



StarFive  
赛昉科技

## 昉·星光 2 AI套件操作指南

版本：1.1

日期：2024/11/29

Doc ID: VisionFive 2-QSGCH-002

# 法律声明

阅读本文件前的重要法律告知。

## 版权注释

版权 © 广东赛昉科技有限公司，2024。版权所有。

本文档中的说明均基于“视为正确”提供，可能包含部分错误。内容可能因产品开发而定期更新或修订。广东赛昉科技有限公司（以下简称“赛昉科技”）保留对本协议中的任何内容进行更改的权利，恕不另行通知。

赛昉科技明确否认任何形式的担保、解释和条件，无论是明示的还是默示的，包括但不限于适销性、特定用途适用性和非侵权的担保或条件。

赛昉科技无需承担因应用或使用任何产品或电路而产生的任何责任，并明确表示无需承担任何及所有连带责任，包括但不限于间接、偶然、特殊、惩戒性或由此造成的损害。

本文件中的所有材料受版权保护，为赛昉科技所有。不得以任何方式修改、编辑或断章取义本文件中的说明，本文件或其任何部分仅限用于内部使用或教育培训。使用文件中包含的说明，所产生的风险由您自行承担。赛昉科技授权复制本文件，前提是您保留原始材料中包含的所有版权声明和其他相关声明，并严格遵守此类条款。本版权许可不构成对产品或服务的许可。

## 联系我们：

地址：广东省佛山市顺德区大良街道云路社区昊阳路2号A区S201室

网站：<http://www.starfivetech.com>

邮箱：[sales@starfivetech.com](mailto:sales@starfivetech.com)（销售） [support@starfivetech.com](mailto:support@starfivetech.com)（支持）

---

# Contents

|                      |    |
|----------------------|----|
| List of Tables.....  | 4  |
| List of Figures..... | 5  |
| 法律声明.....            | 2  |
| 前言.....              | 6  |
| 1. 简介.....           | 7  |
| 2. 前期准备.....         | 8  |
| 2.1. 硬件准备.....       | 8  |
| 2.2. 软件准备.....       | 8  |
| 3. 编译.....           | 11 |
| 4. 演示案例.....         | 13 |
| 5. 附录.....           | 16 |

## List of Tables

|                     |   |
|---------------------|---|
| Table 0-1 修订历史..... | 6 |
|---------------------|---|



StarFive  
赛昉科技

# List of Figures

|                                 |    |
|---------------------------------|----|
| Figure 2-1 昉·星光 2 AI套件.....     | 8  |
| Figure 2-2 选择选项1.....           | 10 |
| Figure 3-1 缺少头文件.....           | 11 |
| Figure 3-2 报错.....              | 11 |
| Figure 3-3 Hailo-8L模块信息.....    | 12 |
| Figure 4-1 开启Hailo Monitor..... | 13 |
| Figure 4-2 指定环境变量.....          | 13 |
| Figure 4-3 监测数据.....            | 13 |
| Figure 4-4 修改启动脚本.....          | 14 |
| Figure 4-6 报错.....              | 14 |
| Figure 4-7 报错.....              | 15 |

# 前言

关于本指南和技术支持信息

## 关于本手册

用户通过该手册能快速获取有关赛昉科技昉·星光 2 AI套件的基本信息和编译方法，包括前期软硬件准备、编译、演示案例等。

## 修订历史

Table 0-1 修订历史

| 版本  | 发布说明       | 修订              |
|-----|------------|-----------------|
| 1.1 | 2024/11/29 | 替换使用Hailo-8L型号。 |
| 1.0 | 2024/11/07 | 首次发布。           |

