

Fiche technique

Hailo-8L sur Raspberry Pi 5

Contents

1	Présentation	2
2	Matériel Requis	2
3	Installation Matérielle	2
4	Installation Logicielle	3
4.1	Installation de Raspberry Pi OS	3
4.2	Mise à jour système	3
5	Installation du driver	4
5.1	Installation des paquets nécessaires	4
5.2	Installation manuelle du driver PCIe	5
5.3	Installation des exemples Hailo	5
5.4	Vérification des versions installées	5
5.5	Exécution d'un exemple	6
6	Hailortcli et Dataflow Compiler	7
6.1	Utilisation de <code>hailortcli</code>	7
6.2	Compilation de modèle avec Hailo Dataflow Compiler	8
6.3	Création d'un environnement et compilation d'un modèle	9
6.3.1	Étape 1 : Convertir le modèle en format <code>.har</code>	9
6.3.2	Étape 2 : Compiler le modèle <code>.har</code> en <code>.hef</code>	10
6.4	Utilisation de Hailo Zoo	10
6.4.1	Guide de démarrage rapide	10

Présentation

Objectif : Intégrer l'accélérateur Hailo-8L au Raspberry Pi 5 ainsi que les divers packages pour le traitement de l'IA dans un système embarqué.

Cette fiche technique permettra d'avoir une base d'installation pour divers projet et utilisations de la carte Hailo-8L pour une raspberry

Composants concernés :

- Raspberry Pi 5
- Hailo-8L (via PCIe ou USB)
- Alimentation adaptée (USB-C 5V/5A)

Matériel Requis

Composant	Référence	Remarques
Raspberry Pi 5	-	Dernière version avec port PCIe
Hailo-8L	M.2 / USB / HAT	Attention si modèle change
Carte microSD	16GB	Avec Raspberry Pi OS (64-bit)

Installation Matérielle

Connexion du Hailo-8L

- **Via PCIe :** Connecter à l'adaptateur PCIe du Pi 5
- **Via USB :** Brancher directement sur port USB 3.0

Note : Vérifiez que le Raspberry Pi 5 est éteint lors des diverses installations et branchements.

Alimentation

- Utiliser une alimentation 5V/5A
- Vérifier la stabilité sous charge avec le Hailo

Installation Logicielle

Installation de Raspberry Pi OS

Le Raspberry Pi OS est adapté au matériel du Raspberry Pi, ce qui en fait un bon choix pour développer et déployer des applications embarquées, notamment celles reposant sur l'IA avec le Hailo-8L, et pour vérifier leur bon fonctionnement avant une exportation vers un système minimaliste comme Buildroot (dans le cadre de ce projet).

Avec Raspberry Pi Imager

1. Télécharger Raspberry Pi Imager
2. Insérer une carte microSD (min 16 Go)
3. Choisir l'OS : Raspberry Pi OS 64-bit (Lite ou Desktop) tout en choisissant le type de modèle de votre raspberry
4. Cibler la carte SD, puis flasher.
5. Insérer la carte dans le Raspberry Pi 5 et démarrer.

Conseil : Activer SSH et le Wi-Fi dans les options avancées de Raspberry Pi Imager si nécessaire.

Mise à jour système

Une fois le Raspberry Pi allumé, connectez-vous à internet et effectuez les mises à jour nécessaires :

```
sudo apt update && sudo apt full-upgrade -y
```

Installation du driver

Rendez-vous sur le site officiel de Hailo Developer Zone, où vous devrez créer un compte afin d'accéder aux ressources nécessaires. Téléchargez le driver adapté à votre version matérielle et logicielle (PCIe dans notre cas).

Software Package	Software Sub-Package	Architecture	OS	Python Version
<input checked="" type="radio"/> AI Software Suite	<input checked="" type="radio"/> AI Software Suite <input type="radio"/> Dataflow Compiler <input type="radio"/> HailoRT <input type="radio"/> Model Zoo <input type="radio"/> TAPPAS	<input type="radio"/> ARMv8 <input type="radio"/> ARMv8L <input type="radio"/> ARMv8HF <input checked="" type="radio"/> x86	<input checked="" type="radio"/> Linux <input type="radio"/> Windows	<input checked="" type="radio"/> 3.8 <input type="radio"/> 3.9 <input type="radio"/> 3.10 <input type="radio"/> 3.11

Package Name	Version	Documentation	Date Modified
HailoRT - PCIe driver Ubuntu package (deb)	4.21.0	Installation guide	April 2, 2025

Figure 1: Page développeur de Hailo avec lien vers le fichier `.deb` du driver PCIe.

