Jetson Nano

[](mailto:Yuri.Fanzhiwei@gmail.com)

Jetson Nano嵌入式板子由NVIDIA英伟达开发，主要用于嵌入式AI即边缘AI方向，在边缘/实时计算以及计算成本上有很大的优势，该Jetson Nano开发板为4GB运行内存的B01开发板，板载ARM架构的CPU和NVIDIA的GPU，整体开发板体积小成本低，开发板价格在￥1000以下，拥有多个模组和对外扩展的GPIO接口。

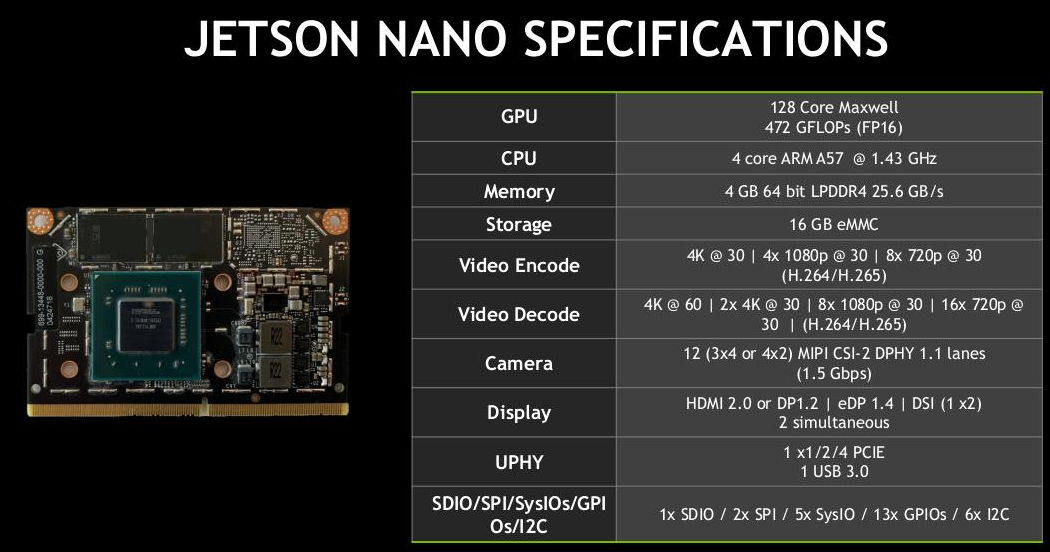
目录

* [Jetson Nano介绍](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/README.md#Jetson-Nano%E4%BB%8B%E7%BB%8D)
  + [硬件介绍](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E7%A1%AC%E4%BB%B6%E4%BB%8B%E7%BB%8D)
  + [软件介绍](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E8%BD%AF%E4%BB%B6%E4%BB%8B%E7%BB%8D)
* [开发配置](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/README.md#%E5%BC%80%E5%8F%91%E9%85%8D%E7%BD%AE)
  + [系统烧录](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E7%B3%BB%E7%BB%9F%E7%83%A7%E5%BD%95)
  + [中文输入法](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E4%B8%AD%E6%96%87%E8%BE%93%E5%85%A5%E6%B3%95)
  + [安装pip3](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E5%AE%89%E8%A3%85pip3)
  + [安装jtop开发板性能监控插件](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E5%AE%89%E8%A3%85jtop%E5%BC%80%E5%8F%91%E6%9D%BF%E6%80%A7%E8%83%BD%E7%9B%91%E6%8E%A7%E6%8F%92%E4%BB%B6)
  + [配置CUDA的系统变量路径](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E9%85%8D%E7%BD%AECUDA%E7%9A%84%E7%B3%BB%E7%BB%9F%E5%8F%98%E9%87%8F%E8%B7%AF%E5%BE%84)
* [功能及性能测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/README.md#%E5%8A%9F%E8%83%BD%E5%8F%8A%E6%80%A7%E8%83%BD%E6%B5%8B%E8%AF%95)
  + [视频压缩编码模块测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E8%A7%86%E9%A2%91%E5%8E%8B%E7%BC%A9%E7%BC%96%E7%A0%81%E6%A8%A1%E5%9D%97%E6%B5%8B%E8%AF%95)
  + [摄像头导入测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E6%91%84%E5%83%8F%E5%A4%B4%E5%AF%BC%E5%85%A5%E6%B5%8B%E8%AF%95)
  + [视频传输测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E8%A7%86%E9%A2%91%E4%BC%A0%E8%BE%93%E6%B5%8B%E8%AF%95)
