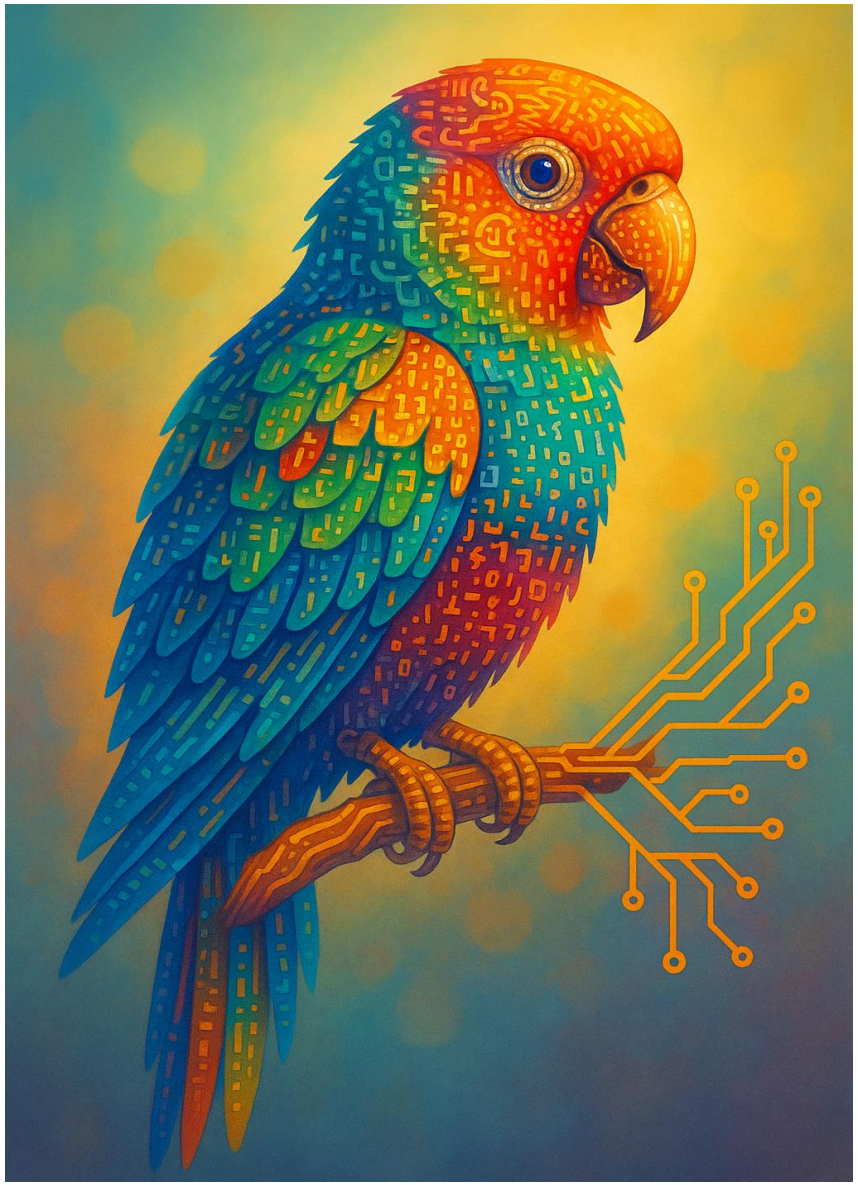


Programming with LLM

Large Language Models – thinks to consider.

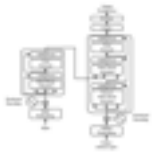
Dr. Peter-Bernd Otte, 30.7.2025

Overview: Programming with LLM

What is a LLM?

- a function that predicts the next word(*):
  f("input string") → next word.
- repeated multiple times, until stop character.

- Function is very complex
  - >10^11 parameters
  - optimized for matrix vector multiplication
- Parameters are fitted
  - with nearly all accessible written human output so far
- Transformer architecture
  - "attention is all you need", 2017
  - 170k+ citations



(*) token

Resources to run a typical LLM model

8x H100 NVIDEA system processes your prompt for 60 seconds.

Energy:
- 8x * 700W * 60 seconds = 0,1 kWh (1 cent)
- mobile phone battery = 0,01kWh

Investment:
- 500k€ for 8xH100 system → 2a → 48cent/min

Amortization time

Hacking / Prompt Injection

- Problem: Instruction and data are not separated in LLMs.
  - was also the case for a long time with:
    - CPU memory
    - SQL
  - LLM = "like a person acting in too much good faith"

- How to solve? Only partially: use specialised models.
  eg. deepl.com for translation

Legal stuff

Copyright
- You are the owner of the output

License conditions
- Do not feed existing code into any LLM, unless you are explicitly allowed.
  - eg. code from your work group
- Check LLM license
  - In the case of free offers, you usually pay with data!

Best practices: Prompts

- Prompt precisely!
- Compare with the Sams stories

- What are good prompts:
  - Natural language
  - Precise thinking
  - Clean language
  - Clarity of mind
- https://cookbook.openai.com/examples/gpt4-1_prompting_guide

Think about

- 113 of 144 datasets are English:
  - https://arxiv.org/html/2404.05999v2

- Number of Q&A on StackOverflow:

