**Repository structure**

hacker-news-gpt/

│

├─ mainrun/ # main Python package

│ ├─ \_\_init\_\_.py

│ ├─ config/ # configuration files

│ │ ├─ \_\_init\_\_.py

│ │ ├─ hyperparams.yaml

│ │ └─ training.yaml

│ ├─ data/ # dataset handling

│ │ ├─ \_\_init\_\_.py

│ │ └─ dataset.py

│ ├─ model/ # models

│ │ ├─ \_\_init\_\_.py

│ │ ├─ gpt.py # your GPT model

│ │ └─ base.py # base classes for future models

│ ├─ utils/ # utility functions

│ │ ├─ \_\_init\_\_.py

│ │ └─ tokenizer.py

│ ├─ train.py # main training loop

│ ├─ train\_helper.py # helper functions (e.g., get\_batch, evaluate)

│ └─ logs.py # logging configuration

│

├─ notebooks/ # optional: Jupyter notebooks for analysis

├─ scripts/ # CLI or data preprocessing scripts

├─ requirements.txt

├─ README.md

└─ setup.py # optional, if you want pip installable package

**2️⃣ Configuration using OmegaConf**

* Keep all hyperparameters and training settings in **YAML** files under config/.
* Example:

**hyperparams.yaml**

hyperparams:

seed: 1337

epochs: 7

block\_size: 128

batch\_size: 64

vocab\_size: 16000

n\_layer: 6

n\_head: 8

d\_model: 512

dropout: 0.1

lr: 6e-3

weight\_decay: 0.0

val\_frac: 0.1

Then in train.py:

from omegaconf import OmegaConf

cfg = OmegaConf.load("mainrun/config/hyperparams.yaml")

hparams = OmegaConf.to\_container(cfg.hyperparams, resolve=True)

**3️⃣ Current task**

* **Dataset:** julien040/hacker-news-posts
* **Goal:** Predict the next token in a list of post titles.
* **Pipeline:**
  1. Load titles → train\_titles, val\_titles
  2. Tokenize using BPE tokenizer
  3. Concatenate titles with <eos> as separator
  4. Encode into token IDs → train\_ids, val\_ids
  5. Train GPT model to predict **next token** at each position

**4️⃣ Future extensions**

* Add more models under model/ (e.g., Transformer-XL, LSTM).
* Each model inherits from a **base class** (base.py) to standardize forward and generate.
* Config for each model can be separate YAML files (e.g., config/gpt.yaml, config/lstm.yaml).
* Logging, evaluation, and batch preparation can stay generic in train\_helper.py.

💡 This structure makes it easy to:

* Swap or add models
* Experiment with hyperparameters via YAML
* Keep code modular and maintainable
* Integrate with W&B or other logging systems