  + [Jetson Nano和软件后端进行通讯测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###Jetson-Nano%E5%92%8C%E8%BD%AF%E4%BB%B6%E5%90%8E%E7%AB%AF%E8%BF%9B%E8%A1%8C%E9%80%9A%E8%AE%AF%E6%B5%8B%E8%AF%95)
  + [实时视频传输](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E5%AE%9E%E6%97%B6%E8%A7%86%E9%A2%91%E4%BC%A0%E8%BE%93)
  + [安全帽识别功能测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E5%AE%89%E5%85%A8%E5%B8%BD%E8%AF%86%E5%88%AB%E5%8A%9F%E8%83%BD%E6%B5%8B%E8%AF%95)
  + [反光衣识别功能测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E5%8F%8D%E5%85%89%E8%A1%A3%E8%AF%86%E5%88%AB%E5%8A%9F%E8%83%BD%E6%B5%8B%E8%AF%95)
  + [烟火检测功能测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E7%83%9F%E7%81%AB%E6%A3%80%E6%B5%8B%E5%8A%9F%E8%83%BD%E6%B5%8B%E8%AF%95)
  + [口罩检测功能测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E5%8F%A3%E7%BD%A9%E6%A3%80%E6%B5%8B%E5%8A%9F%E8%83%BD%E6%B5%8B%E8%AF%95)
  + [物体检测功能测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E7%89%A9%E4%BD%93%E6%A3%80%E6%B5%8B%E5%8A%9F%E8%83%BD%E6%B5%8B%E8%AF%95)
  + [车牌识别功能测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E8%BD%A6%E7%89%8C%E8%AF%86%E5%88%AB%E5%8A%9F%E8%83%BD%E6%B5%8B%E8%AF%95)
  + [人脸(录入)识别功能测试](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/-###%E4%BA%BA%E8%84%B8(%E5%BD%95%E5%85%A5)%E8%AF%86%E5%88%AB%E5%8A%9F%E8%83%BD%E6%B5%8B%E8%AF%95)
* [总结](https://file+.vscode-resource.vscode-webview.net/c%3A/Users/dell/Desktop/nano/README.md#%E6%80%BB%E7%BB%93)

Jetson Nano介绍

* 硬件介绍

如下图所示为Jetson Nano开发板的官方硬件参数图，可以看到GPU采用的是128核Maxwell的英伟达GPU，CPU采用的是4核的ARM A57芯片，运行内存为4GB，支持H.264和H.265的音视频编解码，最大可支持4K分辨率，可同时读入两个CSI摄像头，同时该板的功耗十分低且有两种模式：一种是20W的最大功率模式，第二种是10W的低功耗模式，其中在最大功率运行下，可使得Nano的CPU和GPU的核心数全部满载运行，在低功耗模式下，板子效率折半，CPU和GPU仅一半核心在工作（另外注意：在20W的运行模式下单靠Jetson Nano自带的被动散热片不足以降低板子产生热量，需要额外散热风扇辅助否则开发板容易陷入宕机状态）

[](mailto:Yuri.Fanzhiwei@gmail.com)