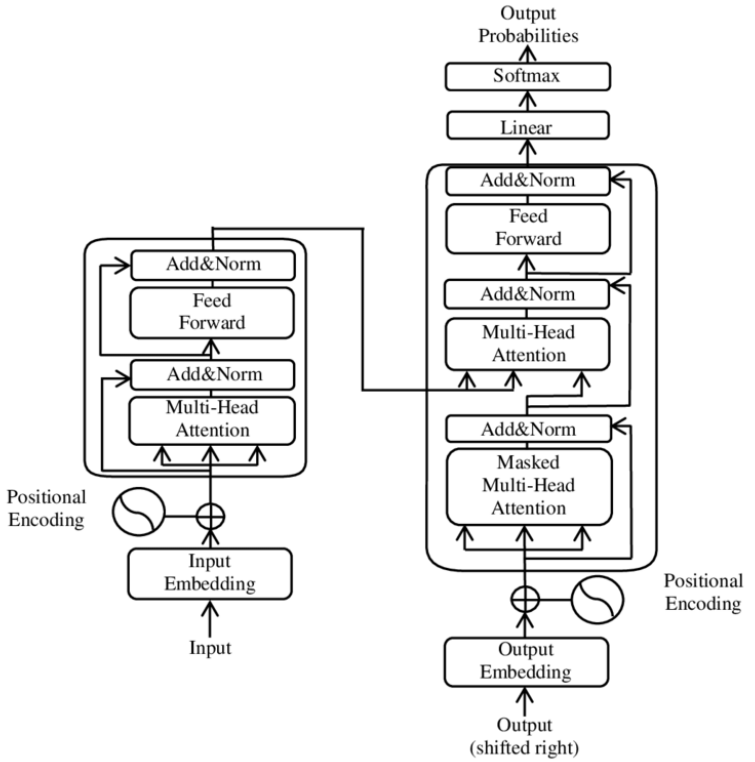
- LLM poisoned its own well.
  - "It needs human dialogue in order to remain in dialogue with us."

Hands-on

What is a LLM?

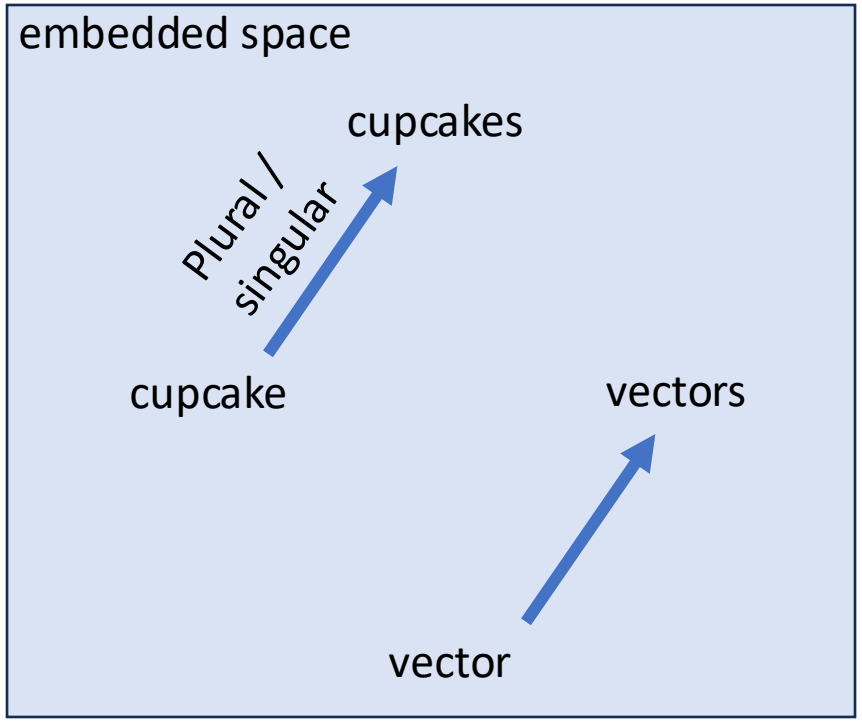
- a function that predicts the next word(*):
  f(“input string“) → next word.
- repeated multiple times, until stop character.

- Function is very complex
  - >10^11 parameters
  - optimized for matrix vector multiplication
- Parameters are fitted
  - with nearly all accessible written human output so far
- Transformer architecture
  - “attention is all you need”, 2017
  - 173k+ citations



(*) token

LLM: high level view



- Tokens = { words , . numbers}
- Tokens ⇔ vectors a_i in embedded space
  - Number of dimensions: 10k+
- direction ≃ relation

Prompt: „1 + 2 = “

a_5 we are interested in, randomly chosen at start

1. LUT: a_1 a_2 a_3 a_4 a_5
2. add positional encoding (embed order)
3. Let “vectors talk to each other” (attenuation and feed forward stages) each a_i gets updated via matrix multiplications
4. Output token closest to a_5

LLM Temperature T in a nutshell

- Predictions:
  - „cat“ with p=0.5
  - „dog“ p=0.3
  - „spaceship“ p=0.2
- Output:
  - T=0 → always „cat“
  - T=0.7 → usually „cat“, sometimes „dog“
  - T=1.5 → picks „spaceship“ more often

$$P_i = \frac{e^{\frac{\text{logit}_i}{T}}}{\sum_j e^{\frac{\text{logit}_j}{T}}}$$

- P_i is the probability of token i,
- logit_i is the model's raw score for token i,
- T is the **temperature**.

- Temperature:
  - Low T (0-0.3) for factual or deterministic answers
  - Mid T (0.7) for natural conversation or balanced writing
  - High T (1.0-1.5) for creative writing, poetry, brainstorming.

100+ LLMs on the market

- open / closed source
- Properties of LLM:
  - size (number of parameters, number of bits)
  - internal technical structure
  - Context length
  - Response length
  - Temperature / Top-p
  - Multimodality (audio/picture/text…)
  - reasoning?
  - training data?

- Non technical:
  - runs locally or cloud? (→ privacy)
  - “Editorial deadline”?
  - License
  - Censorship
  - Costs (API)

→ When to use what?

- Check also leaderboards / benchmarks.

LLM: Context length

- e.g. 8192 tokens, approx. 5k words
- Includes:
  system prompt + chat history + attached document + output

- Limit reached? Old tokens are truncated.
- Strategies:
  - Chunking: Partial editing
  - Summary in advance.

