (optional), string • dim: (deprecated) equivalent to axis <p>[Constraint]</p> <p>$H*W*2 < 256*1024/4$</p> <p>[Output]</p> <p>Tensor of the identical data type as x</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
83	tf.keras.backend.hard_sigmoid	Hardsigmoid	<p>[Parameter] x: tensor input</p> <p>[Constraint] None</p> <p>[Output] Output Tensor. If $x < -2.5$, 0 is returned. If $x > 2.5$, 1 is returned. If $-2.5 \leq x \leq 2.5$, $0.2 * x + 0.5$ is returned.</p> <p>[Quantitative tool support] No</p>
84	tf.keras.layers.ThresholdedReLU	ThresholdedReLU	<p>[Parameter] theta: scalar of type float32, ≥ 0</p> <p>[Constraint] None</p> <p>[Output] Tensor</p> <p>[Quantitative tool support] No</p>
85	tf.math.acos	Acos	<p>[Parameter] x: Tensor of type float32, int32, or int64 name: (optional), string</p> <p>[Constraint] The input data range is $(-1 \leq x \leq 1)$, and the output data range is $(0 \leq y \leq \pi)$.</p> <p>[Output] Tensor of the identical data type as x</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
86	tf.math.atan	Arctan	<p>[Parameter] x: Tensor of type float32, int32, or int64 name: (optional), string</p> <p>[Constraint] The input data range is $(-65504 \leq x \leq 65504)$, and the output data range is $(-\pi/2 \leq y \leq \pi/2)$.</p> <p>[Output] Tensor of the identical data type as x</p> <p>[Quantitative tool support] No</p>
87	tf.math.asin	Asin	<p>[Parameter] <ul style="list-style-type: none"> x: Tensor of type float32, int32, or int64 name: (optional), string </p> <p>[Constraint] The input data range is $(-1 \leq x \leq 1)$, and the output data range is $(-\pi/2 \leq y \leq \pi/2)$.</p> <p>[Output] Tensor of the identical data type as x</p> <p>[Quantitative tool support] No</p>
88	tf.math.atanh	Atanh	<p>[Parameter] <ul style="list-style-type: none"> x: Tensor of type float32, int32, or int64 name: (optional), string </p> <p>[Constraint] Input data range: x within $(-1, 1)$</p> <p>[Output] Tensor of the identical data type as x</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
89	tf.math.tan	Tan	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32, int32, or int64 • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type as x</p> <p>[Quantitative tool support]</p> <p>No</p>
90	tf.math.logical_or	LogicalOr	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type bool • y: Tensor of type bool • name: (optional), string <p>[Constraint]</p> <p>Broadcasting is supported, so shape of x and shape of y are compared. For a right-aligned dimension, the values of xdim[i] and ydim[i] are the same. If not, one of them must be 1 or missing.</p> <p>[Output]</p> <p>Tensor of type bool</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
91	tf.math.reduce_min	ReduceMin	<p>[Parameter]</p> <ul style="list-style-type: none"> • input_tensor: Tensor, must be one of the following types: float32, int64, int32, uint8, uint16, int8, int16 • axis: dimension to reduce • keepdims: scalar of type bool • name: (optional), string • reduction_indices: (deprecated) equivalent to axis • keep_dims: (deprecated) equivalent to keepdims <p>[Constraint]</p> <ul style="list-style-type: none"> • When the input tensor has four dimensions: input axis = {3,{1,2,3}}, keepDims = true, $H * W * 16 * 2 \leq 16 * 1024$ • When the input tensor has two dimensions: input axis = {1,{1}}, keepDims = true, $H * W * \text{CEIL}(C, 16) * 16 * 2 \leq 16 * 1024$ <p>[Output]</p> <p>Tensor of the identical data type as input_tensor</p> <p>[Quantitative tool support]</p> <p>No</p>
92	tf.math.negative	Neg	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32, int64, or int32 • name: (optional), string <p>[Constraint]</p> <p>The input data range is $(-65504 \leq x \leq 65504)$, and the output data range is $(-65504 \leq y \leq 65504)$.</p> <p>[Output]</p> <p>Tensor. Returns $-x$.</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
93	tf.math.greater_equal	GreaterEqual	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor, must be one of the following types: float32, int64, int32, uint8, uint16, int8, int16 • y: tensor of the identical data type as x • name: (optional), string <p>[Constraint]</p> <p>The input data range is $(-65504 \leq x \leq 65504)$</p> <p>[Output]</p> <p>Tensor of type bool</p> <p>[Quantitative tool support]</p> <p>No</p>
94	tf.space_to_depth	SpaceToDepth	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: tensor, must be one of the following types: float32, int64, int32, uint8, int8 • block_size: scalar of type integer, ≥ 2 • data_format: string, must be NHWC (default), NCHW, or NCHW_VECT_C • name: (optional), string <p>[Constraint]</p> <p>$blockSize \geq 1$ and blockSize must be a divisor of both the input height and width.</p> <p>[Output]</p> <p>Tensor of the identical data type as input</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
95	tf.depth_to_space	DepthToSpace	<p>[Parameter]</p> <ul style="list-style-type: none"> • input: tensor, must be one of the following types: float32, int64, int32, uint8, int8 • block_size: scalar of type integer, ≥ 2 • data_format: string, must be NHWC (default), NCHW, or NCHW_VECT_C • name: (optional), string <p>[Constraint]</p> <p>blockSize ≥ 1, and blockSize * blockSize must be exactly divided by C.</p> <p>[Output]</p> <p>Tensor of the identical data type as input</p> <p>[Quantitative tool support]</p> <p>No</p>
96	tf.math.round	Round	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32, int64, or int32 • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type and shape as x</p> <p>[Quantitative tool support]</p> <p>No</p>
97	tf.math rint	Rint	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: tensor, must be one of the following types: float32, int64, int32, uint8, int8 • name: (optional), string <p>[Constraint]</p> <p>None</p> <p>[Output]</p> <p>Tensor of the identical data type and shape as x</p> <p>[Quantitative tool support]</p> <p>No</p>

No.	Python API	C++ API	Boundary
98	tf.math.less	Less	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor, must be one of the following types: float32, int64, int32, uint8, uint16, int8, int16 • y: tensor of the identical data type as x • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of type bool</p> <p>[Quantitative tool support] No</p>
99	tf.math.sinh	Sinh	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of the identical output type as x</p> <p>[Quantitative tool support] No</p>
100	tf.math.cosh	Cosh	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32 • name: (optional), string <p>[Constraint] None</p> <p>[Output] Tensor of the identical output type as x</p> <p>[Quantitative tool support] No</p>

No.	Python API	C++ API	Boundary
101	tf.math.squared_difference	Squared_difference	<p>[Parameter]</p> <ul style="list-style-type: none"> • x: Tensor of type float32, int64, or int32 • y: tensor of the identical data type as x • name: (optional), string <p>[Constraint]</p> <p>Broadcasting is supported only in the following scenarios:</p> <p>One NCHW tensor and one tensor of the following format: dim{} = [1,1,1,1], [N,C,H,W], [N,1,H,W], [1,C,H,W], [N,C,1,1], [1,C,1,1], [1,1,H,W], or [N,1,1,1]</p> <p>[Output]</p> <p>Tensor of the identical data type as x</p> <p>[Quantitative tool support]</p> <p>No</p>