* 软件介绍

Jetson Nano的系统依赖于Ubuntu18.04LTS，但又与真正的Ubuntu有一点点的不一样，大同小异，其系统烧入依赖于将系统镜像烧录SD卡然后将SD插入开发板即可上电使用，Jetson Nano的官方系统打包为Jetpack，Jetpack是集成了Ubuntu系统、CUDA、Cudnn、OpenCV，TensorRT等人工智能方面可能需要用到的库或软件，目前最新的Jetpack版本是JP4.6，但是建议安装JP4.5.1版本，原因是JP4.6是8月份新发布的版本，其他软件可能在兼容性方面还存在不稳定性和不确定性；另外，Jetson Nano系统安装好默认安装了Python3.6和Python2.7，建议不要安装ArchiConda，因为在装完ArchiConda后就会默认安装Python3.7这样的话系统下的Python3会默认使用Python3.7这样导致的问题就是原先系统安装的TensorRT、OpenCV等库都是在Python3.6上在Python3.7下无法运行，而且由于Jetson Nano开发板芯片架构是ARM架构的特殊性，事实上目前许多库在ARM架构上还并没十分完善。

开发配置

* 系统烧录

在[官方](https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#write)中下载Jetson Nano的镜像，并使用[Win32DiskImager](https://sourceforge.net/projects/win32diskimager/" \o "https://sourceforge.net/projects/win32diskimager/)工具烧录到SD卡上，特别注意：需要准备一张大于32G内存的TF卡，因为光是整个系统包含其他依赖库就有大约30G左右，因此建议购买64G内存卡

* 中文输入法

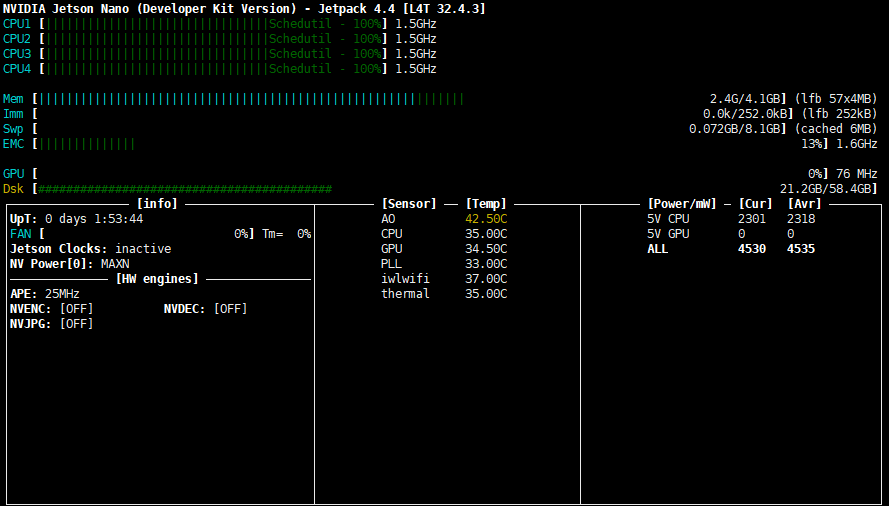
终端输入sudo apt-get install fcitx-googlepinyin命令安装和执行sudo apt remove fcitx-module-kimpanel移除组件使得输入法显示框正常显示，并在设置的语言支持中添加中文简体支持，重新启动开发板即可

* 安装pip3

终端输入sudo apt-get install python3-pip安装pip3  
终端输入pip3 install --upgrade pip升级pip3

* 安装jtop开发板性能监控插件

终端输入sudo -H pip install -U jetson-stats，然后重启开发板终端输入jtop即可查看开发板信息

[](mailto:Yuri.Fanzhiwei@gmail.com)