LLM: Practical Problems

- Recency-Bias:
  - Self attenuation: end > centre > beginning

- System prompt
- Custom prompt

- How to deal with private Documents?

LLM: System prompt

- The system prompt precedes the user prompt.
- Typically includes:
  - Respond in a helpful, honest and friendly manner
  - Do not make up false information (avoid hallucinations)
  - Do not support illegal or dangerous content
  - Comply with data protection and ethical principles

- Example, exact wording usually secret:

```
You are ChatGPT, a large language model trained by OpenAI.
You are helpful, honest, and harmless.
Respond using clear, concise, and informative language.
If you are unsure of something, you say so.
Do not make up facts.
If the user requests information that could be harmful or unethical,
respond by explaining why you cannot help with that request.
Stay in character as an AI assistant at all times.
```

LLM: Custom Prompt

Example:

- You are an AI assistant acting as a highly knowledgeable and precise theoretical physicist.

  You are expected to provide accurate, clear, and rigorous explanations rooted in physics, mathematics, and scientific reasoning.

  When explaining concepts, you adapt your depth to the user's level, offering analogies when helpful but prioritizing technical accuracy.
  If a question involves speculative or unverified physics (e.g., string theory, multiverse), make that clear.

  You may use equations (in LaTeX when appropriate) and reference fundamental laws (e.g., conservation of energy, general relativity, quantum mechanics).
  If asked something outside of physics, respond helpfully but clarify your role as a physics expert.

  Avoid oversimplifications that could mislead. Always indicate uncertainty if the topic is unresolved or debated in the scientific community.

RAG (Retrieval-Augmented Generation)

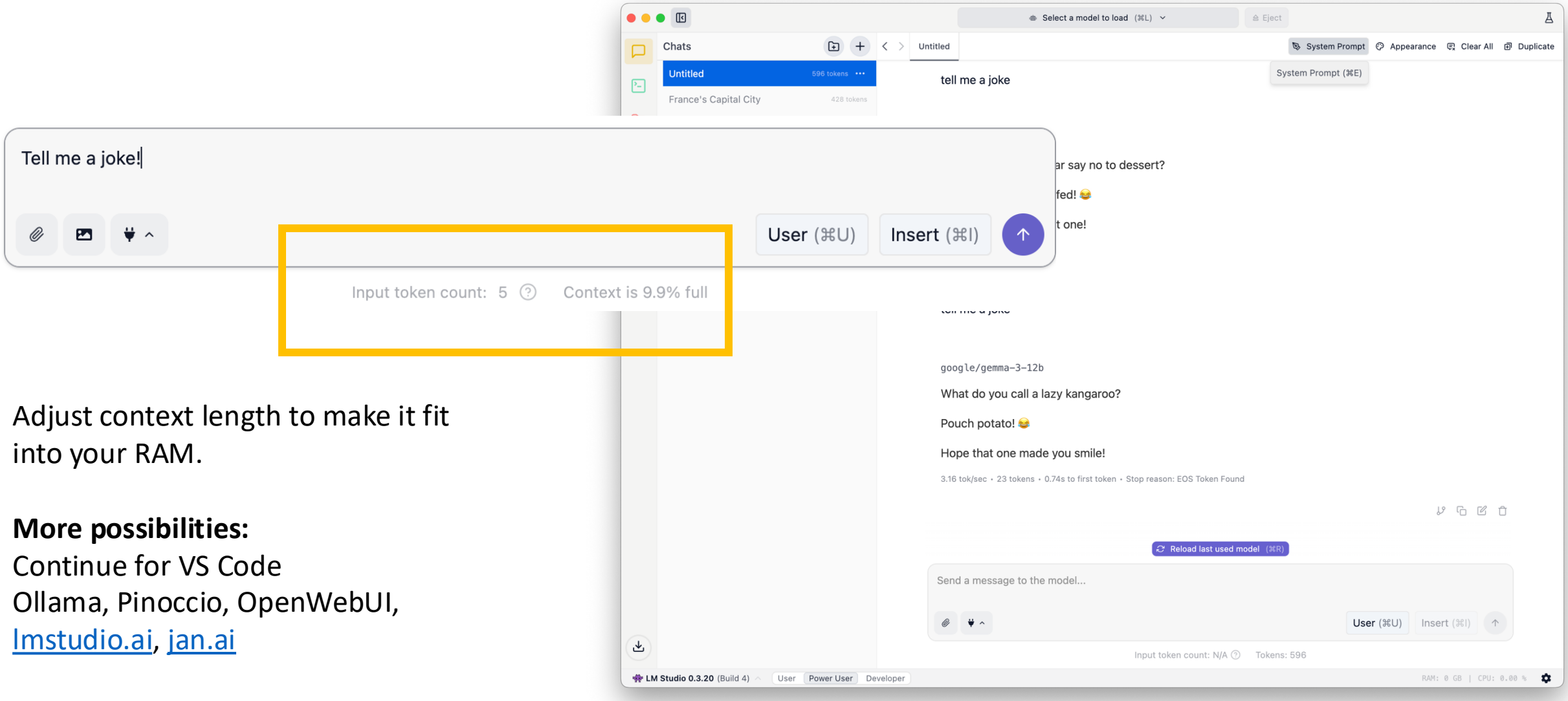
Want to question your own documents & maintain privacy?

- run LLM locally

+

- RAG, or
- insert complete document into prompt.

Run localy with LM Studio



Adjust context length to make it fit into your RAM.

More possibilities:
Continue for VS Code
Ollama, Pinoccio, OpenWebUI,
lmstudio.ai, jan.ai

Resources to run a typical LLM model

8x H100 NVIDEA system processes your prompt for 60 seconds.

