Instructions about running a simple LSTM model in PyTorch

Installation recommendations:

There is **no separate PyTorch GPU library** like pytorch-gpu. You choose the CUDA version explicitly during installation of PyTorch. However, you must manually install compatible CUDA/cuDNN versions with TensorFlow.

Windows: It is recommended, but not required, that your Windows system has an NVIDIA GPU in order to harness the full power of PyTorch’s CUDA support.

Mac: PyTorch can be installed and used on macOS.

Linux: It is recommended, but not required, that your Linux system has an NVIDIA or AMD GPU in order to harness the full power of PyTorch’s [CUDA](https://developer.nvidia.com/cuda-zone) [support](https://pytorch.org/tutorials/beginner/blitz/tensor_tutorial.html?highlight=cuda&__hstc=76629258.724dacd2270c1ae797f3a62ecd655d50.1746547368336.1746547368336.1746547368336.1&__hssc=76629258.10.1746547368336&__hsfp=2230748894" \l "cuda-tensors) or [ROCm](https://rocm.docs.amd.com/) support.

**CUDA** → for **NVIDIA** GPUs only

**ROCm** → for **AMD** GPUs only

CUDA:

* **Sends data** to the GPU
* **Runs computations** on the GPU
* **Brings the result back** to the CPU

Installation on my Windows Laptop:

Go to this link: <https://pytorch.org/get-started/locally/>

A screenshot of a computer

AI-generated content may be incorrect.

As you can see, PyTorch is compatible with Python >=3.9.

First step: conda create -n pytorch python=3.10

Then, conda activate pytorch

Then, copy the command in the above figure. It also installs the appropriate CUDA platform.

By doing that, PyTorch starts to be installed:

A screen shot of a computer screen

AI-generated content may be incorrect.

After successful installation, to check whether everything is ok:

import torch

print('torch version: ', torch.\_\_version\_\_)

print('CUDA available: ', torch.cuda.is\_available())

print('GPU device: ', torch.cuda.get\_device\_name())

it gives something like this on my ASUS laptop:  
  
torch version: 2.7.1+cu118

CUDA available: True

GPU device: NVIDIA GeForce GTX 960M

it gives something like this on my PC ubuntu laptop:  
  
torch version: 2.3.1+cu121

CUDA available: True

GPU device: NVIDIA GeForce RTX 3050

For implementing a simple LSTM

import torch

import torch.nn as nn

torch.nn is PyTorch's neural network module. It provides:

* Neural network layers (e.g., nn.Linear, nn.LSTM, nn.Conv2d)
* Loss functions (e.g., nn.MSELoss, nn.CrossEntropyLoss)
* Utilities to build and train neural networks

LSTM accepts and input of (sample, timestep, feature) for example (246, 10, 3) which 10 indicates the timseries of length 10, 3 are the number of features

Then, you have to convert your input to tensor (you cannot pass directly a numpy array ta PyTorch model). tensor is also an array but another format only. Most importantly, tensor has GPU support which numpy does not.

torch.nn.LSTM(input\_size = numoffeatures, hidden\_size = numoffeatures in LSTM layer, num\_layers=1 (if you put it more it will be converted to a stack of lstms with hidden size features), bias=True, batch\_first=True to accept input of (sample, timestep, feature), dropout = 0.0 introduces a Dropout layer on the outputs of each LSTM layer except the last layer)

Basics or vanilla RNN:

Has weight, bias the same as ANNs but they have also feedback loops

The big difference in pytorch and keras is that pytorch return the sequence instead of the final only (in keras you can do both)