## 注释和注意事项

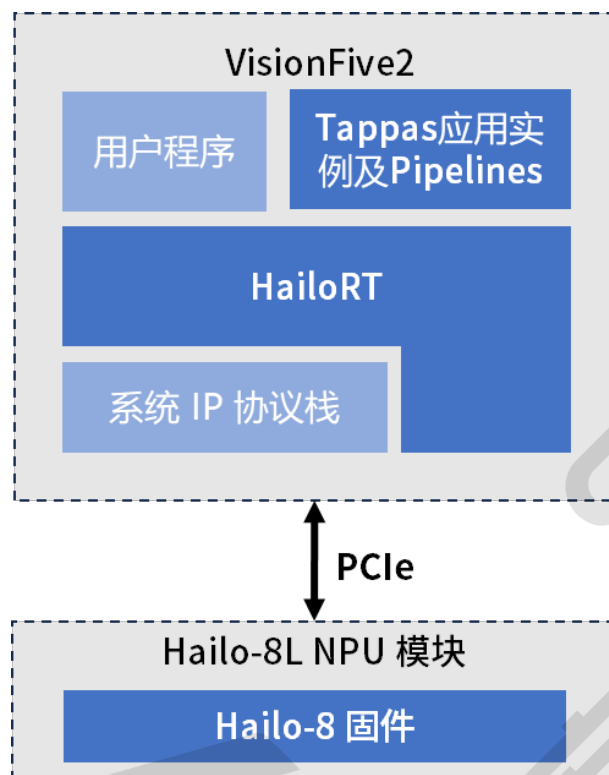
本指南中可能会出现以下注释和注意事项：

-  **Tip:**  
建议如何在某个主题或步骤中应用信息。
-  **Note:**  
解释某个特例或阐释一个重要的点。
-  **Important:**  
指出与某个主题或步骤有关的重要信息。
-  **CAUTION:**  
表明某个操作或步骤可能会导致数据丢失、安全问题或性能问题。
-  **Warning:**  
表明某个操作或步骤可能导致物理伤害或硬件损坏。

# 1. 简介

昉·星光 2 AI套件由赛昉科技携手边缘人工智能（AI）处理器的领先芯片制造商Hailo共同推出，旨在为边缘计算、工业智能、安全、机器人、网关路由、智能家居等领域应用提供最具性价比的高性能RISC-V人工智能解决方案。

昉·星光 2 AI套件包括昉·星光 2高性能RISC-V单板计算机，以及Hailo-8L M.2 AI加速模块：以昉·星光 2作为平台，移植并运行了HailoRT、HailoRT-driver、Tappas等组件，通过PCIe与Hailo的 Hailo-8L NPU模块结合，实现了AI应用的加速，成功运行了YOLOv5、YOLOv8、MobileNet\_SSD等模型。该套件具有低延迟和低功耗等优势，可快速构建复杂的AI视觉应用程序，高效执行深度学习推理任务，如目标检测、姿态检测、图像分割和人脸识别等。



## 2. 前期准备

本章主要介绍了昉·星光 2 AI套件操作所需的准备，包括：

- [硬件准备 \(on page 8\)](#)
- [软件准备 \(on page 8\)](#)

### 2.1. 硬件准备

#### 所需硬件

- 昉·星光 2
- 32 GB（或更大）的Micro SD卡
- Hailo-8L M.2 AI加速模块
- USB摄像头（型号：logi HD 1080P）
- HDMI显示器
- 散热器或风扇（用于Hailo-8L M.2 AI加速模块散热）
- 键盘
- 鼠标

#### 连接方式

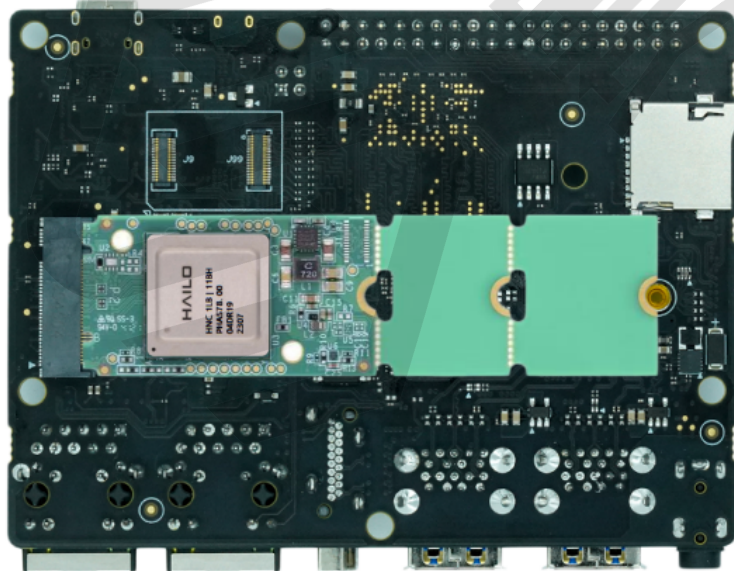
Hailo-8L M.2 AI加速模块通过M.2 M-Key接入昉·星光 2，如下图：



