# Installation User Manual

## Python Environment Setup

For the project, Python >=3.7 is preferred as most of the library used in this project are supported only on this version of Python. This version of python can be easily downloaded from the official Python website.

If you have multiple versions of Python installed in the system, you can use virtual environment package of Python to create an isolated environment for the project.

## Trainer Setup

The Wav2Vec model trainer is built using Jupyter Lab. The code can be accessed through Jupyter lab. The installation of Jupyter Lab can be done through the following code:

**pip install jupyter-lab**

The lab can be opened through the terminal/ command line using the code:

**jupyter lab**

After Jupyter Labs opens, you can use the file explorer on the left to open the notebook named “openslr-54-nepali.ipynb”.

For the dataset, you can download the complete audio files from the site:

[**https://openslr.org/54/**](https://openslr.org/54/)

Then, you can unzip all the files and store the files in the ./content folder. Please edit the file location in the code as required.

In the notebook, you can edit the dataset to be loaded for training. Note that the training data is loaded through csv or tsv files and the column format used is filepath, userId and transcript of the audio. If your training data file contains format different from the specified template, please change the code as required to meet your needs.

The training parameters can be changed as required.  
Note that due to the large number of audio files, a minimum of 32 GB free RAM should be available to load the audio data into the RAM. Also, be careful when changing the batch size and eval\_accumulation\_steps as higher the number in these parameters, higher the GPU memory is required in this case.

## API usage

There are two ways of using the model trained from the trainer. One is using the Python API directly. This requires each and every machine where the code is deployed to have full Python and requirements setup. While the other requires a network with the deployed server.

For both the cases, the deployed machines should have Python 3.7 installed and running. Also, the requirements are listed in the file “requirements.txt”. You can directly use this file to install all the dependencies using the command in the terminal or command line:

**pip install -r requirements.txt**

Or you can manually choose to install all the libraries.

Additionally, you need to install PyTorch with CUDA enabled for the project. This can be done through a sample code:

**pip install torch==1.9.0+cu111 torchvision==0.10.0+cu111 torchaudio===0.9.0 -f** [**https://download.pytorch.org/whl/torch\_stable.html**](https://download.pytorch.org/whl/torch_stable.html)

If you have CUDA version 11.1, this code works. If your version of CUDA is different, please refer to [**https://pytorch.org/**](https://pytorch.org/) for installing the required version of PyTorch.

### Python API

You can run the Python API by directly adding instance creating and filename input in the “nepali\_asr.py” file. Or you can create another file and import the class NepaliASR from the “nepali\_asr.py” file.

### Client-server API

For deploying the API in the server, please copy all the codes to the deploying machine i.e., the server. You can run the API on the server by using Python. The code is:

**python api.py**

This runs the code in the server which can be accessed through the client machine through exposure of the server IP and routing using nginx, apache and other such software. The port for running the API is defined in the .env file which you can change as required.