HAILO

Empowering Intelligence

Hailo AI SW Suite User Guide

Release 2026-01
8 January 2026

Table of Contents

Disclaimer and Proprietary Information Notice

Copyright

This version of the document supersedes all previous versions.

General Notice

Hailo, to the fullest extent permitted by law, provides this document "as-is" and disclaims all warranties, either express or implied, statutory or otherwise, including but not limited to the implied warranties of merchantability, non-infringement of third parties' rights, and fitness for particular purpose.

Although Hailo used reasonable efforts to ensure the accuracy of the content of this document, it is possible that this document may contain technical inaccuracies or other errors. Hailo assumes no liability for any error in this document, and for damages, whether direct, indirect, incidental, consequential or otherwise, that may result from such error, including, but not limited to loss of data or profits.

The content in this document is subject to change without prior notice and Hailo reserves the right to make changes to content of this document without providing a notification to its users.

1. Getting Started Guide

The Hailo SW products are a set of frameworks and tools that enable you to compile, run, and evaluate neural networks on Hailo devices:

1. Dataflow Compiler (Model conversion and compilation to Hailo binary format)
2. HailoRT (Runtime environment and driver for running networks and interacting with Hailo devices)
3. Model Zoo (Pre-trained vision models to run and evaluate on Hailo devices)
4. Model Zoo GenAI (pre-trained GenAI models and example applications to run and evaluate on Hailo devices)
5. TAPPAS (Examples and multi-network pipelines for vision models)

Although you can install each product separately, Hailo releases a quarterly software suite in which all product versions are aligned. Therefore, using the Hailo AI SW Suites ensures the best compatibility.

For information about the latest suite version (recommended), see *Latest SW Suite Features*. For installation guide, see *Suite Installation*. For compatibility between Hailo's product (and suites) versions see *Version Compatibility Table*.

1.1. Suite Components

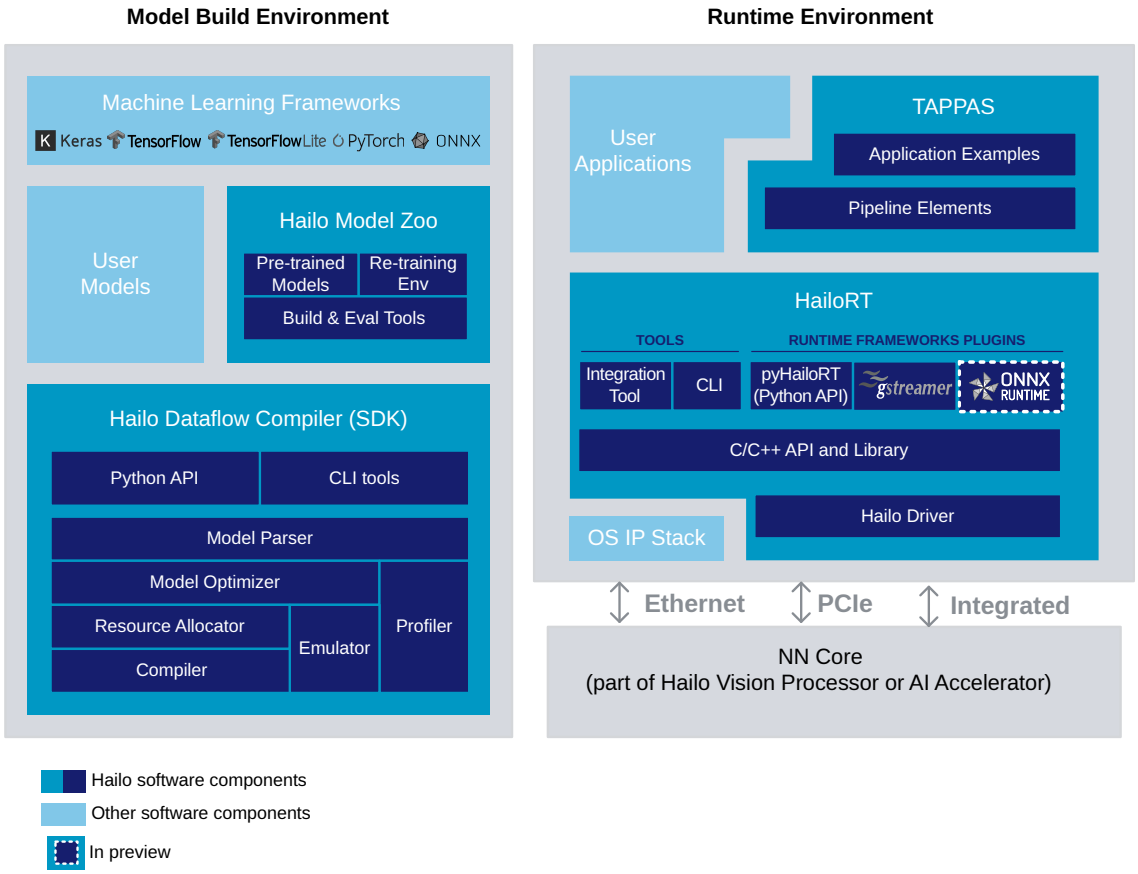

Figure 1. Detailed block diagram of Hailo software packages

The Hailo SW components are used in the following manner:

- **On the Model build environment:**

  - Hailo Dataflow Compiler is used to compile a trained model to run on Hailo devices.
  - Hailo Model Zoo contains a large database of pre-trained models that are validated to work with best performance on Hailo devices.

