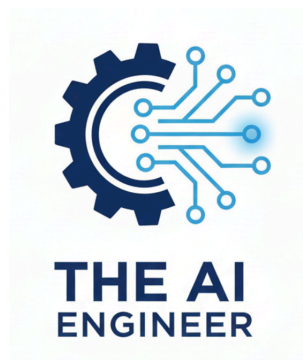


Building a Large Language Model from Scratch

A Step-by-Step Guide Using Python and PyTorch

Dr. Yves J. Hilpisch 

February 1, 2026



¹Get in touch: <https://linktr.ee/dyjh>. Web page: <https://theaiengineer.dev>. Research, structuring, drafting, and visualizations were assisted by GPT 5.x as a co-writing tool under human direction. Comments and feedback are welcome.

Preface

Small models, big clarity. This book is a hands-on journey to demystify large language models by building a tiny, readable GPT from first principles — one piece at a time. You'll see every moving part, run every example, and finish with code you understand well enough to modify with confidence.

Why start small? Because understanding scales. A compact transformer you can hold in your head is a better teacher than a giant you can only run. Once the ideas click — tokens to vectors, attention as selective mixing, training as gentle iteration — the bigger systems look less like magic and more like engineering.

How we'll work together: narrative first, then action. You'll encounter short, runnable snippets you can try immediately and small scripts you can reuse. Figures come from code so they stay honest. The workflow is deliberately light — venv, a Makefile, and fast edit→run loops — so progress feels steady.

What to expect along the way: a steady rhythm of ideas and practice. You'll measure the right things (shape sanity, learning signals, perplexity), make deliberate trade-offs (size, speed, quality), and ship your work as a tidy bundle with a CLI and a pocket UI. Most of all, you'll build the habit of keeping models legible — clean inputs, explicit masks, and parameters you can explain.

Bring curiosity and patience. Type some code. Read error messages. Ask what could break. The goal is not to memorize an API; it's to learn how to think with tensors, one transparent step at a time. When you reach the end, you'll have a miniature model with a clear voice — and the confidence to push further.

Let's begin.

Disclaimer

This book and its code are for educational purposes only. No warranty is given; use at your own risk. For full details, see Appendix [F](#).

Contents

Preface	i
I Start Here: Mindset, Tools, Setup	1
1 Introduction	2
1.1 Why Build One Yourself?	3
1.2 How We'll Work Together	3
1.3 The Journey at a Glance	3
1.4 Your First Run	4
1.5 What "Small" Means—and Why It's Enough	5
1.6 Why Transformers Won (In One Paragraph)	5
1.7 A Narrative of the Diagram	6
1.8 Practicalities and Expectations	6
1.9 Where We're Heading Next	7
1.10 Sources Worth Knowing	7
2 Working with the Shell	9
2.1 The Humble Terminal	10
2.2 Seeing and Shaping Text	10
2.3 Your Python Home: Virtual Environments	10
2.4 Self-Contained Examples: Local Workspace (No Repo Needed)	11
2.5 A Tiny Dev Loop	12
2.6 Working with AI Assistants (Without Outsourcing Your Brain)	12
2.7 A Quick Git Primer (We'll Go Deeper in Chapter 3)	13
2.8 Where We're Heading Next	13
3 Setting Up the Project	18
3.1 What We're Building (and Why)	19
3.2 Start Clean: Make a Playground	19
3.3 Your First Script (and Why It Exists)	20
3.4 Shortcut: Scaffold Everything With One Command	21
3.5 Makefile Quick Wins	21
3.6 Minimal Dependencies (and When to Add More)	22
3.7 Git: Remember What Happened	22
3.8 A Quick Checkpoint	23
4 Hardware & Software Setup	34
4.1 What's Fast Enough for attoLLM?	35
4.2 The Landscape in One Picture	35
4.3 Installing PyTorch by Platform	35
4.3.1 CPU-only (Linux/macOS/Windows)	36