Tip:

建议加上散热器或用风扇进行散热。

Figure 2-1 昉·星光 2 AI套件



### 2.2. 软件准备

本节介绍了以下三个所需软件准备：

- [所需镜像/代码版本 \(on page 9\)](#)
- [镜像烧录/依赖库安装 \(on page 9\)](#)
- [获取源码并应用补丁 \(on page 10\)](#)

## 所需镜像/代码版本

- [Debian12](#) (202409)
- [debian-deb](#) (为适配hailo所需更新的Debian内核包等文件)
- Tappas (v3.30.0, [daffd36ecab5110d47107255fd7ec4c779758f2e](#))
- HailoRT (v4.19.0, [ac19e12b86170e1b0967e7d8aa607a0100cb0077](#))
- HailoRT-driver (v4.19.0, [eb2a8649752abd424c6d2e5109e9ec92d6d2d5f6](#))
- [Hailort与Tappas补丁](#) (用于适配昉·星光 2的HailoRT与Tappas补丁)
- [NpuDetectorLib demo](#)

## 镜像烧录/依赖库安装

1. 请参考[该链接](#)，将Debian烧录到SD卡。



### Note:

后续操作建议均在user用户下操作，避免root下操作导致权限异常。

2. 更新内核、头文件等：

- a. 执行以下命令，下载并解压hailort-deb-1.tar.gz文件：

```
$ tar -xvzf hailort-deb-1.tar.gz
```

- b. 执行以下命令，更新u-boot-menu：

```
$ cd hailort-deb
$ sudo dpkg -i u-boot-menu_4.2.2-SF113_all.deb
```

- c. 执行以下命令，更新内核、头文件、libc库以及VPU驱动：

```
$ cd 6.6
$ sudo dpkg -i ./*
```

- d. 执行以下命令，重启系统：

```
$ sudo reboot
```

3. 执行以下命令，安装编译环境依赖：

```
$ sudo apt install make cmake automake build-essential autoconf bc bison meson flex wget curl git
git-lfs libgirepository1.0-dev gcc g++ rsync x11-utils -y
```

4. 安装Python环境。安装python3-platformdirs，新版本Snapshot不支持低于3.0版本的python3-platformdirs，需要手动获取并安装：

```
$ wget
https://snapshot.debian.org/archive/debian/20221210T034654Z/pool/main/p/platformdirs/python3-platformdirs_2.6.0-1_all.deb && sudo dpkg -i python3-platformdirs_2.6.0-1_all.deb
$ sudo apt install python3.11 python3.11-dev python3-setuptools python3-virtualenv python3-pip
python-gi-dev -y
```

5. 执行以下命令，安装GStreamer相关库：

```
$ sudo apt install libgstreamer-plugins-bad1.0-dev libgstreamer-plugins-base1.0-dev libgstreamer1.0-dev
-y
```

6. 执行以下命令，安装OpenCV与GStreamer插件：

```
$ sudo apt install libopencv*-dev libssl-dev pciutils libcairo2-dev libzmq3-dev gstreamer1.0-tools -y
```

7. 执行以下命令，安装Rust：



**Note:**

编译pydantic等包时需要该语言。

```
$ curl --proto 'https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
```

如下图所示，弹出Rust安装选项时，请选择选项1：

Figure 2-2 选择选项1

```
1) Proceed with standard installation (default - just press enter)
2) Customize installation
3) Cancel installation
>1
```

完成安装后，添加Rust的环境变量：

```
$ source "$HOME/.cargo/env"
```

8. 执行以下命令，指定python3为python3.11：

```
$ sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.11 1
```