Installation des paquets nécessaires

Commencez par installer les paquets essentiels, y compris ceux requis pour le fonctionnement de base du Hailo-8L :

```
sudo apt install hailo-all
```

Cette commande n'installe pas tous les composants, mais ceux nécessaires pour les premières manipulations.

Installation manuelle du driver PCIe

Installez le fichier `.deb` du driver PCIe, adapté à votre système :

```
sudo dpkg -i chemin/vers/hailort-pcie-driver_4.20.0_all.deb
```

En cas d'erreur liée au paquet hailofw : Certains fichiers étant partagés entre le firmware et le driver, il est parfois nécessaire de suivre cette procédure pour éviter les conflits :

1. Supprimer le firmware Hailo existant :

```
sudo apt remove hailofw
```

2. Réinstaller le driver PCIe :

```
sudo dpkg -i chemin/vers/hailort-pcie-driver_4.20.0_all.deb
```

3. Réinstaller le firmware si besoin :

```
sudo apt install hailofw
```

Cela va permettre de garder le firmware de la version de votre driver.

Installation des exemples Hailo

Clonons maintenant les exemples fournis par Hailo pour Raspberry Pi 5 et procédons à leur installation afin de vérifier que tout marche bien:

```
git clone https://github.com/hailo-ai/hailo-rpi5-examples.git
cd hailo-rpi5-examples
./install.sh
```

En cas d'erreur de droits, relancer la commande avec `sudo ./install.sh`

Vérification des versions installées

Assurez-vous que toutes les versions des paquets Hailo sont compatibles :

```
dpkg -l | grep hailo
```

Si des conflits de version apparaissent (paquets en double ou incompatibles), désinstallez-les manuellement pour éviter les erreurs à l'aide des commandes suivantes :

- `sudo dpkg -r <nom_du_package>` — si vous avez installé le paquet avec `dpkg`
- `sudo apt remove <nom_du_package>` — si vous l'avez installé avec `apt`

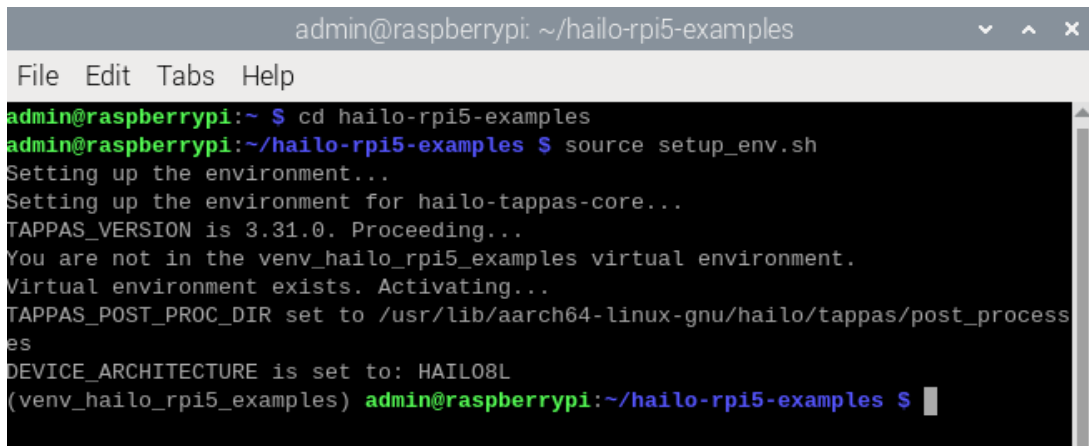
Les versions évoluant rapidement, il peut être nécessaire de refaire cette commande après chaque étape.

Exécution d'un exemple

Avant de lancer les exemples, il est important d'exécuter le script de configuration de l'environnement :

```
source setup_env.sh
```

Si tout se passe correctement, vous devriez obtenir ce type de message :



```
admin@raspberrypi: ~/hailo-rpi5-examples
File Edit Tabs Help
admin@raspberrypi:~ $ cd hailo-rpi5-examples
admin@raspberrypi:~/hailo-rpi5-examples $ source setup_env.sh
Setting up the environment...
Setting up the environment for hailo-tappas-core...
TAPPAS_VERSION is 3.31.0. Proceeding...
You are not in the venv_hailo_rpi5_examples virtual environment.
Virtual environment exists. Activating...
TAPPAS_POST_PROC_DIR set to /usr/lib/aarch64-linux-gnu/hailo/tappas/post_process
es
DEVICE_ARCHITECTURE is set to: HAILO8L
(venv_hailo_rpi5_examples) admin@raspberrypi:~/hailo-rpi5-examples $
```

S'il y a une erreur liée à un module Tappas manquant ou tout autre problème, vérifiez attentivement les étapes précédentes.