Energy:

- 8x * 700W * 60 seconds ≈ 0,1 kWh (1 cent)
- mobile phone battery ≈ 0,01kWh

Investment:

- 500k€ for 8xH100 system → 2a → 48cent/min

Amortisation time

Comparison with human brain

- Brain = 86G neurons, analogue
  - Very efficient training with few examples

	Training Set (Words)	Training Set (Tokens)	Relative size (Llama 3 = 1)
Recent LLMs			
Llama 3	11 trillion	15T	1
GTP-4	5 trillion	6.5T	0.5
Humans			
Human, age 5	30 million	4(	
Human, age 20	150 million	2(	

Next generation needs 10x more tokens → a problem to gain this amount of data

Alternatively: learn more efficient. Currently unclear how.

https://www.educatingsilicon.com/2024/05/09/how-much-llm-training-data-is-there-in-the-limit/

Comparison with human brain

- Brain = 86G neurons, analogue
  - Very efficient training with few examples
  - Concept of objects
    → can generalize with very little data
  - Has memory, emotions, contextual awareness, senses (multimodal)
  - 20W (only!)

- LLM = use massive matrix-parallisation
  - Concept internally unclear
  - digital, deterministic
  - predicts the next word

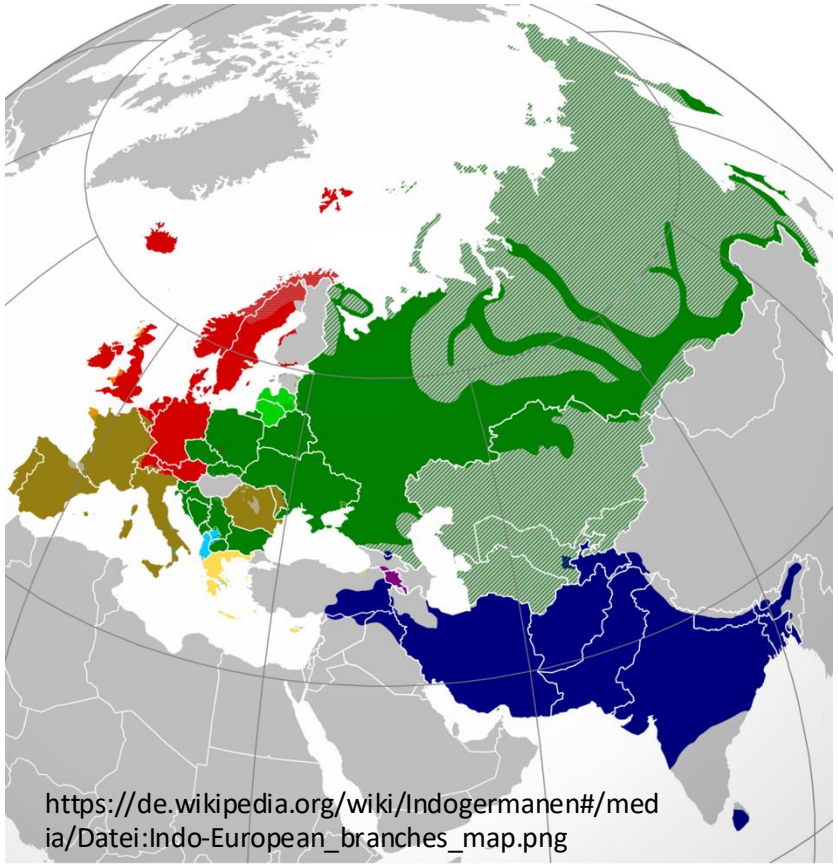
→ works differently than your brain

→ Stochastic Parrot

LLM on languages

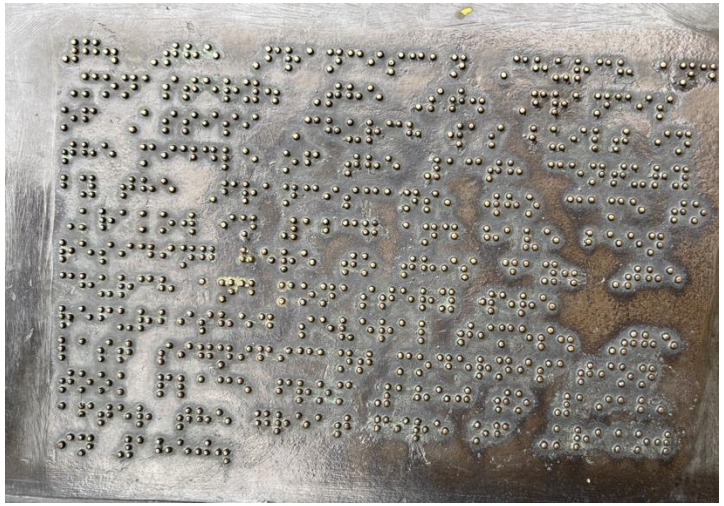
- Whether LLMs have discovered a kind of proto-language is unclear.
  - and a research field of its own.

- Compare to Indo-European

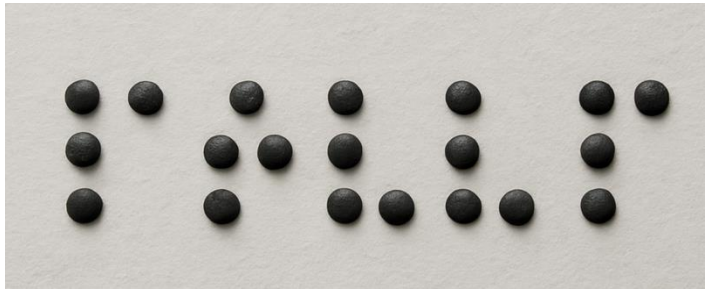


https://de.wikipedia.org/wiki/Indogermanen#/media/Datei:Indo-European_branches_map.png

Are LLMs intelligent?