## 获取源码并应用补丁

- 执行以下命令，获取Tappas源码并应用：

```
$ git clone https://github.com/hailo-ai/tappas.git
$ cd tappas
$ git checkout daffd36ecab5110d47107255fd7ec4c779758f2e
$ cp path_to_hailo_patches/tappas_h8l/* ./
$ git apply 0001-Added-tappas-adaptation-to-VisionFive2.patch
$ git apply 0002-Add-demo-for-h8l-on-vf2.patch
```

- 执行以下命令，获取HailoRT源码并应用：



**Note:**

请拉取并存放在tappas路径下，便于后续Tappas编译。

```
$ mkdir -p tappas/hailort
$ cd tappas/hailort
$ git clone https://github.com/hailo-ai/hailort.git sources
$ cd source
$ git checkout ac19e12b86170e1b0967e7d8aa607a0100cb0077
$ cp path_to_hailo_patches/0001-Added-HailoRT-adaptation-to-VisionFive2.patch ./
$ git apply 0001-Added-HailoRT-adaptation-to-VisionFive2.patch
```

- 执行以下命令，获取HailoRT-drivers源码并应用：

```
$ cd ~
$ git clone https://github.com/hailo-ai/hailort-drivers.git
$ cd hailort-driver
$ git checkout eb2a8649752abd424c6d2e5109e9ec92d6d2d5f6
```

## 3. 编译

### HailoRT

1. 执行以下命令，编译HailoRT:

```
$ cd tappas/hailort/sources
$ cmake -S. -Bbuild -DCMAKE_BUILD_TYPE=Release -DHAILO_BUILD_GSTREAMER=1
$ sudo cmake --build build --config release --target gsthailo install -j4
```

如下图所示，HailoRT编译安装完毕后，HailoRT的头文件安装路径下缺少了hailort\_dma\_heap.h头文件，需要自行拷贝到对应路径下：

Figure 3-1 缺少头文件

```
user@starfive:/usr/local/include/hailo$ ls
buffer.hpp          hailort_common.hpp      platform.h
device.hpp          hailort_defaults.hpp    quantization.hpp
dma_mapped_buffer.hpp hef.hpp                 runtime_statistics.hpp
event.hpp           infer_model.hpp          stream.hpp
expected.hpp        inference_pipeline.hpp   transform.hpp
hailort.h           network_group.hpp        vdevice.hpp
hailort.hpp         network_rate_calculator.hpp vstream.hpp
user@starfive:/usr/local/include/hailo$ ls /home/user/tappas/hailort/sources/hailort/libhailort/include/hailo/
buffer.hpp          hailort_defaults.hpp    quantization.hpp
device.hpp          hailort_dma_heap.h      runtime_statistics.hpp
dma_mapped_buffer.hpp hef.hpp                 stream.hpp
event.hpp           infer_model.hpp          transform.hpp
expected.hpp        inference_pipeline.hpp   vdevice.hpp
hailort.h           network_group.hpp        vstream.hpp
hailort.hpp         network_rate_calculator.hpp
hailort_common.hpp platform.h
user@starfive:/usr/local/include/hailo$
```

2. 执行以下命令，将hailort\_dma\_heap.h头文件自行拷贝到对应路径下：

```
$ sudo cp hailort/libhailort/include/hailo/hailort_dma_heap.h /usr/local/include/hailo/
```

否则在Tappas编译安装时会遇到以下错误：

Figure 3-2 报错

```
-- Build files have been written to: /home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/build
[ 6%] Building CXX object CMakeFiles/gsthailo.dir/gst-hailo/gsthailoplugin.cpp.o
In file included from /home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/gst-hailo/os/linux/dma_b
uf_allocator_wrapper.hpp:24,
                 from /home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/gst-hailo/gsthailonet.h
p:31,
                 from /home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/gst-hailo/gsthailoplugin
.cpp:23:
/home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/gst-hailo/os/linux/../../gsthailo_dmabuf_alloc
ator.hpp:24:10: fatal error: hailo/hailort_dma_heap.h: No such file or directory
   24 | #include "hailo/hailort_dma_heap.h"
      |
compilation terminated.
gmake[2]: *** [CMakeFiles/gsthailo.dir/build.make:76: CMakeFiles/gsthailo.dir/gst-hailo/gsthailoplugin.cpp.o] Error 1
gmake[1]: *** [CMakeFiles/Makefile2:83: CMakeFiles/gsthailo.dir/all] Error 2
gmake: *** [Makefile:136: all] Error 2
```