Puis exécuter un exemple de pipeline de détection (à modifier selon vos besoins) :

```
python basic_pipelines/detection.py --input rpi
```

Si vous avez branché une caméra Raspberry Pi, celle-ci sera automatiquement détectée et utilisée. Cependant, si vous n'en avez pas ou que vous préférez ne pas tester avec un flux vidéo, utilisez simplement :

```
python basic_pipelines/detection.py
```

N'hésitez pas à explorer les différents exemples fournis sur le dépôt GitHub pour mieux comprendre le fonctionnement du système.

Hailortcli et Dataflow Compiler

Cette deuxième partie présente plus précisément l'utilisation de la carte Hailo. La partie précédente servait principalement à tester et valider l'installation matérielle et logicielle. Nous allons maintenant explorer deux outils essentiels pour déployer un modèle IA sur la Hailo-8L :

- **hailortcli** : outil en ligne de commande pour interagir directement avec l'accélérateur.
- **Hailo Dataflow Compiler** : pour compiler un modèle neuronal dans un format optimisé pour la puce Hailo (.hef).

Utilisation de hailortcli

Le programme **hailortcli** permet de diagnostiquer l'état de la carte Hailo ainsi que d'exécuter un modèle déjà compilé.

Scanner la carte Hailo

```
hailortcli scan
```

La commande ci-dessus permet de s'assurer que le périphérique Hailo est bien détecté.

Si vous souhaitez plus d'informations, je vous invite à entrer la commande suivante :

```
hailortcli fw-control identify
```

Vous aurez alors ce genre de résultat :

```
admin@raspberrypi:~ $ hailortcli scan
Hailo Devices:
[-] Device: 0001:01:00.0
admin@raspberrypi:~ $ hailortcli fw-control identify
Executing on device: 0001:01:00.0
Identifying board
Control Protocol Version: 2
Firmware Version: 4.20.0 (release,app,extended context switch buffer)
Logger Version: 0
Board Name: Hailo-8
Device Architecture: HAILO8L
Serial Number: HLDDLBB242600139
Part Number: HM21LB1C2LAE
Product Name: HAILO-8L AI ACC M.2 B+M KEY MODULE EXT TMP
```

Cela vous retournera la version et type de votre hailo, cela nous sera utile lors de la compilation de votre modèle IA en .hef

Exécution d'un modèle compilé HEF (Hailo Executable Format)

Si vous possédez déjà votre modèle où que vous en avez pris un en ligne (par exemple dans la section modèle du site hailo), alors vous pouvez directement tester celui-ci avec :

```
hailortcli run chemin/vers/modele.hef
```

Avec la commande suivante :

```
hailortcli --help
```

*vous pourrez découvrir les différentes options offertes par **hailortcli** en fonction de vos besoins.*

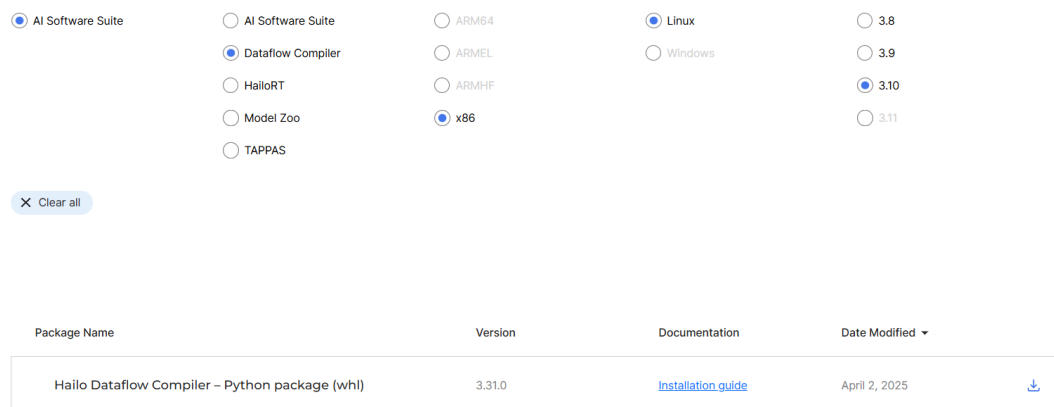
Compilation de modèle avec Hailo Dataflow Compiler

Le Hailo Dataflow Compiler permet de convertir un modèle IA (TensorFlow, ONNX...) en un fichier HEF utilisable sur la puce Hailo.