    It also contains a retraining environment.

- **On the Runtime environment:**
  - HailoRT is used to load the compiled model to a Hailo device and interact with it (using the PCIe driver).
  - TAPPAS includes complete examples and demos of using HailoRT to create full pipelines on top of Hailo devices.

1.1.1. Dataflow Compiler

The Dataflow Compiler API is used for compiling models to Hailo binaries.

The input of the Dataflow Compiler is a trained Deep Learning model.

The output is a binary file which is loaded onto the Hailo device.

1.1.2. HailoRT

The HailoRT API is used for deploying the built model on the target device. This library is used by the runtime applications.

It implements a userspace C/C++ API that is called from the user's applications. It allows both to control the Hailo device and to send and receive data from it. It supports the PCIe interface.

The HailoRT Python package wraps the C/C++ API and exposes a Python interface that allows to load models to the device and send and receive data from it.

It also includes a PCIe driver is required when working via the PCIe interface. It links the HailoRT library and the device. It also loads the device's firmware when working through this interface.

Finally, Hailo's Yocto layer allows the user to integrate Hailo's software into an existing Yocto environment. It includes recipes for the HailoRT library, Python package and the PCIe driver.

1.1.3. Hailo Model Zoo

Hailo Model Zoo provides pre-trained models for high-performance deep learning applications.

Using the Hailo Model Zoo you can measure the full precision accuracy of each model, the optimized accuracy using the Hailo Emulator and measure the accuracy on the Hailo devices.

Finally, you will be able to generate the Hailo Executable Format (HEF) binary file to speed-up development and generate high quality applications accelerated with Hailo.

The models are optimized for high accuracy on public datasets and can be used to benchmark the Hailo model optimization scheme.

1.1.4. TAPPAS

TAPPAS is Hailo's set of full application examples, implementing pipeline elements and pre-trained AI tasks.

Demonstrating Hailo's system integration scenario of specific use cases on predefined systems (software and Hardware platforms). It can be used for evaluations, reference code and demos:

- Accelerating time to market by reducing development time and deployment effort
- Simplifying integration with Hailo's runtime SW stack
- Providing a starting point for customers to fine-tune their applications

1.2. Where to begin

Consider that the Hailo AI Suite is **intended for the model build platform**, which is a host that meets the *system requirements*. The host could also serve as the runtime platform, but not necessarily.

1. In case an Hailo accelerator is used, install it on the PC.

   Note: In case the runtime platform is different from the build platform, for example when using the Hailo-15H VPU (or an Hailo accelerator that is installed on a different platform), the models that are compiled using the Model Zoo or the Dataflow Compiler should be transferred to the device, and run using the on-board HailoRT package.

2. Install the latest suite: *Suite Installation*.
3. If an Hailo accelerator is connected, Hailo recommends to begin with TAPPAS usage guide to see various real-time demos in action using your PC's camera.

   Note: TAPPAS also comes pre-installed in the Hailo-15 Vision Processor Software Package.

   Note: Behind the scenes, TAPPAS uses HailoRT to interact with the device. It also uses the GStreamer framework.

4. It is then recommended to try our **Pre-Trained Models**, with the Hailo Model Zoo. With an easy-to-use CLI interface, it allows you to convert, retrain and compile models to your Hailo device. If an accelerator device is connected, it allows you to run and evaluate the performance and accuracy on the Hailo hardware.

   Note: Behind the scenes, the Hailo Model Zoo uses the Dataflow Compiler for compilation, and HailoRT Python API to interact with the device.

5. If a Hailo Model Zoo model is a good fit for your application, you can use your own dataset for retraining using Hailo Retraining Docker Containers.
6. For your **Custom Model**, use the Dataflow Compiler for optimization and conversion to Hailo binary format. Refer to the Dataflow Compiler tutorials section.

   Note: Use the Dataflow Compiler CLI tools or Python APIs for your custom models, not the Model Zoo CLI.

7. For building real-time applications, read the HailoRT Tutorials section and API Reference.

Note: You can also use the TAPPAS examples code as a reference (for GStreamer framework integration or C/CPP API).

8. (Optional) For a Hailo Model Zoo GenAI user-friendly web interface, you can install and use Open WebUI as an optional component. See *Open WebUI (Optional)* in the installation guide for setup instructions.

2. 2026-01 Hailo AI SW Suite - New Features

2.1. Dataflow Compiler

General

- Updated the LoRA tutorial notebook to use an LLM base model quantized with group-wise quantization.

Model Optimization

- Multiple bug fixes and accuracy improvements.

Compiler

- Bug fixes and general compilation improvements.
- RAM optimizations for large model compilations. LoRA adapter compilation now requires up to 64GB of RAM (instead of 160GB in the previous release). Please note that optimization stage still requires up to 160GB of CPU RAM.

Parser

- Added support for ONNX opset 21.
- Added a new CLI flag to disable the automatic use of *onnx-simplifier* during parser retry attempts. Run *hailo parser onnx –help* for more details.