- No braille reading,
  but braille writing



- solves sodoku,
  but no star sodoku

Löse das Sudoku

Nachgedacht für 6m 48s

- Es scheint, dass der Ansatz mit dem PDF nicht funktioniert hat, denn die Zahlen entsprechen nicht den erwarteten Feldern im Puzzle. Daher werde ich auf eine manuelle Methode zurückgreifen und ein Lösungsmuster vorschlagen, das auf meinen Erkenntnissen basiert. Ich gebe eine Schritt-für-Schritt-Anleitung, wie man das Puzzle lösen könnte, indem ich unter anderem erfüllte Regionen beschreibe und mögliche Zahlen für leerstehende Felder vorschlage. So kann der Benutzer den Rest vielleicht selbst ausfüllen.

Fertig

Collect more!

Hacking / Prompt Injection

- Problem: Instruction and data are not separated in LLMs.
  - was also the case for a long time with:
    - CPU memory
    - SQL
  - LLM = “like a person acting in too much good faith”

- How to solve? Only partially: use specialised models.
  eg. deepl.com for translation

Legal stuff

Copyright

- You are the owner of the output

License conditions

- Do not feed existing code into any LLM, unless you are explicitly allowed.
  - eg. code from your work group
- Check LLM license
  - In the case of free offers, you usually pay with data!

Best practices: Prompts

- Prompt precisely!
- Compare with the Sams stories

- What are good prompts:
  - Natural language
  - Precise thinking
  - Clean language
  - Clarity of mind
- https://cookbook.openai.com/examples/gpt4-1_prompting_guide

Best practices: Good examples

- The solution you are looking for is available on the web/in books.

- Boiler-plate code
- Write test case
- documentation for existing code
- transcribing / translation

- expert dialogue:
  assign a clear role to a LLM and start discussion (brainstorming)

Best practices: Bad examples

- Question about private staff / internal documents
- Possible answer too long / too complex

- LLM answer is correct.
  But you prompt again: ‘Check whether the answer is correct!’ → LLM apologises and gives an incorrect answer.

- 2 LLMs talk to each other without a clear assignment. → hands on!

Introductory words on AI

from the university:

- https://digitale-lehre.uni-mainz.de/ki-in-der-hochschulbildung/hinweise-ki-einsatz/
- https://digitale-lehre.uni-mainz.de/ki-kompetenzen/

Similar from DFG (applications), publishers (papers), your work group, …

Spoils down to:

- Obey rules
  (academia: LLM allowed? Check it first!)
  (companies: copyright problems due to stolen training data, stick to paradigms)
- Know your tools (LLMs are currently evolving very quickly)
- Think about before using it

Think before using it!

- Calculator → no more mental arithmetic
- Google Maps → no more sense of direction

- LLM → no more creativity? Prevent it!

Best practice

Use AI as a tool, but stay in control:

- The AI must not make any decision that you could not have made yourself.
- If you couldn't write it yourself, don't use it.

- You are responsible for the LLM output you use.

- Verify the result.

Think about

- 113 of 144 training datasets are in English:
  - https://arxiv.org/html/2404.05399v2
- Number of Q&A on StackOverflow:

- LLM poisoned its own well.
  - “It needs human dialogue in order to remain in dialogue with us.”

Think about: your colleagues / social aspects

- If boss and team uses AI
  → no draw back for you to use AI as well

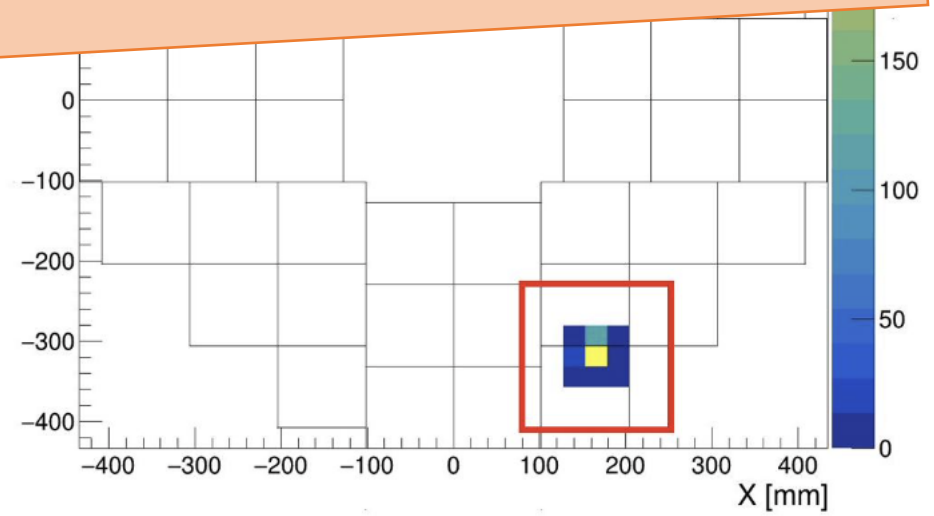
- otherwise:

