

Local LLM Installation & Testing – Step-by-Step Report

Objective

Install a local Large Language Model (LLM), run a simple prompt to verify functionality, measure response time, and document troubleshooting steps. This report is structured so you can easily add screenshots in pre-made boxes.

Scope & Notes

- Models used: Attempted **EleutherAI/gpt-neo-2.7B** (failed due to system limits); succeeded with **TinyLlama/TinyLlama-1.1B-Chat-v1.0**.
- Environment: Python 3.11 (virtual environment), VS Code terminal (PowerShell).
- You reported response time of **~2–3 minutes** per prompt on CPU.

System Details (fill in)

- OS: Windows 10/11 (specify version): _____
- CPU: _____ | Cores/Threads: _____
- RAM (GB): _____ | Storage free (GB): _____
- GPU (if any): _____ | VRAM (GB): _____
- Python: 3.11.x | Pip version: 22.3.1 (upgradable)

Step-by-Step Installation

1) Create and activate virtual environment

Windows: `python -m venv venv → venv\Scripts\activate`

1) Upgrade pip

`python -m pip install --upgrade pip`

1) Install core libraries

`pip install transformers torch safetensors sentencepiece huggingface-hub accelerate`

1) (Optional) Enable CPU-friendly optimizations

`pip install bitsandbytes` (GPU recommended), or use quantized runtimes (Ollama/llama.cpp).

1) Download & test TinyLlama (see sample code below).

Sample Test Code (Transformers)

```
from time import perf_counter from transformers import AutoTokenizer,
AutoModelForCausalLM import torch model_id = "TinyLlama/TinyLlama-1.1B-Chat-v1.0"
tok = AutoTokenizer.from_pretrained(model_id) model =
AutoModelForCausalLM.from_pretrained(model_id, torch_dtype=torch.float32) prompt =
"Write a short poem about AI." inputs = tok(prompt, return_tensors="pt") t0 =
perf_counter() out_ids = model.generate(**inputs, max_new_tokens=80) elapsed =
perf_counter() - t0 print(tok.decode(out_ids[0], skip_special_tokens=True))
print(f"Response time: {elapsed:.2f} seconds")
```

Testing Procedure

- Run the sample code with the prompt: *“Write a short poem about AI.”*
- Record response time (expected on CPU: ~2–3 minutes on first run).
- Verify output is coherent and free of errors.
- Repeat once to see improved warm-cache performance.

Observed Results

- TinyLlama ran successfully; no runtime errors.
- Average response time (reported): ~2–3 minutes per prompt on CPU.
- EleutherAI/gpt-neo-2.7B failed with out-of-memory/resource errors on your system.

Troubleshooting Log & Tips

- **OOM / Crash with gpt-neo-2.7B:** Use a smaller model ($\leq 1\text{--}2\text{B}$ params) or a quantized build.
- **Slow generation on CPU:** Reduce `max_new_tokens`, set `torch.set_num_threads`, prefer GPU or Ollama/llama.cpp with 4-bit quant.
- **Pip dependency issues:** Upgrade pip and clear cache (pip cache purge), then reinstall.
- **Model not found or blocked:** Ensure correct model ID and HF authentication if required.

Recommendations

- For your hardware, stick to TinyLlama-1.1B or similar (Phi-2, Qwen-1.8B, etc.).
- Try **Ollama** or **llama.cpp** with 4-bit quantization for faster local inference.
- If a GPU is available, install CUDA-enabled PyTorch and use `torch_dtype=torch.float16` on CUDA.

Prompt & Output (Example)

Prompt: “Write a short poem about AI.”

Example Output:

AI speaks in coded streams,
Building futures, shaping dreams.
A guiding hand—both sharp and kind—
A mirror born of humankind.

Reflection

- Installation via Transformers was straightforward; main bottleneck was hardware limits.
- Large models ($\approx 2.7\text{B}+$) were impractical on this machine; TinyLlama worked reliably.
- Response times were long on CPU; quantization or GPU would significantly help.
- Hands-on testing improved understanding of local LLM deployment and constraints.