### HailoRT-driver

1. 执行以下命令，编译HailoRT-driver:

```
$ cd hailort-drivers/linux/pcie
$ make all -j$(nproc)
$ sudo make install
$ sudo modprobe hailo_pcie
```

2. 固件下载与自动加载设置。

- a. 执行以下命令，进入pcie-driver源码顶层路径下：

```
$ cd hailort-drivers
$ ./download_firmware.sh
$ sudo mkdir -p /lib/firmware/hailo/
$ sudo mv hailo8_fw.<VERSION>.bin /lib/firmware/hailo/hailo8_fw.bin
```

```
$ sudo cp ./linux/pcie/51-hailo-udev.rules /etc/udev/rules.d/  
$ sudo udevadm control --reload-rules && sudo udevadm trigger
```

**Note:**

<VERSION>与当前HailoRT-driver版本相同，自行查看确认，此例中为4.19.0。

- b. HailoRT-driver的上述操作完成后需要执行重启，重启后可通过以下命令验证HailoRT与HailoRT-driver是否正常编译安装：

```
$ hailortcli fw-control identify
```

可查看连接的Hailo-8L模块信息：

**Figure 3-3 Hailo-8L模块信息**

```
user@starfive:~$ hailortcli fw-control identify  
Executing on device: 0001:01:00.0  
Identifying board  
Control Protocol Version: 2  
Firmware Version: 4.19.0 (release,app,extended context switch buffer)  
Logger Version: 0  
Board Name: Hailo-8  
Device Architecture: HAIL08L  
Serial Number: HLDDLBB234500007  
Part Number: HM21LB1C2LAE  
Product Name: HAIL0-8L AI ACC M.2 B+M KEY MODULE EXT TMP
```

## Tappas

进入Tappas源码顶层目录：

```
$ cd tappas/  
$ ./install.sh --skip-hailort --target-platform vf2  
$ source /home/user/.hailo/tappas/tappas_env
```

## 4. 演示案例

### 系统设置

开机后建议关闭CPU自动调频，在root下执行以下命令：

```
$ su -  
$ echo performance > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

### 开启HailoRT的Monitor（可选）

1. 执行以下命令，开启Hailo monitor，实时监控NPU设备的利用率，使用的模型以及输出的FPS等数据：

```
$ hailortcli monitor
```

Figure 4-1 开启Hailo Monitor



2. 另开一个shell（此shell必须为运行演示程序的shell），指定环境变量：

```
$ export HAILO_MONITOR=1
```

Figure 4-2 指定环境变量

```
user@starfive:~/tappas/apps/h8/gstreamer/vf2/detection$ export HAILO_MONITOR=1  
user@starfive:~/tappas/apps/h8/gstreamer/vf2/detection$ ./detection.sh -i /dev/video4 --network yolov5
```

3. 执行演示案例时，HailoRT检测到此shell进程的HAILO\_MONITOR变量为1，则会实时输出NPU模块的相关监测数据：

Figure 4-3 监测数据

| Device ID    | Utilization (%) | Architecture |
|--------------|-----------------|--------------|
| 0001:01:00.0 | 22.5            | HAIL08       |

| Model              | Utilization (%) | FPS  | PID  |
|--------------------|-----------------|------|------|
| yolov5m_wo_spp_60p | 22.5            | 15.0 | 4456 |

| Model              | Stream                          | Direction | Avg  | Max | Min | Capacity |
|--------------------|---------------------------------|-----------|------|-----|-----|----------|
| yolov5m_wo_spp_60p | yolov5m_wo_spp_60p/input_layer1 | H2D       | 0.50 | 1   | 0   | 4        |
| yolov5m_wo_spp_60p | yolov5m_wo_spp_60p/conv93_132   | D2H       | 0.50 | 1   | 0   | 4        |
| yolov5m_wo_spp_60p | yolov5m_wo_spp_60p/conv84_132   | D2H       | 0.50 | 1   | 0   | 4        |
| yolov5m_wo_spp_60p | yolov5m_wo_spp_60p/conv74_132   | D2H       | 0.50 | 1   | 0   | 4        |

### Tappas演示案例

昉·星光 2下可直接使用Tappas下的部分演示案例。

**Note:**

由于Tappas下的演示案例使用的GStreamer videosink为ximagesink以及ximagesink，而昉·星光 2的Debian使用的Wayland协议，这些演示均需修改其启动脚本，将其中的video\_sink\_element中的ximagesink修改为waylandsink：

**Figure 4-4 修改启动脚本**