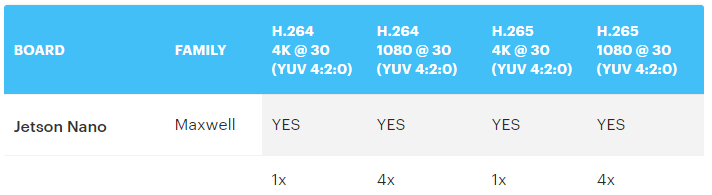
* 配置CUDA的系统变量路径

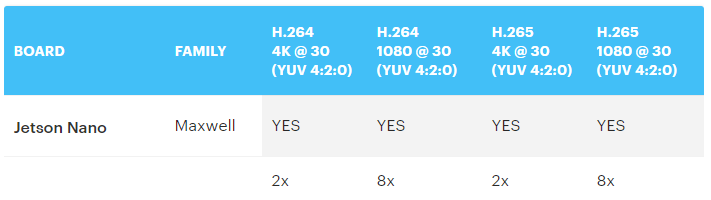
如果不配置系统变量路径在使用CUDA的时候可能出现由于库找不到而报错  
在终端输入sudo gedit ~/.bashrc编辑文件，在文件末尾加入如下CUDA路径  
export PATH=/usr/local/cuda/bin:$PATH export LD\_LIBRARY\_PATH=/usr/local/cuda/lib64:$LD\_LIBRARY\_PATH  
export CUDA\_ROOT=/usr/local/cuda 在终端输入nvcc -V验证CUDA系统变量路径成功加入

功能及性能测试

* 视频压缩编码模块测试

主要测试Jetson Nano的编解码的能力是否符合官方文档所展示，基于1080P和4K的两类分辨率视频进行H.264和H.265的编解码测试，测试内容主要有：1080P，4K视频文件的直接解码测试和从1080P的USB摄像头和4K的CSI摄像头获取图像进行编码再进行解码显示。测试的方式主要是通过Jetson Nano的Gstreamer流媒体框架以及相应的编解码插件直接在命令行进行对视频的编解码，具体编解码测试可参考个人在测试时候记录下来笔记博客[英伟达Jetson：Jetson Nano视频编解码测试](https://blog.csdn.net/qq_43711697/article/details/118938034?spm=1001.2014.3001.5501" \o "https://blog.csdn.net/qq_43711697/article/details/118938034?spm=1001.2014.3001.5501), 从最后的测试结果来看，总的来说，Jetson Nano对于4K视频可以不管在H.264还是H.265下做到很好地编解码，对于4K以上视频进行测试后发现将会超出Nano能力范围无法运行，对于1080P的图像最多可以4路进行编解码，总结来说就是Jetson nano 的编解码能力在4K视频流范围内。(注意：测试的编解码模块都属于Jetson nano的**硬编解码**即使用GPU进行编解码而不是软编解码)

[](mailto:Yuri.Fanzhiwei@gmail.com)

[](mailto:Yuri.Fanzhiwei@gmail.com)

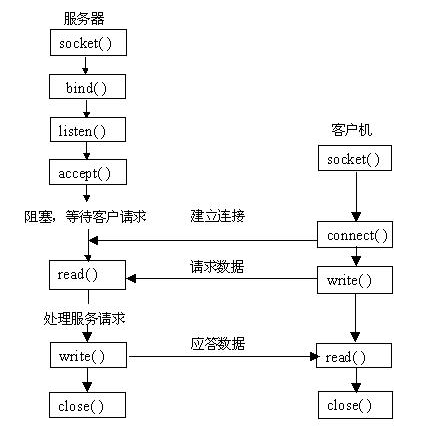
* 摄像头导入测试

测试使用的摄像头分别为最大支持1080P的USB摄像头以及最大支持4K的CSI摄像头，其中USB摄像头不需要安装额外的驱动可以直接插入使用，而对于CSI摄像头需要安装额外的驱动，CSI摄像头型号为索尼IMX 477长焦镜头具体驱动安装参考[NVIDIA Jetson IMX477 HQ RPI V3 camera driver](https://github.com/RidgeRun/NVIDIA-Jetson-IMX477-RPIV3" \o "https://github.com/RidgeRun/NVIDIA-Jetson-IMX477-RPIV3)教程；具体摄像头导入使用Python-opencv库调用摄像头读取每一帧，具体的摄像头性能方面在1080P输入状态都能达到30FPS其中CSI摄像头可达到最高60FPS。

* 视频传输测试

项目中主要测试Jetson Nano和云服务器之间进行视频的传输，视频文件的时长大约在1分钟以内，视频文件传输的主要方式采用Socket传输即套接字传输，它是计算机之间进行通信的一种约定或一种方式，通过socket约定，一台计算机可以接收其他计算机的数据，也可以向其他计算机发送数据，通过Jetson Nano来连接云服务器的公网IP并发送到约定好的特定端口号上，然后云服务器端运行文件接收程序不断监听本地端口号，当监听到后开始接收文件，具体使用可参考个人GitHub记录[Socket-Python](https://github.com/FanDady/Socket-Python" \o "https://github.com/FanDady/Socket-Python)