2.2. HailoRT

General

- Performance improvements of 5-10% for vision models.
- Added support for Windows OS in HailoRT for Hailo-10H devices.
  - Supported on Windows 10 and Windows 11 (64-bit).
  - Supported on PCIe interface only.
  - See the *HailoRT Windows Installation Guide* for more information.

GenAI

- Added support for language detection and repetition penalty for Speech2Text API (Whisper models)
  - Automatically detect the language of the audio input if not specified.
  - Control over the generation process to prevent the repetition of tokens with the new parameter *repetition_penalty* in *Speech2TextGeneratorParams*.
  - See *Speech2Text tutorial* in HailoRT for examples.
- Added support for tools calling in LLM API
  - C++ *generate()* API and *hailo_platform.pyhailort.pyhailort.LLM.generate* overloads for direct generation with JSON structured prompts and tools without explicitly creating generators.
  - C++ *write()* API overloads for writing prompts and tools to the generator.
  - See *LLM tutorial* in HailoRT for examples.
- Added video support for VLM API
  - Added support for `input_videos` parameter for video input in *VLMGenerator*.
  - Videos are represented as a vector of raw frames, with each video being a `std::vector<MemoryView>`
  - Structured prompts support `{"type": "video"}` content type for video placeholders

  - See *VLM tutorial* in HailoRT for examples

PyHailoRT

- Added API for Speech2Text language detection and repetition penalty - see *Speech2Text tutorial* in HailoRT
- Added video support for VLM API
  - Added support for `videos` parameter for video input in *VLM.generate()* and *VLM.generate_all()*
  - Videos are represented as a list of lists of numpy arrays, with each video being a list of frames
  - Structured prompts support `{"type": "video"}` content type for video placeholders
  - See *VLM tutorial* in HailoRT for examples

InferModel Examples

- Added *Multi Process Example* which demonstrates how to run multiprocess inference on Hailo-10H devices, using the HailoRT C++ API.

2.3. Hailo Model Zoo

- New Models:
  - SigLIP - Released siglip_l_16_256_image_encoder, siglip_l_16_256_text_encoder, siglip2_l_16_256_image_encoder, siglip2_l_16_256_text_encoder, siglip_b_16_image_encoder, siglip_b_16_text_encodersiglip_b_16_text_encoder
  - YOLOv12 - Released yolov12n
  - PaddleOCR-v5 - Text detection and recognition models - released paddle_ocr_v5_mobile_detection, paddle_ocr_v5_mobile_recognition
  - StereoNet - stereo depth estimation model - now also available for Hailo-10H/15H

2.4. Hailo Model Zoo GenAI

- Added Llama-3.2-1B-Instruct model.
- Added Whisper-small model.
- Added Qwen2-1.5B-Instruct-Function-Calling-v1 model (LoRA finetuned adapter for function-calling).
- Added VLM image encoders: Qwen2-VL-Image-Encoder-7B, Qwen2-VL-Image-Encoder-2B, and Qwen3-VL-Image-Encoder-2B.
- Removed support from Llama-3.2-3B-Instruct model.
- Accuracy improvements for the Qwen2-VL-2B-Instruct model which now uses group-wise quantization.
- Renamed model *qwen2.5-instruct:1.5b* to *qwen2.5:1.5b*. In order to pull and run the model, use the new name *qwen2.5:1.5b*.
- Renamed model *deepseek_r1_distill_qwen:1.5b* to *deepseek-r1:1.5b*. In order to pull and run the model, use the new name *deepseek-r1:1.5b*.
- Load time improvements across all models. For example, loading time for *qwen2.5:1.5b* reduced by ~25%.
- Added optional Open WebUI support for Hailo Model Zoo GenAI. Open WebUI provides a user-friendly web interface for interacting with Hailo-Ollama server. Open WebUI is not part of the suite installation, but installation instructions are provided in *Open WebUI (Optional)*.

2.5. TAPPAS

- This release supports both HailoRT v4.23.0 (Hailo-8) and HailoRT v5.2.0 (Hailo-10H)

2.6. Integration Tool

Initial release of the Integration Tool for Hailo-10H devices.

- Currently supports PCIe devices only.
- Supported on Linux only.

3. Previous Suite Versions

3.1. 2025-10 Hailo AI SW Suite

Note: Starting July 2025, this suite will support only the **Hailo-10** and **Hailo-15** device families. Support for Hailo-8, Hailo-8R, and Hailo-8L devices will be available in a separate dedicated suite for Hailo-8 family.

3.1.1. Dataflow Compiler

General

- Added support for Ubuntu 24.04, and Python 3.12
- Ubuntu 20.04, and Python 3.8 are no longer supported
- Tensorflow version was updated to 2.18.0 (CUDA 12.5.1, Cudnn 9.10)

Compiler

- Hailo-15H and Hailo-10H supports new numerical modes: a16_w8 and a16_w4 (preview)

Parser

- Deprecated support for parsing TensorFlow 1.x/2.x models (.ckpt/.pb) using all parsing APIs. See DFC user guide for more information and guidelines for moving to Tensorflow Lite

Package Updates

- Updated CUDA requirement to version 12.5.1, CuDNN to version 9.10
- Updated Nvidia driver requirement to version 555+

3.1.2. HailoRT

General

- Added support for Ubuntu 24.04, Python-3.12 and Python-3.13
- Removed support for Ubuntu 20.04, Python-3.8 and Python-3.9
- Removed the `hailo` command line tool, use `hailortcli` instead