```
video_sink_element=$( [ "$XV_SUPPORTED" = "true" ] && echo "ximagesink" || echo "ximagesink")
```

```
video_sink_element=$( [ "$XV_SUPPORTED" = "true" ] && echo "ximagesink" || echo "waylandsink")
```

否则在运行时会产生以下报错：

**Figure 4-6 报错**

```
Setting pipeline to PAUSED ...
Config file doesn't exist, using default parameters
Pipeline is PREROLLING ...
Redistribute latency...
X Error of failed request: BadValue (integer parameter out of range for operation)
Major opcode of failed request: 131 (XInputExtension)
Minor opcode of failed request: 46 ()
Value in failed request: 0xd
Serial number of failed request: 51
Current serial number in output stream: 55
```

- Instance\_segmentation:

```
$ cd /home/user/tappas/apps/h8/gstreamer/vf2/instance_segmentation/
$ ./instance_segmentation.sh -i /dev/video4
```

**Note:**

命令中的/dev/video4指定使用USB Camera。

- Detection:

```
$ cd /home/user/tappas/apps/h8/gstreamer/vf2/detection/
$ ./detection.sh -i /dev/video4 --network yolov5
```

**Note:**

通过--network指定使用yolov5模型。

- Cascading\_networks:

```
$ cd /home/user/tappas/apps/h8/gstreamer/vf2/cascading_networks/
```

- object\_detection\_and\_pose\_estimation用例:

```
$ ./object_detection_and_pose_estimation.sh -i /dev/video4
```

**Note:**

如出现以下报错，请检查是否缺少了TAPPAS\_WORKSPACE环境变量。



Figure 4-7 报错

```
[HailoRT] [error] CHECK failed - Failed opening file, path: /apps/h8/gstreamer/general/cascading_networks/resources/lightface_slim.hef
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] Failed parsing HEF file
[HailoRT] [error] Failed creating HEF
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_EXPECTED failed with status=13
[HailoRT] [error] CHECK failed - Failed opening file, path: /apps/h8/gstreamer/general/cascading_networks/resources/tddfa_mobilenet_v1.hef
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] Failed parsing HEF file
[HailoRT] [error] Failed creating HEF
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
```

**解决方法：**可设置环境变量为`export TAPPAS_WORKSPACE=path_to_tappas/tappas/`，其中的`path_to_tappas`为Tappas源码目录所在路径。

## NpuDetectorLib演示案例

1. 执行以下命令，下载并编译NpuDetectorLib：

```
$ tar -xvf NpuDetectorLib.tar
$ cd NpuDetectorLib
$ cmake -H. -Bbuild -DSHOW_LABEL=ON -DBUILD_TESTER=ON
$ cmake --build build
```

2. 下载所需的资源：

```
$ wget -O models/yolov8s_nms.hef
https://hailo-model-zoo.s3.eu-west-2.amazonaws.com/ModelZoo/Compiled/v2.13.0/hailo81/yolov8s.hef
```

3. 执行以下命令，进行目标识别演示：

```
$ ./build/tests/TestExecutable -i /dev/video4 -m models/yolov8s_nms.json -a yolov8_nms
```

---

## 5. 附录

可点击[链接](#)，购买USB Camera。