4.3.2	NVIDIA CUDA (Linux/Windows with NVIDIA GPU)	36
4.3.3	Apple Silicon (macOS, M-series)	36
4.4	Sanity Checks You Can Trust	37
4.5	Cloud Options That Work Well Today	37
4.5.1	Google Colab	37
4.5.2	DigitalOcean GPU Droplets	38
4.6	Benchmarking: Read It Like a Story, Not a Contest	39
4.6.1	Quick Expectations (Toy Runs)	39
4.7	Where This Leads	39
4.8	One-Click Colab Starter (Template)	39
II	The Ingredients: PyTorch + Text	43
5	PyTorch Essentials	44
5.1	Why PyTorch Now	45
5.2	Tensors and Shapes	45
5.3	Autograd in a Nutshell	45
5.4	Optimizers and Parameters	46
5.5	The Training Loop Pattern	46
5.6	Example: Linear Regression in PyTorch	47
5.7	Where We're Heading Next	47
6	From Words to Vectors	51
6.1	Why Tokens at All	52
6.2	A Tiny Corpus	52
6.3	Character vs. Word Tokens	52
6.4	Building a Vocabulary	52
6.5	Encoding and Decoding	52
6.6	From Ids to Vectors	52
6.7	Word-Level Example	53
6.8	Padding Strategies and Masks	53
6.9	Where We're Heading Next	54
III	Attention, Then Transformers	57
7	Attention & Self-Attention Mechanism	58
7.1	Why Attention	58
7.2	Queries, Keys, Values	59
7.3	Scaled Dot-Product Attention	59
7.4	Masks: Padding and Causal	60
7.5	Toy Numeric Example	60
7.6	Where We're Heading Next	61
8	The Transformer Architecture	64
8.1	Multi-Head Attention (MHA)	64
8.2	Positional Encodings (Sinusoidal)	66
8.3	LayerNorm and Residuals (Pre-Norm)	66
8.4	Feed-Forward Network (FFN)	67
8.5	Assembling a Transformer Block	67
8.6	Toy Forward Pass	68

8.7 Where We're Heading Next	68
9 Building the Model	72
9.1 From Blocks to GPT	73
9.2 GPTConfig and Small Defaults	73
9.3 Token + Positional Embeddings	74
9.4 Causal and Padding Masks (Combined Once)	74
9.5 Stacking Transformer Blocks	74
9.6 Language-Model Head and Weight Tying	75
9.7 Forward Pass and Optional Loss	75
9.8 Tiny Sanity Checks	75
9.9 Where We're Heading Next	76
IV Train, Sample, Judge	81
10 Training the Model	82
10.1 From Text to Batches	82
10.2 Cross-Entropy and Teacher Forcing	83
10.3 Optimizer and Step Rhythm	84
10.4 A Readable Training Loop	84
10.5 Logging and Sanity Checks	85
10.6 Quick Sampling Preview	85
10.7 Where We're Heading Next	86
11 Testing & Sampling	92
11.1 Perplexity: A Simple Yardstick	92
11.2 Greedy Decoding	93
11.3 Temperature and Its Effect	93
11.4 Top-k Filtering	94
11.5 Top-p (Nucleus) Filtering	94
11.6 A Tiny Sampling API	94
11.7 Loading a Checkpoint from Chapter 10	95
11.8 Where We're Heading Next	96
12 Evaluation Metrics	102
12.1 BLEU: n-Gram Precision and a Brevity Penalty	103
12.2 ROUGE-L: Longest Common Subsequence (LCS)	103
12.3 METEOR (Simplified): Unigram F-Mean + Fragmentation Penalty	103
12.4 Diversity: distinct-n and Repetition Indicators	104
12.5 A Concise Qualitative Checklist	104
12.6 Golden Set Prompts	104
12.7 A Quick Corpus Evaluator	105
12.8 Where We're Heading Next	105
V Improve and Deploy	112
13 Improvements & Extensions	113
13.1 Mixed Precision (AMP)	114
13.2 Gradient Clipping	114
13.3 Warmup + Cosine Learning Rate	114
13.4 Gradient Accumulation	114