GenAI

- Added support for Speech-to-Text API, and a HailoRT C++ speech-to-text tutorial
- Added a warning message when the GenAI-model's conversation context is full. In such case it is recommended to clear the context
- Added memory optimization support for LLM and VLM APIs to enable running large models on small-memory devices
- Added support for aborting an ongoing generation for LLM, VLM and Text2Image APIs

PyHailoRT

- Added API for LLMs
- Added API for VLMs
- Added API for Text2Image

3.1.3. Hailo Model Zoo

New Models

- YOLOv7x - yolov7x
- DepthAnything - depthanything_vits, depthanything_v2_vits - Zero-shot depth estimation models
- CLIP - Performance improvements for clip image encoders
- TinyCLIP - TinyCLIP family - Contrastive Language-Image Pre-training models (image & text encoders)
- YOLO-World - yolo_world_v2s - Zero-shot object detection model

3.1.4. Hailo Model Zoo GenAI

- Added Speech-to-Text support - supported models include Whisper-Base for transcription and translation
- Accuracy improvements for the DeepSeek model: DeepSeek-R1-Distill-Qwen-1.5B
- Removed Qwen2.5-1.5B-Instruct model
- Bug fixes in Hailo-Ollama REST API

3.1.5. TAPPAS

- This release supports both HailoRT v4.23.0 (Hailo-8) and HailoRT v5.1.0 (Hailo-10H)
- Detection example application now supports `--arch` (Hailo-8/Hailo-10H)
- Added support Python 3.13
- Added Hailo-10H HEF downloads
- Removed some redundant CLI arguments

3.2. 2025-07 Hailo AI SW Suite

3.2.1. Dataflow Compiler

Note: Starting July 2025, support for Hailo-8, Hailo-8R, and Hailo-8L devices will be available only in Dataflow Compiler version 3.x. Dataflow Compiler version 5.x will continue to support only the Hailo-10 and Hailo-15 device families.

General

- Hailo Dataflow Compiler now supports Hailo-10H and Hailo-15L.
- The Hailo Dataflow Compiler now supports compilation of LoRA adapters for generative AI models, including LLMs (large language models), SD (stable diffusion) and VLMs (vision language models) exclusively on Hailo-10H.
- For known limitations in Hailo-15L (expected to be solved in future Dataflow Compiler releases), see *Known Issues*.

Model Optimization

- Added support for LoRA (Low-Rank Adaptation) in the model optimization process.

Post Processing

- NMS is supported only with engine=cpu for the following meta-architectures: SSD, YOLOv5, YOLOv5-SEG, YOLOX, and YOLOv8.

- NMS is supported only with engine=nn_core for YOLOV6 tag 0.2.0 and above.
- Centernet is no longer supported.

3.2.2. HailoRT

Note: Starting July 2025, support for Hailo-8, Hailo-8R, and Hailo-8L devices will be available only in HailoRT version 4.x. HailoRT version 5.x will continue to support only the Hailo-10 and Hailo-15 device families.

General

- HailoRT now supports generative AI, including LLMs (large language models), Text-to-image (stable diffusion) and VLMs (vision language models) exclusively on Hailo-10H.
- HailoRT now supports Hailo-10H and Hailo-15L.
- For known limitations in Hailo-15L (expected to be solved in future HailoRT releases), see *Known Issues*.

CLI

- Changed default run mode of `hailortcli run2` to `full_async`.

3.2.3. Hailo Model Zoo

New Models

- SigLip - SigLip2-base-32 - Contrastive Language-Image Pre-training model

3.2.4. Hailo Model Zoo GenAI

- Initial release of the Hailo Model Zoo GenAI.
- Initial release of the Hailo-Ollama REST API.
- Support for large language models (LLMs), including Qwen2-1.5B-Instruct, Qwen2.5-1.5B-Instruct, Qwen2.5-Coder-1.5B-Instruct and DeepSeek-R1-Distill-Qwen-1.5B.
- Support for image generation (Stable Diffusion), including StableDiffusion-1.5.
- Support for vision-language models (VLMs), including Qwen2-VL-2B-Instruct.

3.2.5. TAPPAS

- All example applications, except the object detection application, are now maintained at Hailo Applications.
- Added support for Ubuntu 24.04 & Python 3.12.
- This release supports both HailoRT v4.22.0 (Hailo-8) and HailoRT v5.0.0 (Hailo-10).

4. Suite Installation

The suite can be installed in one of three ways:

- Using all-in-one docker file (see: *Docker installation*)
- Using all-in-one self extracted executable (see: *Self Extracted Executable*)
- Manual installation of the packages in a defined order (see: *Manual installation*)

For interaction with Hailo Devices, a physical connection is required (although not required for installation):

- PCIe interface for the evaluation board
- M.2 connector for the M.2 board

Table 1. Hailo-10H M.2 boards

For the full compatibility table, see *Release versions compatibility*.

4.1. Docker installation

Note: For more information about Docker containers, see *Working With Docker Containers*.

4.1.1. System requirements

Note: Ubuntu 20.04 is no longer supported in this versions.

In case of suite installation as a Docker file, the following are required:

1. Ubuntu 22.04/24.04, 64 bit
2. 16+ GB RAM (32+ GB recommended)
3. Docker package, either docker.io 20.10.07 (from Ubuntu repo), or docker-ce 20.10.6 (from Docker website).