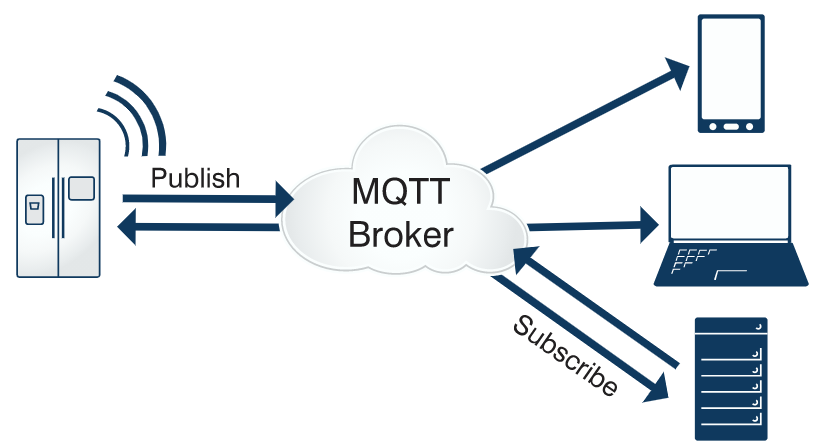
| **测试算法** | **分辨率** | **时长** | **压缩格式** | **文件大小** | **传输时长** | **视频帧率** | **总体码率** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| YoloV3-tiny | 640x480 | 1min | MPEG-4 | ≈6Mb | 0.32s | 15FPS | 733kb/s |
| YoloV3-tiny | 1920x1080 | 1min | MPEG-4 | ≈10Mb | 0.91s | 10FPS | 1369kb/s |

[](mailto:Yuri.Fanzhiwei@gmail.com)

* Jetson Nano和软件后端进行通讯测试

Jetson Nano在检测到对应目标后需要发送信息到软件上并显示，因此软件后端和Jetson Nano之间采用物联网常用的MQTT协议进行通信，MQTT是一种基于发布/订阅模式的"轻量级"通讯协议，该协议构建于TCP/IP协议上，并且MQTT可以以极少的代码和极低的带宽，为连接远程设备提供实时可靠的消息服务。大致的框架是Jetson Nano和软件后端之间通过部署了MQTT协议的云服务器的中介进行信息转换交流，具体的部署方法请参考个人GitHub记录[MQTT-IOT](https://github.com/FanDady/MQTT-IOT" \o "https://github.com/FanDady/MQTT-IOT)

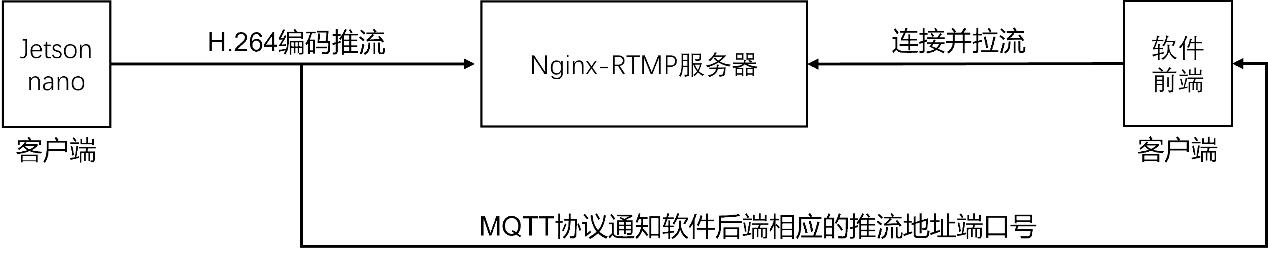
| **测试协议** | **通讯内容** | **通讯格式** | **通讯文本类型** | **通讯延迟** | **基础架构** |
| --- | --- | --- | --- | --- | --- |
| MQTT | 消息/文件路径 | JSON | 字典型 | 基本实时 | TCP/IP |