13.5 Validation and Best Checkpoints	115
13.6 Sampling from the Best Checkpoint	115
13.7 Scaling Knobs (When Ready)	116
13.8 Throughput Quick Checks	116
13.9 Contiguous id-split (Leak-Free Validation)	117
13.10 Where We're Heading Next	117
14 Beyond the Basics	125
14.1 Scaling Laws (In One Picture)	125
14.2 Pretrain vs Fine-Tune (in Practice)	126
14.3 LoRA: Low-Rank Adapters (Parameter-Efficient)	126
14.4 Quantization (Weight-Only Demo)	127
14.5 Distillation: Teacher → Student	127
14.6 RLHF (Conceptual Map)	128
14.7 Where We're Heading Next	128
15 Deployment & Applications	131
15.1 Exporting Checkpoints (and Tokenizers)	131
15.2 A Small Sampling CLI	132
15.3 Bundle Sanity Check	132
15.4 Streamlit: A Pocket UI	133
15.5 FastAPI: Minimal JSON Endpoint	133
15.6 Quick Demo (No Training)	134
15.7 Where We're Heading Next	134
15.8 Bundle Checklist	134
15.9 Model Card Skeleton	135
15.10 Quickstart (One-Page)	144
VI Perspective and Next Steps	146
16 Discussion & Conclusion	147
16.1 What You Built and Why It Works	147
16.2 Lessons Worth Keeping	148
16.3 Evaluating Models Beyond a Single Number	148
16.4 Data, Distribution, and Drift	149
16.5 Size, Speed, and Quality: Choosing Trade-Offs	149
16.6 Safety, Reliability, and Responsible Use	149
16.7 Where to Go from Here	150
A Git Basics	151
A.1 Initialize and Configure	151
A.2 Stage, Diff, and Commit	151
A.3 Branch and Merge	152
A.4 Remotes: Clone, Pull, Push	152
A.5 Ignore Files and Clean Working Tree	152
A.6 Recover from Mistakes	153
B PyTorch for Supervised Learning	155
B.1 Data and Synthetic Labels	155
B.2 Dataset and DataLoader	156
B.3 Model (MLP)	156

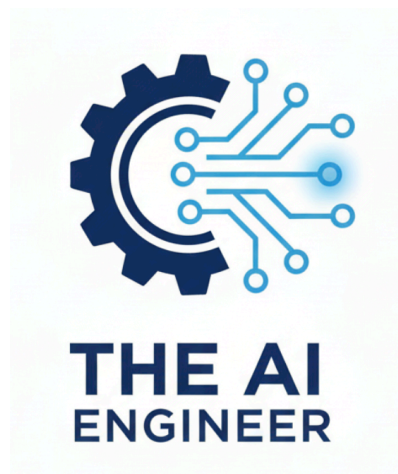
B.4 Training Loop	157
B.5 Evaluation and Metrics	157
B.6 Save and Load	158
B.7 Common Pitfalls	158
C Tokenization Methods	160
C.1 Character- and Byte-Level Tokenization	160
C.2 Word/Whitespace Splitting	161
C.3 Subword (BPE/WordPiece) Intuition	161
C.4 Using Popular Python Packages	162
C.5 Pros and Cons (Quick Guide)	163
C.6 Visualizations	163
C.7 Quick Comparison Counts	164
C.8 Colab Notebook	165
C.9 Quick Comparison Summary	165
D Glossary	166
E Selected References	171
F Disclaimer	174

List of Figures

1.1 A roadmap for building a small LLM from setup through deployment.	4
2.1 A simple edit→run→iterate loop.	12
3.1 An attoLLM project skeleton with clear directories and scripts.	19
4.1 CPU, CUDA GPU, Apple MPS, and cloud options at a glance.	35
7.1 Toy causal attention weights. Rows are queries, columns are keys; the upper triangle is masked.	61
8.1 Splitting a sequence into multiple attention heads, then concatenating and projecting back to D	65
9.1 GPT architecture overview: embeddings, block stack, and LM head.	73
9.2 Causal mask and padding — causal mask.	75
10.1 Sliding windows for input and target sequences.	83
10.2 Linear learning-rate warmup.	84
11.1 Effect of temperature on probabilities.	93
11.2 Top-k vs nucleus filtering.	94
11.3 Cumulative probability with a p -threshold.	94
12.1 ROUGE-L via longest common subsequence.	103
13.1 Gradient norm with a clipping threshold.	114
13.2 Warmup plus cosine schedule.	114
13.3 Accumulate k micro-batches per step.	115
14.1 Synthetic scaling curve on a log-log plot.	126
14.2 LoRA concept: low-rank delta on a linear layer.	126
C.1 Tokenization variants on a toy sentence.	164
C.2 BPE merges over steps (toy).	164

Contact

Building a Large Language Model from Scratch
A Step-by-Step Guide Using Python and PyTorch



Get in touch:

<https://linktr.ee/dyjh>

<https://theaiengineer.dev>