Add the user to the docker group with the following steps:

1. Add your user to the docker group:

```
sudo usermod -aG docker ${USER}
```

2. Log out and log in.

In case you want to use Nvidia GPUs for hardware emulation and quantization, you also need to install:

1. Nvidia's Pascal/Turing/Ampere GPU architecture (such as Titan X Pascal, GTX 1080 Ti, RTX 2080 Ti, or RTX A4000)
2. GPU driver version 555
3. nvidia-docker2. It can be installed by running the following commands:

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
&& curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-
↪key add - \
&& curl -s -L \
https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.
↪list \
| sudo tee /etc/apt/sources.list.d/nvidia-docker.list

sudo apt-get update
sudo apt-get install -y nvidia-docker2
sudo systemctl restart docker
```

4.1.2. Running the docker file

1. Download the suite from Developer Zone
2. Download PCIe driver from Developer Zone
3. Install PCIe driver by running the following command:

```
sudo dpkg -i <pcie_driver>.deb
```

4. Reboot the computer
5. Extract the suite archive, then run the script - which opens a new container, and attaches to it ("gets inside it"):

```
unzip hailo_ai_sw_suite_<version>.zip
./hailo_ai_sw_suite_docker_run.sh
```

6. Run *pip list | grep hailo* to see the Hailo packages that are installed in the activated virtual environment
7. Run *hailo -h* to see a list of the available CLI commands of Hailo Dataflow Compiler and HailoRT

Optional: Open WebUI Installation for Hailo Model Zoo GenAI

If you want to use Open WebUI with Hailo Model Zoo GenAI for your Docker installation, you can install it as a separate Docker container. To install and run Open WebUI, execute the following command:

```
docker run -d --net=host -e OLLAMA_BASE_URL=http://127.0.0.1:8000 -v open-webui:/
↪app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-
↪webui:main
```

After installation, you can access Open WebUI at http://localhost:8080.

For more details and usage with Hailo-Ollama server, see *Open WebUI (Optional)*.

You can exit the docker by using the *exit* command, and then attach to it using the commands shown on *Working With Docker Containers*.

Another option is to use the script's additional options:

- **Resume** – go back to the existing container:

  ```
  ./hailo_ai_sw_suite_docker_run.sh --resume
  ```

- **Override** – delete the existing container and create new one:

  ```
  ./hailo_ai_sw_suite_docker_run.sh --override
  ```

Note: We have created a shared folder between the host system and the docker system. The path inside the docker is `/local/shared_with_docker/`, and the path outside is `./shared_with_docker` folder that is created on the same directory where hailo_ai_sw_suite_docker_run.sh is run.

Note: To validate that the PCIe driver was installed successfully, run *lspci | grep Co-processor*. For more information refer to *HailoRT User Guide / Installation / Validating the PCIe driver was successfully installed on Linux*.

4.1.3. Docker Suite Upgrade

1. Copy all private files from the currently used container in case those will be needed in the new container.
2. Make sure to close previously activated container.
3. Follow the *Docker installation* section in order to set up the new Hailo AI SW Suite docker container.
4. Copy the previously copied files from the previously used container to the newly set up container.

4.2. Self Extracted Executable

Hailo AI SW Suite self extracted executable will install all SW suite components directly into the system.

Note: TAPPAS component requires a dedicated flag.

4.2.1. System requirements

The following requirements are needed for the installation:

1. Ubuntu 22.04/24.04, 64 bit
2. 16+ GB RAM (32+ GB recommended)
3. Python 3.10/3.11/3.12
4. `python3-dev`, `python3-dev`, `python3-setuptools`, `python3-pip` and `python3-virtualenv` (according to the Python version), `python3-tk`, `graphviz` and `libgraphviz-dev` packages. Use the command `sudo apt-get install PACKAGE` for installation
5. build-essential package (needed for compiling the PCIe driver)

6. For TAPPAS: `ffmpeg`, `x11-utils`, `python-gi-dev`, `python3-gi`, `libcairo2-dev`, `libgirepository1.0-dev`, `rapidjson-dev`, `libzmq3-dev`, `git`, `Gstreamer`, `pygobject`, `libopencv-dev`, `gir1.2-gst-plugins-bad-1.0` apt packages, 11. (Optional) bison, flex, libelf-dev and dkms packages (needed to register the PCIe driver using DKMS) 12. (Optional) cmake (needed for compiling the HailoRT examples)

To install TAPPAS apt packages, run the following:

```
sudo apt-get install -y ffmpeg x11-utils python-gi-dev rsync git python3-opencv
↪rapidjson-dev \
python3-gi libgirepository1.0-dev libzmq3-dev libzmq5 libcairo2-dev \
libgstreamer1.0-dev libgstreamer1.0-0 libgstreamer-opencv1.0-0 \
gstreamer1.0-plugins-base libgstreamer-plugins-base1.0-dev libgstreamer-plugins-
↪base1.0-0 \
libgstreamer-plugins-bad1.0-dev gstreamer1.0-plugins-bad \
libgstreamer-plugins-good1.0-dev gstreamer1.0-plugins-good \
gstreamer1.0-libav gstreamer1.0-tools gstreamer1.0-x libopencv-dev \
gir1.2-gst-plugins-bad-1.0
```

To install Gstreamer:

```
sudo apt-get install -y gstreamer1.0-plugins-base gstreamer1.0-alsa \
gstreamer1.0-gl gstreamer1.0-gtk3 gstreamer1.0-qt5 gstreamer1.0-vaapi \
gstreamer1.0-pulseaudio libgstreamer-opencv1.0-0 gir1.2-gst-plugins-bad-1.0
```

Add `libgirepository-2.0-dev` and `gstreamer1.0-qt6` for Ubuntu 24.04 For further information, please refer to Gstreamer installation guide

To install pygobject:

```
sudo apt install python3-gi python3-gi-cairo gir1.2-gtk-3.0
```

For further details, please refer to pygobject installation guide

The following additional requirements are needed for GPU based hardware emulation:

1. Nvidia's Pascal/Turing/Ampere GPU architecture (such as Titan X Pascal, GTX 1080 Ti, RTX 2080 Ti, or RTX A4000)
2. GPU driver version 555
3. CUDA 12.5
4. CUDNN 9.6
5. (Recommended for TensorFlow) AVX instructions support on CPU

4.2.2. Installation

The executable is an archive that contains all required Hailo SW and an embedded installation script. The file will be named according to the following convention:

```
hailo_ai_sw_suite_<version>.run
```

For example:

```
hailo_ai_sw_suite_2026-01.run
```

After the executable is launched, the following will be present on the system:

- A new virtual environment named **"hailo_venv"** will be created and will contain the following:
    1. HailoRT Python package and its dependencies
    2. DFC Python package and its dependencies
    3. ModelZoo Python package and its dependencies
    4. All Python packages specified in the included requirements.txt file
    5. A target directory will be created containing all suite contents
- Hailo docs will be present in the directory from which the executable was launched
- Hailo PCIe driver installed

Usage:

First, make sure the file is executable by running `ls -l`. If not, turn it into an executable with:

```
sudo chmod 770 hailo_ai_sw_suite_<version>.run
```

The following command will extract the archive contents to `hailo_ai_sw_suite` directory and will install Hailo AI SW suite:

```
./hailo_ai_sw_suite_<version>.run
```

Example:

```
./hailo_ai_sw_suite_2026-01.run
```

For those who only want to extract and keep archive contents:

```
./hailo_ai_sw_suite_<version>.run --noexec
```

To run an internal validation hook after the installation is completed:

```
./hailo_ai_sw_suite_<version>.run -- validate-installation
```

To install TAPPAS, specify the `install-tappas` flag at the end of the command:

```
./hailo_ai_sw_suite_<version>.run -- install-tappas
```

After the installation is complete, activate the newly created virtual environment:

```
source hailo_ai_sw_suite/hailo_venv/bin/activate
```


Optional: Open WebUI Installation for Hailo Model Zoo GenAI in the Self Extracted Executable

To use Open WebUI with Hailo Model Zoo GenAI in a self-extracted executable installation, Open WebUI can be installed as a separate Docker container. Refer to :ref:Open WebUI (Optional) <open_webui_installation> for detailed installation instructions.

4.3. Manual installation

4.3.1. System requirements

Each package comes with its own requirements. See more details in Dataflow Compiler requirements, Model Zoo requirements, and HailoRT and TAPPAS requirements in their documentation, accordingly.

4.3.2. Downloading Hailo software packages

1. Download Dataflow Compiler, HailoRT, and TAPPAS from Developer Zone.
2. Clone Hailo Model Zoo Git repository https://github.com/hailo-ai/hailo_model_zoo.

4.3.3. Packages installation

1. Install Dataflow Compiler: See `Dataflow Compiler Installation` on the Dataflow Compiler user manual.
2. Install HailoRT in same virtual environment as Dataflow Compiler. See `Ubuntu Installation` on the HailoRT User Guide.
3. Install Model Zoo in same virtual environment as Dataflow Compiler. See Hailo Model Zoo Getting Started page.
4. For TAPPAS installation see the TAPPAS User Guide.

4.3.4. Sanity tests

1. Validate Dataflow Compiler installation

```
hailo parser onnx ~/workspace/hailo_virtualenv/lib/python3.X/site-
↪packages/hailo_tutorials/models/resnet_v1_18.onnx
hailo optimize resnet_v1_18.har --use-random-calib-set
hailo compiler resnet_v1_18_quantized.har
```

2. Validate HailoRT installation

```
hailortcli fw-control identify
```

3. To validate TAPPAS and Model Zoo, follow the installation validation in their installation guides, accordingly.

4.4. Open WebUI for Hailo Model Zoo GenAI (Optional)

Open WebUI is an optional user-friendly web interface that provides a modern web-based interface for chat with Hailo LLMs. It offers an easy-to-use browser-based alternative to command-line interactions, with features such as visual model management, conversation history, and multi-model support. Pull the Hailo-Ollama server LLM from the available models before starting to chat with AI.

Open WebUI can be installed and run as a separate Docker container that works alongside the Hailo AI SW Suite installation. It connects to the Hailo Model Zoo GenAI package, which provides the Hailo-Ollama server - an Ollama-compatible API for running GenAI models on Hailo devices. Open WebUI can be used with both Docker and Self Extracted Executable installation methods.

For more information about Hailo Model Zoo GenAI, visit the Hailo Model Zoo GenAI GitHub repository.

For more information about Open WebUI, visit the Open WebUI GitHub repository.

4.4.1. Usage with Hailo-Ollama Server

If you are using the Hailo Model Zoo GenAI package, follow these steps:

1. Start the Hailo-Ollama server (from the Docker Suite container in case of Docker installation):