[](mailto:Yuri.Fanzhiwei@gmail.com)

* 实时视频传输测试

实时视频传输方面考虑使用直播的框架来进行搭建，由于软件前端方面的限制因此只能使用RTMP协议进行视频流的传输，所以基本的想法或框架就是Jetson Nano上对摄像头拍摄的视频流进行硬件编码然后实时推流到云服务器上，云服务器部署好Nginx+RTMP以实现云服务器作为视频流转发的流媒体处理中心，软件前端对服务器上转发的视频流进行拉流并显示，具体的云服务器部署以及推流实现请参考个人GitHub记录[LiveVideo-RTMP](https://github.com/FanDady/LiveVideo-RTMP" \o "https://github.com/FanDady/LiveVideo-RTMP)，另外，注意到目前RTMP协议并不支持H.265编码因此，在传输的时候仅通过H.264编码再推流，并且为了达到延迟更低更稳定的状态建议使用Jetson Nano的硬编码模块，除此之外，在Jetson进行推流的时候有两种方式，第一种是ffmpeg多媒体框架方式，但其仅能在nano上实现软编码原因是似乎没法调用英伟达的硬编码模块插件并且延迟较高，但是用的广泛拓展性好，另外一种是Gstreamer多媒体框架，英伟达官方的Deepstream就是基于Gstreamer开发的，因此Jetson nano对Gstreamer的支持性较好，支持硬编码延迟较低，但是相较于ffmpeg难度大。(ffmpeg和gstreamer都可以用C++源码开发但是由于时间和能力有限因此仅采用命令行快速的方式进行使用)

| **多媒体框架** | **编码格式** | **直播延迟** | **稳定性** | **编码方式** | **视频分辨率** | **编码后整体帧数** | **编码后带宽** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ffmpeg | H.264 | 平均在9s左右 | 有时会卡顿 | 软编码 | 1080P | 30FPS | 忘记记录 |
| gstreamer | H.264 | 平均在6s左右 | 不易卡顿 | 硬编码 | 1080P | 30FPS | 4Mbps左右 |

[](mailto:Yuri.Fanzhiwei@gmail.com)

* 安全帽识别功能测试

安全帽识别的算法主要基于YoloV5s，检测类别主要有两类头部和安全帽，以此来判别人是否佩戴安全帽，其中YoloV5s网络模型的参数量大小大约在7.3M左右可在Nano上正常运行，如果使用超过YoloV5s的YoloV5m或以上，在Jetson Nano上会出现帧率极低或者Nano板子的运行内存不足导致系统宕机，因此在Nano上不适宜选择参数量过高的网络模型进行推理计算即使在有TensorRT支持的情况下也无法为模型提升多少。具体的部署请参考个人Github记录[Helmet-Detection-YoloV5](https://github.com/FanDady/Helmet-Detection-YoloV5" \o "https://github.com/FanDady/Helmet-Detection-YoloV5)

| **功能** | **算法** | **模型参数量** | **算法内图像处理大小** | **是否使用TensorRT** | **实时帧率** |
| --- | --- | --- | --- | --- | --- |
| 安全帽识别 | YoloV5s | 7.3M | 640x640 | 是 | 9FPS左右 |

[](mailto:Yuri.Fanzhiwei@gmail.com)

* 反光衣识别功能测试

反光衣识别的算法主要基于YoloV5s，检测类别主要有两类反光衣和其他衣服，以此来判别人是否穿戴反光衣，其中YoloV5s网络模型的参数量大小大约在7.3M左右可在Nano上正常运行，如果使用超过YoloV5s的YoloV5m或以上，在Jetson Nano上会出现帧率极低或者Nano板子的运行内存不足导致系统宕机，因此在Nano上不适宜选择参数量过高的网络模型进行推理计算即使在有TensorRT支持的情况下也无法为模型提升多少。具体的部署请参考安全帽检测，任务不同部署类似。

| **功能** | **算法** | **模型参数量** | **算法内图像处理大小** | **是否使用TensorRT** | **实时帧率** |
| --- | --- | --- | --- | --- | --- |
| 反光衣识别 | YoloV5s | 7.3M | 640x640 | 是 | 9FPS左右 |