Important : cette étape est à réaliser sur une machine Linux, via WSL si vous êtes sous Windows, car au moment de la création de cette fiche, il y a pas de SDK disponible pour ARM64.

Installation sur WSL

Rendez-vous sur la Developer Zone de Hailo pour télécharger le whl du Dataflow Compiler.



Package Name	Version	Documentation	Date Modified
Hailo Dataflow Compiler – Python package (whl)	3.31.0	Installation guide	April 2, 2025

Création d'un environnement et compilation d'un modèle

Commencez par installer Python (si ce n'est pas déjà fait) :

```
sudo apt install python3.10-venv
```

Créez ensuite un environnement virtuel Python :

```
python3 -m venv <nom_de_votre_env>
```

Activez-le avec la commande suivante :

```
source <nom_de_votre_env>/bin/activate
```

Installez ensuite le fichier `.whl` du Dataflow Compiler :

```
pip install ~/hailo_dataflow_compiler-3.31.0-py3-none-linux_x86_64.whl
```

Assurez-vous que le chemin vers le fichier `.whl` est correct selon votre dossier de téléchargement.

Pour vérifier que l'installation s'est bien passée, tapez :

```
hailo -h
```

Création d'un environnement et compilation d'un modèle

Attention, pour cette partie, nous n'avons pas réussi à compiler nos modèles en `.hef` à ce jour, mais je vais vous guider à travers les étapes du processus qui semblent correctes.

Tout d'abord, il est important de bien préparer votre modèle IA, car Hailo ne prend pas encore en charge tous les types de modèles (par exemple, les modèles quantifiés, certains types de couches Conv2D avec des paramètres spécifiques, etc.).

Étape 1 : Convertir le modèle en format `.har`

Pour convertir votre modèle (par exemple, un modèle en `.h5`, `.pt`, etc.), il faut d'abord le transformer en format ONNX. Une fois cette conversion effectuée, vous pouvez utiliser la commande suivante dans votre terminal wsl :

```
hailo parser onnx votre_modele.onnx
```

Cette commande générera un modèle au format `.har`, ce qui est une étape nécessaire avant de pouvoir le compiler en `.hef`.

Étape 2 : Compiler le modèle .har en .hef

Une fois que vous avez le fichier .har, vous devrez le compiler avec :

```
hailo compiler votre_modele.har --output-dir output --hw-arch
```

Assurez-vous de spécifier l'architecture matérielle correcte de votre Hailo (dans la commande ci-dessus, il s'agit de "arch") pour que le processus de compilation fonctionne correctement.

Malheureusement, comme mentionné plus haut, nous nous sommes retrouvés bloqués à cette étape et avons dû faire nos tests avec des modèles fournis directement par le site de Hailo car nos modèles n'étaient pas valide pour la conversion.

Utilisation de Hailo Zoo

La suite de ce processus aurait été d'utiliser **Hailo Zoo** car avec il est possible de :

- Mesurer la précision en virgule flottante de chaque modèle.
- Évaluer la précision quantifiée à l'aide de l'émulateur Hailo.
- Mesurer la précision sur l'appareil Hailo.
- Générer un fichier binaire au format HEF pour accélérer le développement et produire des applications de haute qualité avec le Hailo-8.

Le Hailo Model Zoo fournit également des instructions pour le re-entraînement des modèles sur des jeux de données personnalisés, ainsi que des modèles déjà entraînés pour des cas d'utilisation spécifiques à l'aide de jeux de données internes.

Guide de démarrage rapide

1. Activer votre environnement virtuel, celui de votre wsl car nous aurons besoin de DataflowCompiler.
2. Cloner le dépôt du Model Zoo :

```
git clone https://github.com/hailo-ai/hailo_model_zoo.git
```

3. Exécuter le script d'installation :

```
cd hailo_model_zoo  
pip install -e .
```

4. Lancer le Model Zoo pour afficher les informations de votre modèle :

```
hailomz info <votre_modèle>
```