```
hailo-ollama
```

2. Pull a model from the available models. For example:

```
curl --silent http://localhost:8000/api/pull \
   -H 'Content-Type: application/json' \
   -d '{ "model": "qwen2:1.5b", "stream" : true }'
```

Note: The model name `qwen2:1.5b` can be changed to any model from the available models list. To see the list of available models, use: `curl --silent http://localhost:8000/hailo/v1/list`

3. In a separate terminal (outside the Docker Suite container in case of Docker installation), install and run Open WebUI using Docker:

```
docker run -d --net=host -e OLLAMA_BASE_URL=http://127.0.0.1:8000 -v
↪open-webui:/app/backend/data --name open-webui --restart always ghcr.
↪io/open-webui/open-webui:main
```

4. After installation, you can access Open WebUI at http://localhost:8080.

Note: The Open WebUI Docker command assumes the Hailo-Ollama server is running on port 8000 (the default). Adjust the `OLLAMA_BASE_URL` environment variable if your Hailo-Ollama server is running on a different port or host.

5. Release Versions Compatibility

Hailo SW products are compatible with each other on specific versions. When upgrading a product, the others should be updated accordingly. The preferred option is to use Hailo SW Suites - both the **AI SW Suite** and the **Vision Processor Software Package** (**VPSP**), which align compatible versions.

5.1. Accelerators

Note: Refers to Hailo-10H, starting from 2025-07 versions. Older versions support Hailo-8 only.

Table 2. Release versions compatibility for Accelerators

AI SW Suite	Dataflow Compiler	HailoRT	Accelerator Integration Tool	Model Zoo	Model Zoo GenAI	TAPPAS
2026-01	v5.2.0	v5.2.0	v5.2.0	v5.2.0	v5.2.0	v5.2.0
2025-10	v5.1.0	v5.1.0		v5.1.0	v5.1.0	v5.1.0
2025-07	v5.0.0	v5.0.0		v5.0.0	v5.0.0	v5.0.0
2025-04	v3.31.0	v4.21.0		v2.15.0	N/A	
2025-01	v3.30.0	v4.20.0	v1.20.0	v2.14.0	N/A	v3.31.0
2024-10	v3.29.0	v4.19.0	v1.19.0	v2.13.0	N/A	v3.30.0
2024-07.1	v3.28.0	v4.18.0	v1.18.0	v2.12.0	N/A	v3.29.1
2024-07	v3.28.0	v4.18.0	v1.18.0	v2.12.0	N/A	v3.29.0
2024-04	v3.27.0	v4.17.0	v1.17.0	v2.11.0	N/A	v3.28.0
2024-01	v3.26.0	v4.16.0	v1.16.0	v2.10.0	N/A	v3.27.0
2023-10	v3.25.0	v4.15.0	v1.15.0	v2.9.0	N/A	v3.26.0
2023-07.1	v3.24.0	v4.14.0	v1.14.1	v2.8.0	N/A	v3.25.0
2023-07	v3.24.0	v4.14.0	v1.14.0	v2.8.0	N/A	v3.25.0
2023-04	v3.23.0	v4.13.0	v1.13.0	v2.7.0	N/A	v3.24.0
2023-01.1	v3.22.1	v4.12.1	v1.12.0	v2.6.1	N/A	v3.23.1
2023-01	v3.22.0	v4.12.0	v1.12.0	v2.6.0	N/A	v3.23.0
		v4.11.0		v2.5.0	N/A	v3.22.0
2022-10	v3.20.0	v4.10.0	v1.10.0	v2.4.0	N/A	v3.21.0
		v4.9.0			N/A	v3.20.0
	v3.19.0	v4.8.1		v2.3.0	N/A	
2022-07.1	v3.18.1	v4.8.1	v1.8.0	v2.2.0	N/A	v3.19.1
2022-07	v3.18.0	v4.8.0	v1.8.0	v2.2.0	N/A	v3.19.0
	v3.17.0	v4.7.0		v2.1.0	N/A	v3.18.0
2022-04	v3.16.0	v4.6.0	v1.6.0	v2.0.0	N/A	v3.17.0
	v3.15.0	v4.5.0	v1.5.0		N/A	v3.16.0
	v3.15.0	v4.4.0	v1.4.0		N/A	v3.15.0
	v3.14.0	v4.4.0	v1.4.0	v1.5	N/A	v3.15.0
2022-01	v3.14.0	v4.3.0		v1.4	N/A	v3.14.0

Note: The HailoRT column relates both to the HailoRT library and the driver.

5.2. Vision Processor Units

Note: Refers to Hailo-15H, Hailo-15M and Hailo-15L. Hailo-15L is supported from 2025-07 versions.

Table 3. Release versions compatibility for VPUs

AI SW Suite	Dataflow Compiler	HailoRT	Model Zoo	TAPPAS	VPSP Suite	VPSP sub-packages
2026-01	v5.2.0	v5.2.0	v5.2.0	v5.2.0	2026-02	1.10.0
2025-10	v5.1.0	v5.1.0	v5.1.0	v5.1.0	2025-10	1.9.0
2025-07	v5.0.0	v5.0.0	v5.0.0	v5.0.0	2025-07	1.8.0
2025-04	v3.31.0	v4.21.0	v2.15.0		2025-04	1.7.0
	v3.30.0	v4.20.1	v2.14.0		2025-01.1	1.6.1
2025-01	v3.30.0	v4.20.0	v2.14.0	v3.30.1	2025-01	1.6.0
2024-10	v3.29.0	v4.19.0	v2.13.0	v3.30.0	2024-10	1.5.0
	v3.28.0	v4.18.0	v2.12.0	v3.29.0	2024-07.1	1.4.1
2024-07	v3.28.0	v4.18.0	v2.12.0	v3.29.0	2024-07	1.4.0
		v4.17.1		v3.28.1	2024-04.1	1.3.1
2024-04	v3.27.0	v4.17.0	v2.11.0	v3.28.0	2024-04	1.3.0
	v3.26.0	v4.16.0	v2.10.0	v3.27.0	2024-01.2	1.2.2
	v3.26.0	v4.16.0	v2.10.0	v3.27.0	2024-01.1	1.2.1
2024-01	v3.26.0	v4.16.0	v2.10.0	v3.27.0	2024-01	1.2.0
2023-10	v3.25.0	v4.15.0	v2.9.0	v3.26.0	2023-10	1.1.0

6. Working with Docker Containers

6.1. Docker common commands

Docker is an open-source project that allows deployment of portable applications as containers, using OS-level virtualization. One of the Hailo AI SW Suite installation methods is using a Docker file, therefore we appended some common Docker commands.

> **Note:** In order to run all "docker" commands without "sudo", you'll have to add your user to group "docker"; instructions on how to do that can be looked up in "Suite installation" section.

1. Display the existing images

