

# Configurazione ed esecuzione per OS windows

---

## Requisiti

Questo progetto è stato eseguito su un ambiente target avente le seguenti specifiche.

Hardware:

- processore i9 di 13 generazione
- 16 giga ram
- scheda grafica RTX 4050 con memora dedicata di 6 giga

Software:

- sistema operativo Windows 11
- driver cuda installati
- python 3.10
- nodejs 18.20.3

## Note sui requisiti

- Ambienti con specifiche hardware minori potrebbero comunque eseguire correttamente il progetto a scapito delle prestazioni. Non è tuttavia garantito.
- Ambienti con OS diversi o driver cuda non installati potrebbero comunque eseguire correttamente il progetto a scapito delle prestazioni. Non è tuttavia garantito.
- Versioni node diverse potrebbero comunque eseguire correttamente il progetto a scapito delle prestazioni. Non è tuttavia garantito.
- Versioni python superiori, ma non minori, potrebbero comunque eseguire correttamente il progetto a scapito delle prestazioni. Non è tuttavia garantito.

## Esecuzione

Da powershell:

1. Installare le dipendenze node attraverso il comando "npm install"
2. Sulla root di progetto generare un ambiente virtuale con il comando "python -m venv venv"
3. Attivare l'ambiente virtuale con il comando ".\venv\Scripts\Activate.ps1"
4. Installare le dipendenze python con il comando "python -m pip install -r requirements.txt"
5. Eseguire il progetto con il comando "python .\src\main.py datasetNameToTest"

## Note sull'esecuzione

- La stringa "datasetNameToTest" deve essere sostituita con il nome di uno dei dataset supportati (mnist,cifar10 e cifar100).
- Se il nome del dataset non viene specificato, di default, verranno eseguite le sperimentazioni con il dataset mnist.
- I passi descritti sono validi solo per la prima esecuzione. Dalla seconda esecuzione in poi è sufficiente eseguire solo il passo 5.

## Possibili problemi

- Se durante l'installazione delle dipendenze python tramite requiremts.txt si riceve un errore simile a `<<Could not find a version that satisfies the requirement torch==2.3.0+cu121>>` è sufficiente eseguire lo script "manuallyDependencyInstaller.cmd" da un terminale con ambiente virtuale attivo o installare manualmente le dipendenze lanciando i seguenti comandi sempre da un terminale con ambiente virtuale attivo:
  - `python -m pip install torch==2.3.0 torchvision==0.18.0 torchaudio==2.3.0 --index-url https://download.pytorch.org/whl/cu121`
  - `python -m pip install aiohttp==3.9.5 aiosignal==1.3.1 asttokens==2.4.1 async-timeout==4.0.3 attrs==23.2.0 bitarray==2.9.2 blinker==1.9.0 certifi==2024.6.2 cffi==1.16.0 charset-normalizer==3.3.2 ckzg==1.0.2 click==8.1.7 colorama==0.4.6 comm==0.2.2 contourpy==1.2.1 cryptography==42.0.7 cycler==0.12.1 cytoolz==0.12.3 datasets==2.19.1 debugpy==1.8.1 decorator==5.1.1 dill==0.3.8`
  - `python -m pip install eth-account==0.11.2 eth-hash==0.7.0 eth-keyfile==0.8.1 eth-keys==0.5.1 eth-rlp==1.0.1 eth-typing==4.4.0 eth-utils==4.1.1 eth_abi==5.1.0 exceptiongroup==1.2.1 executing==2.0.1 filelock==3.13.1 Flask==3.1.0 flwr==1.8.0 flwr-datasets==0.1.0 fonttools==4.53.0 frozenlist==1.4.1 fsspec==2024.2.0 grpcio==1.64.0 hexbytes==0.3.1 huggingface-hub==0.23.2 idna==3.7 intel-openmp==2021.4.0 ipykernel==6.29.4 ipython==8.25.0 iterators==0.0.2 itsdangerous==2.2.0 jedi==0.19.1 Jinja2==3.1.3 joblib==1.4.2 jsonschema==4.22.0 jsonschema-specifications==2023.12.1 jupyter_client==8.6.2 jupyter_core==5.7.2 kiwisolver==1.4.5 lru-dict==1.2.0 markdown-it-py==3.0.0 MarkupSafe==2.1.5`
  - `python -m pip install matplotlib==3.9.0 matplotlib-inline==0.1.7 mdurl==0.1.2 mkl==2021.4.0 mpmath==1.3.0 msgpack==1.0.8 multidict==6.0.5 multiprocessing==0.70.16 nest-asyncio==1.6.0 networkx==3.2.1 numpy==1.26.4 packaging==24.0 pandas==2.2.2 parsimonious==0.10.0 parso==0.8.4 pillow==10.3.0 platformdirs==4.2.2 prompt_toolkit==3.0.46 protobuf==4.25.3 psutil==5.9.8 pure-eval==0.2.2 pyarrow==16.1.0 pyarrow-hotfix==0.6 pycparser==2.22 pycryptodome==3.20.0 pydantic==1.10.15 Pygments==2.18.0 pyparsing==3.1.2 python-dateutil==2.9.0.post0 pytz==2024.1 pyunormalize==15.1.0 pywin32==306`
  - `python -m pip install PyYAML==6.0.1 pyzmq==26.0.3 ray==2.6.3 referencing==0.35.1 regex==2024.5.15 requests==2.32.3 rich==13.7.1 rlp==4.0.1 rpds-py==0.18.1 scikit-learn==1.6.0 scipy==1.14.1 seaborn==0.13.2 shellingham==1.5.4 six==1.16.0 stack-data==0.6.3 sympy==1.12 tbb==2021.11.0 threadpoolctl==3.5.0 tomli==2.0.1 toolz==0.12.1 tornado==6.4.1 tqdm==4.66.4 traitlets==5.14.3 typer==0.9.4 typing_extensions==4.9.0 tzdata==2024.1 urllib3==2.2.1 wcwidth==0.2.13 web3==6.20.0 websockets==12.0 Werkzeug==3.1.3 xxhash==3.4.1 yarl==1.9.4`
- Se durante l'esecuzione si riceve un errore simile a `<<ModuleNotFoundError: No module named "src">>` e le dipendenze dette al passo 4 sono state installate correttamente, allora, aggiungere la root del progetto alla variabile di ambiente PATH e rieseguire da terminale.

## Configurazione ed esecuzione per OS unix like

Il sistema finale non è stato testato su OS unix like, tuttavia, al di fuori del passo 3 il cui comando descritto dovrebbe essere sostituito con "source env/bin/activate", i restanti requisiti e i passi esposti per Windows dovrebbero rimanere immutati.

