GPU Setup & Installation Guide for PyTorch + NeuralForecast

1. Verify Hardware and OS

Make sure your system has:

- **NVIDIA GPU** (e.g., RTX 4070)
- **Windows 10/11 (64-bit)**
- **Anaconda / Miniconda** installed

---

2. Install or Update NVIDIA GPU Driver

1. Go to NVIDIA Driver Downloads.
2. Select your GPU model and download the latest **Game Ready** or **Studio Driver**.
3. During installation, choose **Custom (Advanced)** → **Clean Installation**.

☑ To verify installation:

```
nvidia-smi
```

Expected output (example):

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 560.94       Driver Version: 560.94       CUDA Version: 12.6     |
| GPU Name: NVIDIA GeForce RTX 4070                                          |
+-----------------------------------------------------------------------------+
```

---

3. (Optional) Install CUDA Toolkit (for developers)

> ⚠ If you only plan to use PyTorch, you **do not need to manually install the CUDA Toolkit**.
> PyTorch comes with its own CUDA runtime.

If you want the toolkit for other GPU tools (like compiling CUDA code):

- Download from: NVIDIA CUDA Toolkit 12.6
- Follow on-screen installer prompts.
- Add CUDA to PATH when prompted.

☑ Verify toolkit:

```
nvcc --version
```

4. Create a New Conda Environment

Create a clean environment to avoid dependency conflicts:

```
conda create -n neuralforecast_env python=3.11 -y
conda activate neuralforecast_env
```

5. Install PyTorch with GPU Support

Use the official NVIDIA + PyTorch channels:

```
conda install pytorch torchvision torchaudio pytorch-cuda=12.1 -c pytorch -c
nvidia
```

CUDA 12.1 build works perfectly with any driver ≥12.1 (e.g., 12.6).

Verify GPU access:

```
import torch
print("Torch version:", torch.__version__)
print("CUDA available:", torch.cuda.is_available())
print("CUDA version:", torch.version.cuda)
if torch.cuda.is_available():
    print("GPU name:", torch.cuda.get_device_name(0))
```

Expected output:

```
Torch version: 2.x.x
CUDA available: True
CUDA version: 12.1
GPU name: NVIDIA GeForce RTX 4070
```

6. Install NeuralForecast and Other Dependencies

Use pip for these to avoid MKL conflicts:

```
pip install neuralforecast numpy pandas matplotlib
```

🧪 7. Test GPU Acceleration

Run a small test to confirm tensors are created on GPU:

```
import torch
x = torch.rand(10000, 10000, device="cuda")
print(x.sum())
```

You can also monitor GPU activity with:

```
nvidia-smi
```

You should see your Python process using GPU memory.

⚠️ 8. (If You See “libiomp5md.dll” Error)

This happens when Intel MKL and PyTorch both load OpenMP.

Temporary fix:

```
set KMP_DUPLICATE_LIB_OK=TRUE
```

Permanent fix:

- Avoid installing MKL-based packages (`nomkl` package is unreliable on Windows).
- Use `pip install` instead of `conda install` for packages like `numpy`, `scipy`, etc.

☑️ 9. Optional: Export Environment for Future Use

Once everything works, save it:

```
conda env export > neuralforecast_env.yml
```

Then you can recreate it anytime with:

```
conda env create -f neuralforecast_env.yml
```

🏁 Summary of Installation Flow

Step	Component	Tool / Source	Notes
1	Verify GPU	`nvidia-smi`	Ensures GPU recognized
2	Install Driver	NVIDIA website	Use latest stable
3	(Optional) CUDA Toolkit	NVIDIA developer site	Only if you need nvcc
4	Conda Environment	`conda create`	Clean setup
5	PyTorch (GPU)	`conda install -c pytorch -c nvidia`	GPU-enabled build
6	NeuralForecast	`pip install`	Avoids MKL conflicts
7	GPU Test	Python + `nvidia-smi`	Confirms GPU use
8	Handle Errors	`set KMP_DUPLICATE_LIB_OK=TRUE`	Fixes OpenMP conflict
9	Export Env	`conda env export`	For reproducibility