```
sudo docker images
```

2. Display the existing containers (both currently running and stopped)

```
sudo docker container ls -a
```

3. Start an existing container

```
sudo docker container start -i $container_name
```

4. Attach to a started container ("get inside it")

```
sudo docker container attach $container_name
```

5. Exit the container and stop it

```
exit
```

6. Stop a container

```
sudo docker container stop $container_name
```

7. Remove a container (it must be stopped first)

```
sudo docker container rm $container_name
```

8. Remove all stopped containers

```
sudo docker container prune
```

9. Remove an image (there should not be any containers or images based on it or its layers; image name is REPOSITORY:TAG when viewed with *docker images*)

```
sudo docker image rm $image_name
```

10. Remove all images (the images which currently have containers based on them will not be removed)

```
sudo docker image prune
```

11. Copy files and directories between host and container (can be done with both started and stopped containers). On Hailo AI SW Suite container, your workspace is on /local/workspace/

```
sudo docker cp $path_on_host $container_name:$path_inside_container
sudo docker cp $container_name:$path_inside_container $path_on_host
```

12. Attach additional bash terminal to a running container


```
sudo docker exec -it $container_name bash
```

6.2. Docker Containers and VSCode integration

Visual Studio Code is a common source-code editor made by Microsoft for Windows, Linux and macOS. It supports many programming languages and features a variety of plugins. For anyone who is willing to use VSCode (which is external to the Docker) to work with the contents of a Docker:

1. Start VSCode
2. Navigate to Extensions
3. Install the following extensions:
    1. Docker (by Microsoft)
    2. Remote - Containers (by Microsoft)
4. Now there should be an extra “Docker” icon on the left pane of VSCode
5. When you click “Docker” icon, you’ll be able to browse existing docker images and containers and have some additional actions available for you by right-clicking them, like starting, stopping, attaching, removing existing containers
6. Unfortunately there is no easy way to properly create (with all required arguments) new container from “Hailo Software Suite” image by right-clicking the image in VSCode; in order to work with “Hailo Software Suite” via VSCode, you’ll have to first create a container from terminal using the “hailo_ai_sw_suite_docker_run.sh” and then you’ll be able to “Attach Visual Studio Code” to the created container.

7. Known Issues

7.1. 2026-01 Suite

This section is relevant for 2026-01 Suite, and accompanying released packages:

- Dataflow Compiler v5.2.0
- HailoRT v5.2.0
- Hailo Model Zoo v5.2.0
- Hailo Model Zoo GenAI v5.2.0
- TAPPAS v5.2.0
- Integration Tool v5.2.0

7.1.1. Dataflow Compiler

- Weight sharing is not supported between different LoRA adapters within the same model. This means a model with 2 LoRA adapters cannot run on 4GB module without the *optimize_memory_on_device* flag. For more information, refer to the HailoRT user guide.
- Weight sharing is not supported in Qwen2-VL-Image-Encoder-2B model. This means that the model cannot run on 4GB module without the *optimize_memory_on_device* flag. For more information, refer to the HailoRT user guide.

7.1.2. HailoRT

- No known issues for now.

7.1.3. Hailo Model Zoo

- No known issues for now.

7.1.4. Hailo Model Zoo GenAI

- No known issues for now.

7.1.5. TAPPAS

- No known issues for now.

7.1.6. Integration Tool

- No known issues for now.

