DHBW Ravensburg

Disciplined Vibe Coding

The unreasonable effectiveness of language

Prof. Dr.-Ing. Mark Schutera | July 6, 2026 · *orange disciplined dino*

[1, 3, 6]

1

Foundations

History, Theory and Mechanics

The hottest new programming language is English

[1,3,6]

Software 1.0

Humans write explicit instructions in code, and the machine executes them.

Software 2.0

Gradient descent writes the weights of neural networks from data.

Software 3.0

The program is a natural-language prompt, and the LLM compiles it to behaviour.

LLMs are a **new computing substrate**, with prompts as programs and context windows as RAM.

Raise the floor, keep the bar

Raise the floor. Everyone can now build software they could not build before. Fast.

> State your intent, accept the diffs, run the result, embrace the exponentials and iterate without necessarily touching or reading every line. That working style is **vibe coding**.

Keep the bar. AI engineering means you adjust your workbench to this abstraction layer.

> Most professional work requires rigor. As you will still own the code, the quality, and are accountable for whatever you ship. That working style is what we term **discipline**.

Making the machine understand words.

Word-level vocabulary

`banana` → `[24 907]`

One token, one entry, one id. A larger vocabulary yields longer subwords, representing complex semantics and patterns directly.

Subword vocabulary

`ba` `na` `na` → `[3021, 512, 512]`

Three tokens from two entries, `ba` and `na`; the repeated `na` reuses id 512. Fewer merges keep the vocabulary small and character-based, which improves generalization.

In the beginning was the token: a unit of text, a word or subword. It is the atomic unit a model reads and writes; a vocabulary is the fixed set of tokens it knows. Tokenizing is how text is sliced into entries of that set.

LLMs use vocabularies of 30,000+ tokens.

[1, 3, 6]

[1–4]

Making the machine understand language

LLMs learn the structure and semantics of language by predicting the next token,

$$\hat{y}_{t+1} = \theta(x_{1:t}).$$

The free lunch, an implicit ground truth inside every token sequence, enables unsuperised learning of θ on unlabeled data,

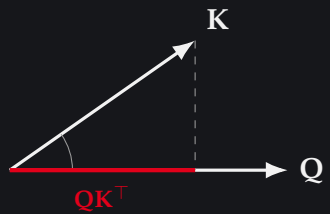
$$\mathcal{D} = \{(x_t, \tilde{y}_t)\}_{t=1}^{N},$$
$$x_{t+1} = \tilde{y}_t.$$

An LLM: the stochastic parrot that predicts the next token.

Attention is all you need

[1, 3, 6]

Attention compares token vectors: the closer a query and a key point in the same direction, the more that token attends to the other, which reflects **semantic similarity**.



The red projection is the dot product $\mathbf{QK}^\top$.

Q, query: what the token is looking for.

K, key: what a token offers to be matched against.

V, value: the information a token passes on once matched.

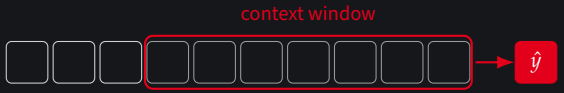
Notice how these computations can be done in parallel, and accelerated by GPUs.

[6]

Context is all you have

[1, 3, 6]

Each token turns into a query and a key, and attention scores every one against every other, a grid of $\mathbf{QK}^\top$ interactions. This runs token by token over one flat sequence, the **context**.



Context is limited - think Goldfish. Its size is capped by GPU memory.

Every token in the window competes for attention:

- your prompt,
- system prompts,
- the files you shared,
- tool and search results,
- the conversation so far,
- and the model's own replies.

Irrelevant tokens still enter the $\mathbf{QK}^\top$ scores and crowd out the signal; doubling the tokens quadruples the compute.

Still a parrot: prompting

[1, 3, 6]

Prompt injection

Steer the model by engineering the input prompt, **injecting instructions** or context that guide and nudge it toward a specific task without retraining.

“The product is awesome is a good review.
The product breaks easily.”
→ [LLM generates the response]

This is zero-shot, the task lives entirely in the tokens you place in the context, so a good prompt is a good program.

[10]

Still a parrot: Instruction tuning

[1, 3, 6]

Instruction Behavior Training

Fine-tune the model to follow an instruction. At this point the *free lunch* is over. An instruct dataset pairs each instruction with a label: Self-Instruct, InstructGPT, FLAN.

Instruction: "Is this review positive? The product breaks easily."
→ Response: "No, it is negative."

Prompting bends behaviour at inference; fine-tuning writes it into the weights. Prompt-to-tune is a recurrent pattern in LLM development.

[11–13]

Vibe a poem on bananas

If not done yet, **set up** your local LLM: clone the repo and run its bootstrap script.

github.com/Quillstacks/disciplinedvibe

- Get comfortable with Open Code
- Write at least one poem, what works well, what does not?
- Share your best poem in the chat.

Not very useful outside the classroom, but it hints at the **semantic capability** of LLMs and their speed, and at the applications they open up.

The muse for your four-line banana poem.

References

1 Tomas Mikolov et al. “Efficient Estimation of Word Representations in Vector Space”. In: *arXiv preprint arXiv:1301.3781* (2013).

2 Jeffrey Pennington, Richard Socher, and Christopher D. Manning. “GloVe: Global Vectors for Word Representation”. In: *EMNLP* (2014).

3 Joseph Weizenbaum. “ELIZA—a computer program for the study of natural language communication between man and machine”. In: *Communications of the ACM* 9.1 (1966), pp. 36–45.

4 Daniel Jurafsky and James H. Martin. “Speech and Language Processing”. In: (2023).

5 Alec Radford et al. “Language Models are Unsupervised Multitask Learners”. In: *OpenAI* (2019).

6 Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

7 Colin Raffel et al. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *arXiv preprint arXiv:1910.10683* (2019).

8 Hugo Touvron et al. “LLaMA: Open and Efficient Foundation Language Models”. In: *arXiv preprint arXiv:2302.13971* (2023).

9 Ashish Vaswani et al. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems* 30 (2017), pp. 5998–6008.

10 Tom B. Brown, Benjamin Mann, Nick Ryder, et al. “Language Models are Few-Shot Learners”. In: *arXiv preprint arXiv:2005.14165* (2020).

11 Xinyi Wang, Jason Wei, Yuxin Wang, et al. “Self-Instruct: Aligning Language Model with Self Generated Instructions”. In: *arXiv preprint arXiv:2212.10560* (2022).

12 Long Ouyang, Jeffrey Wu, Xu Jiang, et al. “Training language models to follow instructions with human feedback”. In: *arXiv preprint arXiv:2203.02155* (2022).

13 Hyung Won Chung, Le Hou, Shayne Longpre, et al. “Scaling Instruction-Finetuned Language Models”. In: *arXiv preprint arXiv:2210.11416* (2022).