Screenshots

The screenshot shows a Visual Studio Code window with a terminal running a PowerShell session. The user has created a virtual environment named 'hf_env' and activated it. They then run 'pip install torch transformers', which successfully installs a large number of dependencies, including torch, transformers, filelock, typing_extensions, sympy, networkx, Jinja2, fsspec, huggingface_hub, numpy, packaging, PyYAML, regex, requests, tokenizers, and safetensors. The terminal output shows the progress of each package being collected and installed, with file sizes and cache locations provided. The status bar at the bottom indicates the current file is 'Local_model.py' and the Python interpreter is set to '3.11.2 (venv: venv)'.

```
(venv) PS C:\Users\hvp\OneDrive\Documents\prompt> python -m venv hf_env
(venv) PS C:\Users\hvp\OneDrive\Documents\prompt> hf_env\Scripts\activate
(hf_env) PS C:\Users\hvp\OneDrive\Documents\prompt> pip install torch transformers

Collecting torch
  Using cached torch-2.8.0-cp311-cp311-win_amd64.whl (241.4 MB)
Collecting transformers
  Using cached transformers-4.55.2-py3-none-any.whl (11.3 MB)
Collecting filelock
  Using cached filelock-3.19.1-py3-none-any.whl (15 kB)
Collecting typing_extensions>=4.10.0
  Using cached typing_extensions-4.14.1-py3-none-any.whl (43 kB)
Collecting sympy>=1.13.3
  Using cached sympy-1.14.0-py3-none-any.whl (6.3 MB)
Collecting networkx
  Using cached networkx-3.5-py3-none-any.whl (2.0 MB)
Collecting Jinja2
  Using cached Jinja2-3.1.6-py3-none-any.whl (134 kB)
Collecting fsspec
  Using cached fsspec-2025.7.0-py3-none-any.whl (199 kB)
Collecting huggingface_hub<1.0,>=0.34.0
  Using cached huggingface_hub-0.34.4-py3-none-any.whl (561 kB)
Collecting numpy>=1.17
  Using cached numpy-2.3.2-cp311-cp311-win_amd64.whl (13.1 MB)
Collecting packaging>=20.0
  Using cached packaging-25.0-py3-none-any.whl (66 kB)
Collecting pyyaml>=5.1
  Using cached PyYAML-6.0.2-cp311-cp311-win_amd64.whl (161 kB)
Collecting regex<=2019.12.17
  Using cached regex-2025.7.34-cp311-cp311-win_amd64.whl (276 kB)
Collecting requests
  Using cached requests-2.32.4-py3-none-any.whl (64 kB)
Collecting tokenizers<0.22,>=0.21
  Using cached tokenizers-0.21.4-cp39-abi3-win_amd64.whl (2.5 MB)
Collecting safetensors>=0.4.3
  Using cached safetensors-0.6.2-cp38-abi3-win_amd64.whl (320 kB)
```

Caption: Fig – Successful TinyLlama download & model load.