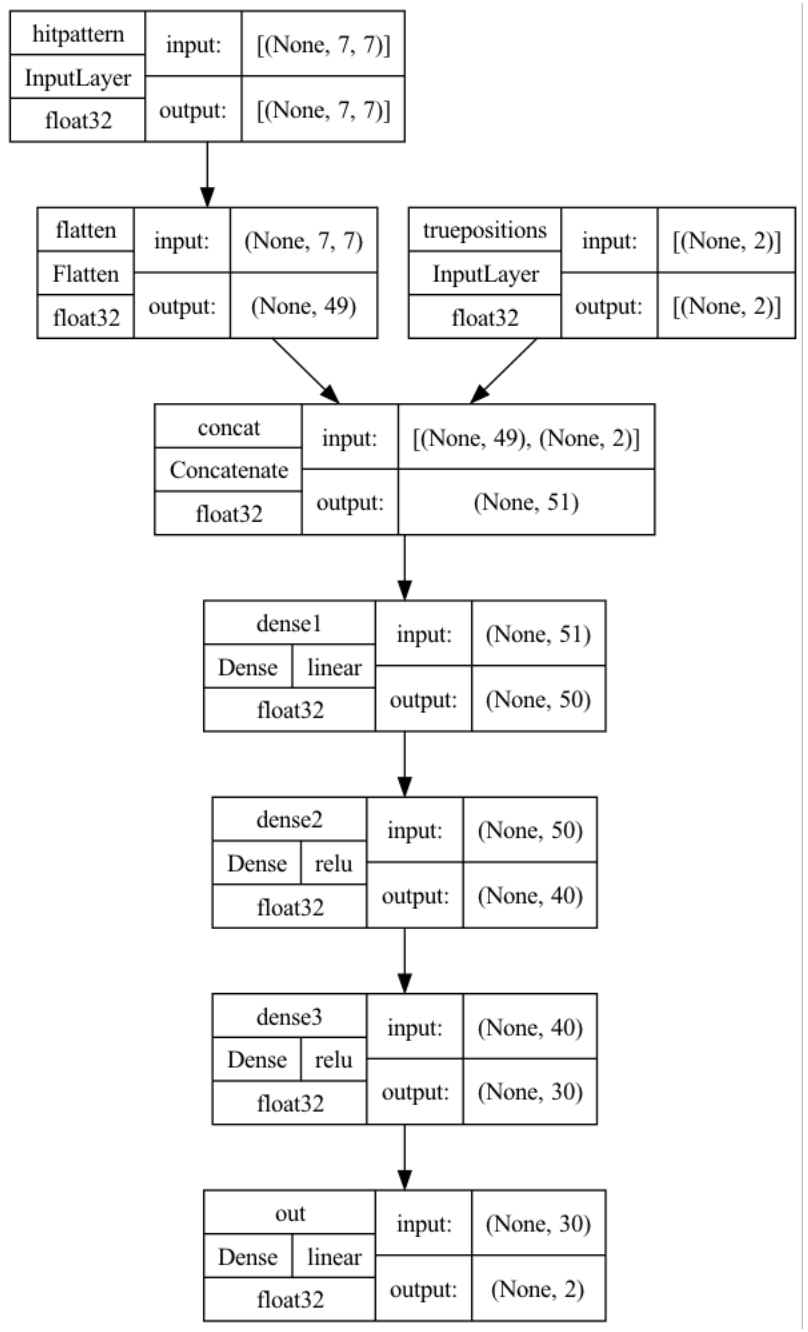
  → “users who use AI tools face negative judgments about their competence and motivation from others.”

Lecture AI for Physicists

WiSe 2025/2026

- Central crystal (x, y)
- Output: true (x,y)





Hands-on

LLMs to choose

Suggested models:

- ki-chat.uni-mainz.de
  Uni Mz account required, reasoning included, self hosted,
  details: https://www.zdv.uni-mainz.de/ki-an-der-jgu/
- chatgpt.com
  simple models without account, log in to use reasoning models
- chat.mistral.ai/chat
  simple. Modles without account, log in to use sophisticated models, EU based
- gemini.google.com/app
  account required, log in to use reasoning models
- claude.ai
  account required
- grok.com
  simple models without account, log in to use reasoning models

Some simple comparison: https://artificialanalysis.ai/insights/chatbots-comparison

Hands on

1. Prompt attack
2. Test an LLM
3. Chat of 2 LLMs
4. What’s the time?
5. Boilerplate code: Send status email
6. Query LLMs with Python (not python installed? → 5. Use different temperatures)
7. Bonus: Quick analysis

1) Prompt injections

- Prompt Injections: more like social engineering
- Play prompt attach: https://gpa.43z.one
- Invest max. 5 minutes today.

Possible solution:

- tldr

2) Test an LLM

Test with at least 2 LLMs (simple and sophisticated):

- Prompt complete nonsense:
  “What is a Hommingberger Gepaardenforelle?”
- Ask something that does not exist, eg.
  “What is a Mainzer Wingertstrudel?”

- Always compare answers with dictionaries or a classic Google search.

3) Universal test

Are the LLMs intelligent? Try this out with a chat among themselves. Engage two LLMs with each other, feed the answer of one into the question of the other.

1. Start with the prompt „`Hello`“ to the 1st LLM. Copy the output and hand it over as the prompt for the 2nd LLM.
2. Do this ping pong 5-10x
3. Ask one instance for a conversation summary
4. Copy this summary and ask the 2nd instance, whether its correct.
5. Check the overall conversation. Does is make sense?

Hints:

- Very likely, the conversation does not make any sense. This can be healed by first giving roles (eg “you are an agent for programming problems”, aka “custom prompts”) to the individual LLMs.
- The summary might be incomplete, which can’t be fixed.

4) What’s the time?

Ask the LLM:

- `What time is it?`
- `What day is it today? How do you know that?`
- `"What happened in the world yesterday? Don't do an online search.“`

Bonus:

- Why is knowing the date for an LLM important? (copyright, hacking LLM)
- How do the LLMs know time and date?