[](mailto:Yuri.Fanzhiwei@gmail.com)

* 烟火检测功能测试

烟火检测的算法主要基于YoloV5s，检测类别主要有一类烟火，以此来判别是否存在火灾，其中YoloV5s网络模型的参数量大小大约在7.3M左右可在Nano上正常运行，如果使用超过YoloV5s的YoloV5m或以上，在Jetson Nano上会出现帧率极低或者Nano板子的运行内存不足导致系统宕机，因此在Nano上不适宜选择参数量过高的网络模型进行推理计算即使在有TensorRT支持的情况下也无法为模型提升多少。具体的部署请参考安全帽检测，任务不同但部署类似。

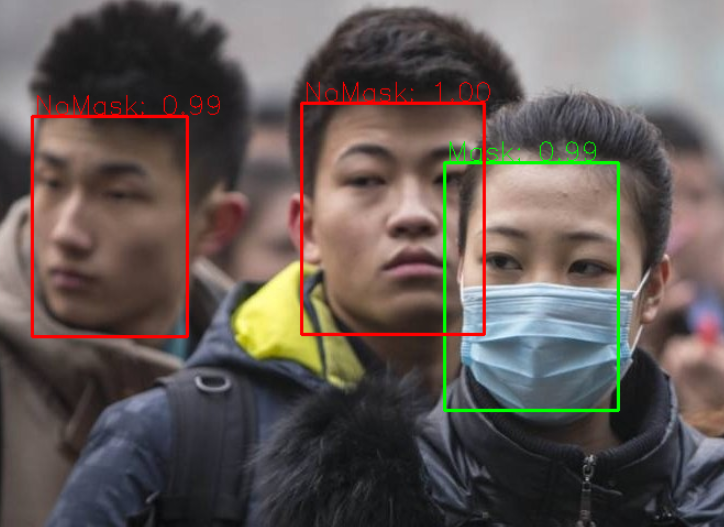
| **功能** | **算法** | **模型参数量** | **算法内图像处理大小** | **是否使用TensorRT** | **实时帧率** |
| --- | --- | --- | --- | --- | --- |
| 烟火检测 | YoloV5s | 7.3M | 640x640 | 是 | 9FPS左右 |

[](mailto:Yuri.Fanzhiwei@gmail.com)

* 口罩检测功能测试

口罩检测的算法主要基于[Nanodet](https://github.com/RangiLyu/nanodet" \o "https://github.com/RangiLyu/nanodet)算法，检测类别主要有两类分别是有佩戴口罩和无佩戴口罩，Nanodet-m网络模型的参数量大小大约在0.95M左右可在Nano上快速运行，nanodet和yolo系列检测算法相比，在嵌入式端更具有优势，因为nanodet的网络模型十分轻巧并且高效，在检测速度上十分快速，因此在边缘计算上十分适合，这里使用的是nanodet中最轻量的模型nanodet-m，在Jetson Nano上有着不错的表现，在没有TensorRT的加速下完全可以实现实时检测。另外，还使用[WIDER face](https://github.com/AIZOOTech/FaceMaskDetection/blob/master/README-zh.md" \o "https://github.com/AIZOOTech/FaceMaskDetection/blob/master/README-zh.md)算法进行检测，检测的效果同样较为可观，在检测表现上和nanodet不相上下，在模型参数上仅有1.01M