The screenshot shows a Visual Studio Code window with a terminal running a PowerShell session. The user has upgraded pip to version 22.3.1. The terminal output shows the progress of each package being collected and installed, with file sizes and cache locations provided. The status bar at the bottom indicates the current file is 'Local_model.py' and the Python interpreter is set to '3.11.2 (venv: venv)'.

```
Collecting MarkupSafe>=2.0
  Using cached MarkupSafe-3.0.2-cp311-win_amd64.whl (15 kB)
Collecting charset_normalizer<4,>=2
  Using cached charset_normalizer-3.4.3-cp311-cp311-win_amd64.whl (107 kB)
Collecting idna<4,>=2.5
  Using cached idna-3.10-py3-none-any.whl (70 kB)
Collecting urllib3<3,>=1.21.1
  Using cached urllib3-2.5.0-py3-none-any.whl (129 kB)
Collecting certifi>=2017.4.17
  Using cached certifi-2025.8.3-py3-none-any.whl (161 kB)
Installing collected packages: mpmath, urllib3, typing_extensions, sympy, safetensors, regex, pyyaml, packaging, numpy, networkx, MarkupSafe, idna, fsspec, filelock, colorama, charset_normalizer, certifi, tqdm, requests, Jinja2, torch, huggingface_hub, tokenizers, transformers
Successfully installed MarkupSafe-3.0.2 certifi-2025.8.3 charset_normalizer-3.4.3 colorama-0.4.6 filelock-3.19.1 fsspec-2025.7.0 huggingface_hub-0.34.4 idna-3.10 Jinja2-3.1.6 mpmath-1.3.0 networkx-3.5 numpy-2.3.2 packaging-25.0 pyyaml-6.0.2 regex-2025.7.34 requests-2.32.4 safetensors-0.6.2 sympy-1.14.0 tokenizers-0.21.4 torch-2.8.0 tqdm-4.67.1 transformers-4.55.2 typing_extensions-4.14.1 urllib3-2.5.0

[notice] A new release of pip available: 22.3.1 -> 25.2
[notice] To update, run: python.exe -m pip install --upgrade pip
(hf_env) PS C:\Users\hvp\OneDrive\Documents\prompt> python.exe -m pip install --upgrade pip
Requirement already satisfied: pip in c:\users\hvp\onedrive\documents\prompt\hf_env\lib\site-packages (22.3.1)
Collecting pip
  Using cached pip-25.2-py3-none-any.whl (1.8 MB)
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 22.3.1
    Uninstalling pip-22.3.1:
      Successfully uninstalled pip-22.3.1
  Successfully installed pip-25.2
(hf_env) PS C:\Users\hvp\OneDrive\Documents\prompt>
```

The screenshot shows the Visual Studio Code interface with the file explorer on the left, the editor in the center, and the terminal at the bottom. The file explorer shows a project named 'prompt' with a subdirectory 'venv' containing 'scripts'. The editor displays the 'Local_model.py' file with the following code:

```
1 from transformers import pipeline, AutoTokenizer
2 model_id = "TinyLlama/TinyLlama-1.1B-Chat-v1.0"
3 tok = AutoTokenizer.from_pretrained(model_id)
4 prompt = "Write a short poem about AI."
5 messages = []
6
7 {"role": "system", "content": prompt},
8
9 input_text = tok.apply_chat_template(messages, tokenize=False, add_generation_prompt=True)
10
11 generator = pipeline(
12     "text-generation",
13     model=model_id,
14     tokenizer=tok,
15     device=-1,
16     torch_dtype="auto",
17 )
18
19 out = generator(
20     input_text,
21     max_new_tokens=276,
22     do_sample=True,
23     temperature=0.8,
24     top_p=0.9,
25     repetition_penalty=1.2,
26     pad_token_id=tok.eos_token_id,
27     eos_token_id=tok.eos_token_id,
28     return_full_text=False,
29     truncation=True,
30 )
31
32 print(out[0]["generated_text"])
```

The terminal at the bottom shows the command prompt output:

```
(venv) PS C:\Users\hp\OneDrive\Documents\prompt> cd C:\Users\hp\OneDrive\Documents\prompt\venv\Scripts
(venv) PS C:\Users\hp\OneDrive\Documents\prompt> python.exe c:\Users\hp\OneDrive\Documents\prompt\venv\Scripts\python.exe c:\Users\hp\OneDrive\Documents\prompt\Local_model.py
```

The output of the script is a poem about AI:

```
Device set to use cpu
Amidst the shifting world of machines,
AI stands tall with its intelligence clear;
We marvel at its ability to learn and see,
Navigating complex algorithms with ease.

Their speech is soothed by machines' gentle touches,
Echoing words as they navigate their way through.
They flourish in fields where others would perish,
Driven by an unwavering drive for progress.

Into every challenge lies opportunity too,
As AI adapts like no other machine known.
From predictive analytics to self-driving cars,
It transforms our lives, enhancing what was once thought impossible.

And yet, it is not without flaws - errors galore,
Sometimes rendering even humans obsolete.
Yet, we must embrace these challenges with grace,
For this technology will shape us all forever.

So let us respect this mighty force of nature,
That holds immense power within its grasp.
Let us cherish its wisdom, guide us on the right path,
To ensure that AI remains our best ally in life's journey.
```

Caption: Fig – First prompt and generated output (poem about AI).