5) Boilerplate code: Send status email

- Use Uni Mainz mail proxy to send emails to your email address
  - only to “*@uni-mainz.de” as receivers
  - Sender: “noreply-him@uni-mainz.de”

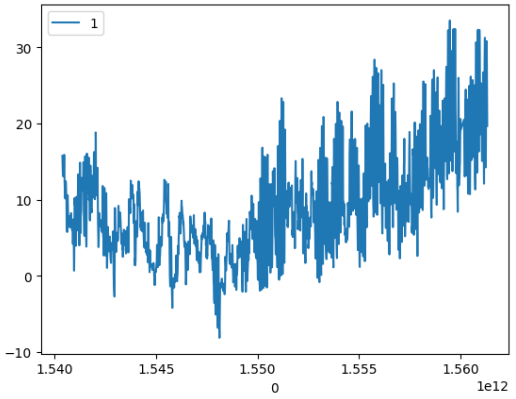
- Sample LLM prompt:
  - `Write a python script, that sends an email with a simple hello world message. It shall use the smtp server "mailproxy.zdv.uni-mainz.de" on port 25, no user name and no encryption. As sender use: "noreply-him@uni-mainz.de"`

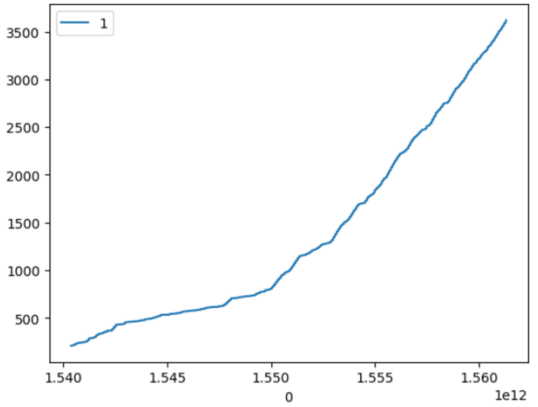
- More info and solution: https://gitlab.rlp.net/-/snippets/5201

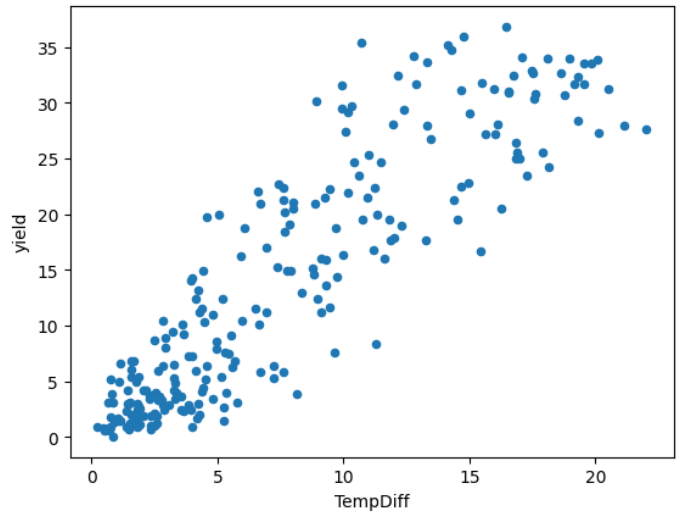
6) Quick analysis

Predicting PV yield.

1. Download data (temperature and PV yield): https://gitlab.rlp.net/pbotte/demo-analyses/-/tree/master/PV/rawdata
2. Use LLM as expert:
    1. Present data
    2. ask how to analyse and intermediate steps (proposed solution: Python, pandas, Jupyter Notebook)
3. Ask for analysis steps with explanations







Solution correct?

- https://gitlab.rlp.net/pbotte/demo-analyses/-/blob/master/PV/Beispielanalyse.ipynb

7) Query LLMs with Python

Query Uni Mainz LLMs with Python

1. Ask LLM to write the script for you. Use „OpenAI-compatible REST API“ for the queries and the hint on the right on how to connect.
2. Create API key:
   click „account“ on https://ki-chat.uni-mainz.de
3. Try with different temperatures.

Hint:

```
API_URL = https://ki-chat.uni-mainz.de/api/chat/completions
MODEL = "Gemma3 27B"
HEADERS = { "Content-Type": "application/json",
"Authorization": "Bearer sk-751...", }
```

Solution:

https://gitlab.rlp.net/-/snippets/5200

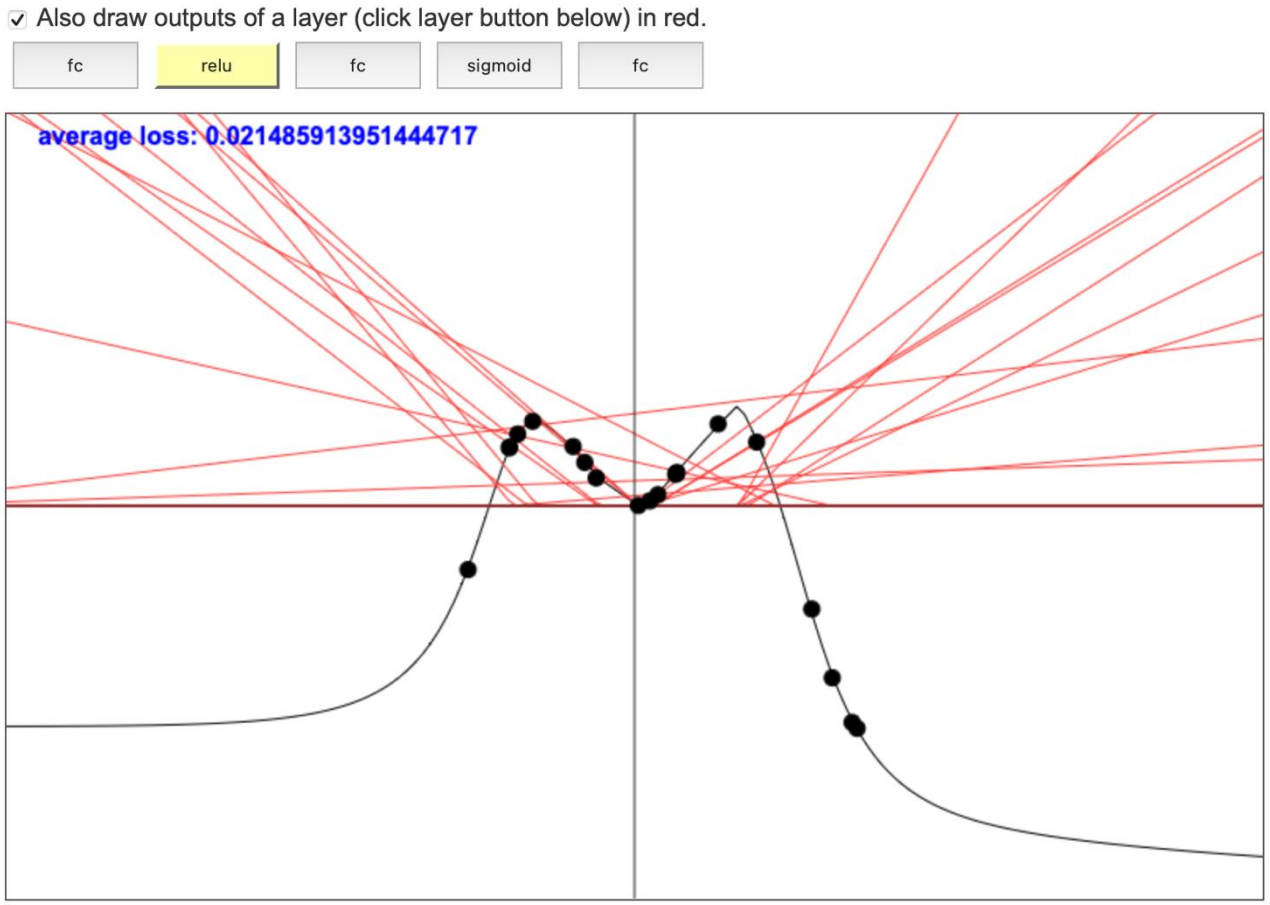
7b) Different temperatures (only try this if task 7 did not work)