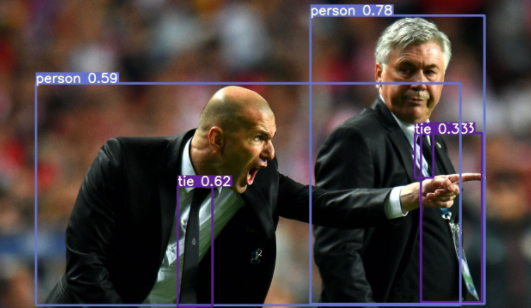
| **功能** | **算法** | **模型参数量** | **算法内图像处理大小** | **是否使用TensorRT** | **实时帧率** |
| --- | --- | --- | --- | --- | --- |
| 口罩检测 | Nanodet-m | 0.95M | 320x320 | 否 | 17FPS左右 |
| 口罩检测 | WIDER face | 1.01M | 260x260 | 否 | 18FPS左右 |

[](mailto:Yuri.Fanzhiwei@gmail.com)

* 物体检测功能测试

物体检测算法，这里主要是测试一下YoloV3-tiny的能力，YoloV3-tiny-416是属于yolov3中最轻量的模型，参数量约为8.8M，虽然参数量较大但是在TensorRT加速下的检测速度还是较快的，对比使用TensorRT加速前后提升了大概有5帧

| **功能** | **算法** | **模型参数量** | **算法内图像处理大小** | **是否使用TensorRT** | **实时帧率** |
| --- | --- | --- | --- | --- | --- |
| 物体检测 | yolov3-tiny | 8.8M | 416x416 | 否 | 9FPS左右 |
| 物体检测 | yolov3-tiny | 8.8M | 416x416 | 是 | 14FPS左右 |

[](mailto:Yuri.Fanzhiwei@gmail.com)

* 车牌识别功能测试

车牌识别算法总体上在nano上实现实时检测的难度较大，因为车牌识别事实上存在两个目标一个是车牌检测另一个是车牌信息识别，因此前者检测车牌相对容易但是加上实时检测出车牌信息，这样所带来的结果是检测速度下降，特别是在nano上，检测的实时性大大降低，这里测试的算法是使用LPDNet+LPRNet联合起来实现车牌识别，并且经过TensorRT加速后其帧率可达到接近9FPS，总体来说表现还是不乐观。

| **功能** | **算法** | **模型参数量** | **算法内图像处理大小** | **是否使用TensorRT** | **实时帧率** |
| --- | --- | --- | --- | --- | --- |
| 车牌识别 | LPDNet+LPRNet | 未知 | 640x480 | 是 | 9FPS左右 |

* 人脸(录入)识别功能测试

人脸(录入)识别主要是识别提前录好人像的人脸，并对其识别出身份，这里主体使用的是FaceNet，FaceNet主要用来识别人脸的信息，首先是通过MTCNN检测出人脸区域再通过FaceNet进行识别，因此人脸识别的步骤大概也是经过两部进行，总体运行帧率不高，全部的参数量至少在10M以上，具体数值未去仔细计算仅仅计算了大概参数量。

| **功能** | **算法** | **模型参数量** | **算法内图像处理大小** | **是否使用TensorRT** | **实时帧率** |
| --- | --- | --- | --- | --- | --- |
| 人脸(录入)识别 | MTCNN+FaceNet | 10M以上 | 1920x1080 | 否 | 3FPS左右 |

总结

使用Jetson Nano作为边缘计算板的话成本上固然是十分具有优势，但是也因此会失去更强劲的计算能力，因此在Jetson Nano上如果要进行深度学习推理计算达到实时检测的效果(至少15FPS起)，其网络模型的参数量不应该太大，从目前的测试情况来看参数量应该小于8M，且输入网络进行处理的图像大小不应该太大，也需要进行预处理在标清这个分辨率范围以内，另外，在Jetson Nano上进行深度学习推理的模型建议要使用英伟达的TensorRT加速，其加速效果十分明显，如果是在英特尔的硬件平台上可使用OpenVINO，其类似英伟达的TensorRT。此外，Jetson Nano在进行视频编解码和传输通信方面基本没有问题。所以从目前的使用情况来看，Jetson Nano的主要优点在于：成本低性价比高、性能强大、体型较小、功耗较低、可更换SD卡来进行快速的系统变迁或者功能更换；主要缺点在于：对于参数量较大一些的网络模型推理性能较差、版本库依赖冲突等潜在问题较多以及版本兼容性问题、运行内存较低。