# Installation von Pytorch mit CUDA-Unterstützung auf NVIDIA Jetson

## 1 Installationsanleitung

Diese Anleitung beschreibt die korrekte Installation von PyTorch mit CUDA-Unterstützung auf NVIDIA Jetson-Geräten (z. B. Jetson Orin) unter JetPack 6.0 (entspricht Jetson Linux 36.x).

**Hinweis:** Es ist empfohlen, zunächst YOLO bzw. Ultralytics zu installieren, da diese bereits PyTorch und TorchVision mitliefern. Allerdings sind die automatisch installierten Versionen häufig nicht mit CUDA kompatibel. Im Rahmen dieser Anleitung werden die durch Ultralytics installierten PyTorch- und TorchVision-Versionen daher zunächst deinstalliert und anschließend durch CUDA-kompatible Versionen ersetzt, um eine reibungslose Nutzung der GPU-Beschleunigung sicherzustellen.

### 1.1 CUDA und GPU-Überprüfung

#### 1.1.1 CUDA-Version prüfen

```
$ nvcc --version
```

Falls der Befehl nicht gefunden wird, ist CUDA entweder nicht installiert oder nicht im `PATH` enthalten.

#### 1.1.2 NVIDIA GPU erkennen

```
$ nvidia-smi
```

**Hinweis:** Jetson-Geräte verwenden integrierte GPUs (`nvgpu`) und nicht den Desktop-Treiber. Daher zeigt `nvidia-smi` „Driver Version: N/A“.

### 1.2 CUDA-Verfügbarkeit mit PyTorch testen

```
$ python3
>>> import torch
>>> print(torch.cuda.is_available())
>>> print(torch.cuda.get_device_name(0))
```

Falls `torch.cuda.is_available()` **False** zurückgibt, ist wahrscheinlich die CPU-only Version von PyTorch installiert.

## 1.3 PyTorch (GPU) Installation

### 1.3.1 1. Alte Version entfernen

```
$ pip uninstall torch torchvision
```

### 1.3.2 2. PyTorch-Wheel für JetPack 6.0 herunterladen und installieren

```
$ wget https://developer.download.nvidia.com/compute/redist/jp
   ↪ /v60/pytorch/torch-2.4.0a0+07cecf4168.nv24.05.14710581-
   ↪ cp310-cp310-linux_aarch64.whl
$ pip install torch-2.4.0a0+07cecf4168.nv24.05.14710581-cp310-
   ↪ cp310-linux_aarch64.whl
```

### 1.3.3 3. Installation von `torchvision`

Kompatible Version auf GitHub prüfen [?].

```
$ git clone https://github.com/pytorch/vision.git
$ cd ~/vision/torchvision
$ git checkout v0.17.0
$ python3 setup.py install --user
```

## 1.4 Installation in einer virtuellen Umgebung

Die Installation in einer virtuellen Umgebung unterscheidet sich von der Installation direkt ins System. Falls Sie in einer virtuellen Umgebung arbeiten, folgen Sie zur Installation den folgenden Anweisungen.

### 1.4.1 1. Virtuelle Umgebung erstellen und aktivieren

```
$ cd ~
$ python3 -m venv myvenv
$ source myvenv/bin/activate
```

### 1.4.2 2. Abhängigkeiten installieren

```
$ pip install --upgrade pip setuptools wheel
$ pip install numpy ninja pyyaml mkl mkl-include typing-
   ↪ extensions
$ pip install pillow six requests dataclasses tqdm
```

### 1.4.3 3. `torchvision` kompilieren und installieren

```
$ git clone https://github.com/pytorch/vision.git ~/vision/
   ↪ torchvision
$ cd ~/vision/torchvision
$ git checkout v0.17.0
$ python setup.py install
```

Falls folgender Fehler auftritt:

```
TypeError: Command.__init__() got an unexpected keyword argument
'no_python_abi_suffix'
```

Dann:

```
$ pip install setuptools==59.5.0
$ python setup.py install
```

## 1.5 Installation verifizieren

```
$ python3
>>> import torchvision
>>> print(torchvision.__version__)
```

## 1.6 CUDA-Verfügbarkeit in Python testen

```
$ python3
>>> import torch
>>> print("CUDA␣available:", torch.cuda.is_available())
>>> print("Device␣name:", torch.cuda.get_device_name(0) if
   ↪ torch.cuda.is_available() else "No␣GPU")
```