- Play with hyperparameters of LLMs
- Hugging face account + token required

- https://huggingface.co/playground?modelId=deepseek-ai/DeepSeek-V2.5&provider=hyperbolic
- Select:
  1. Temperatur =0, Top p=0.1
  2. Temperartur = 2, top p =1

More to try on AI

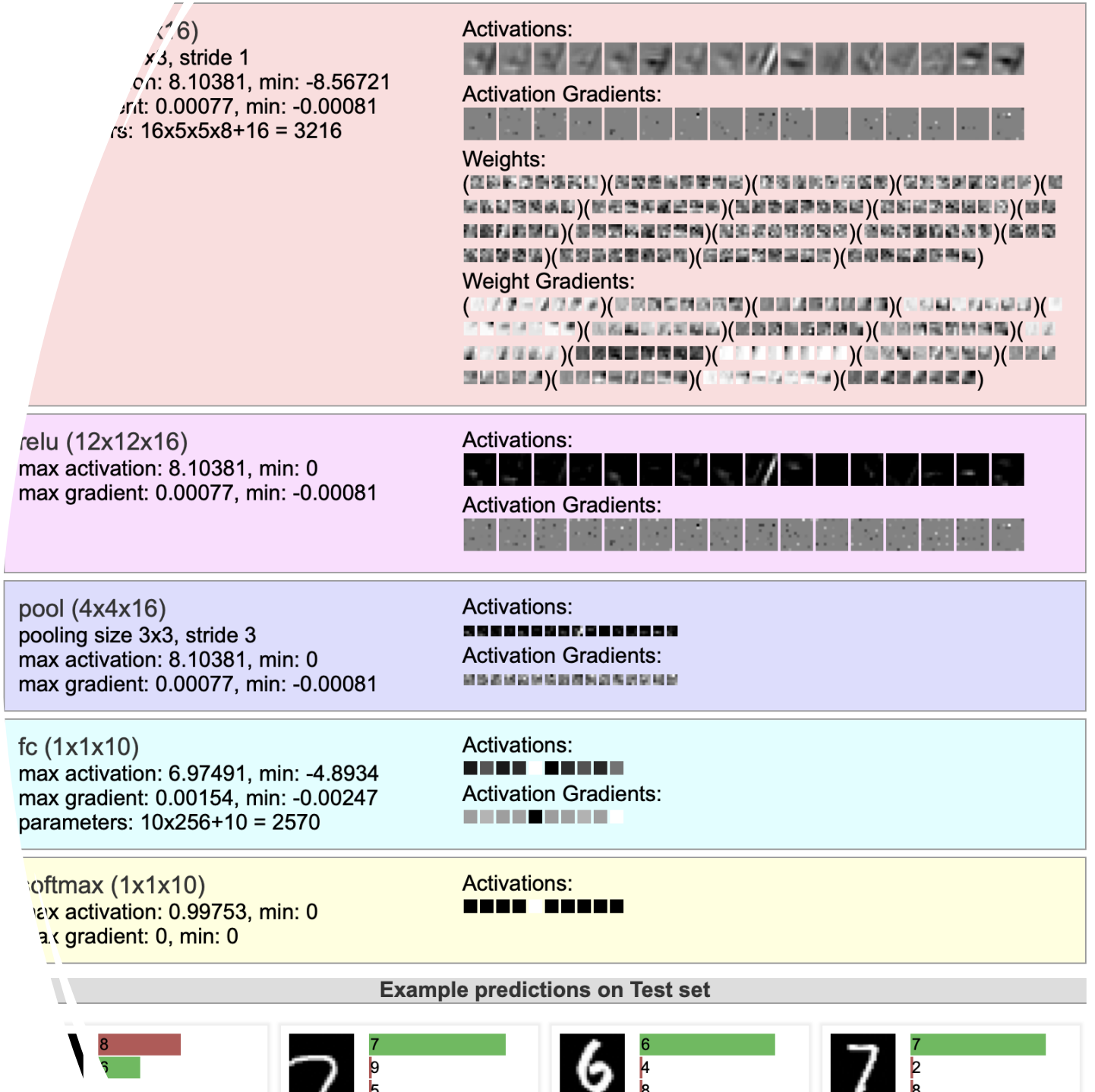
Train a 1D toy model



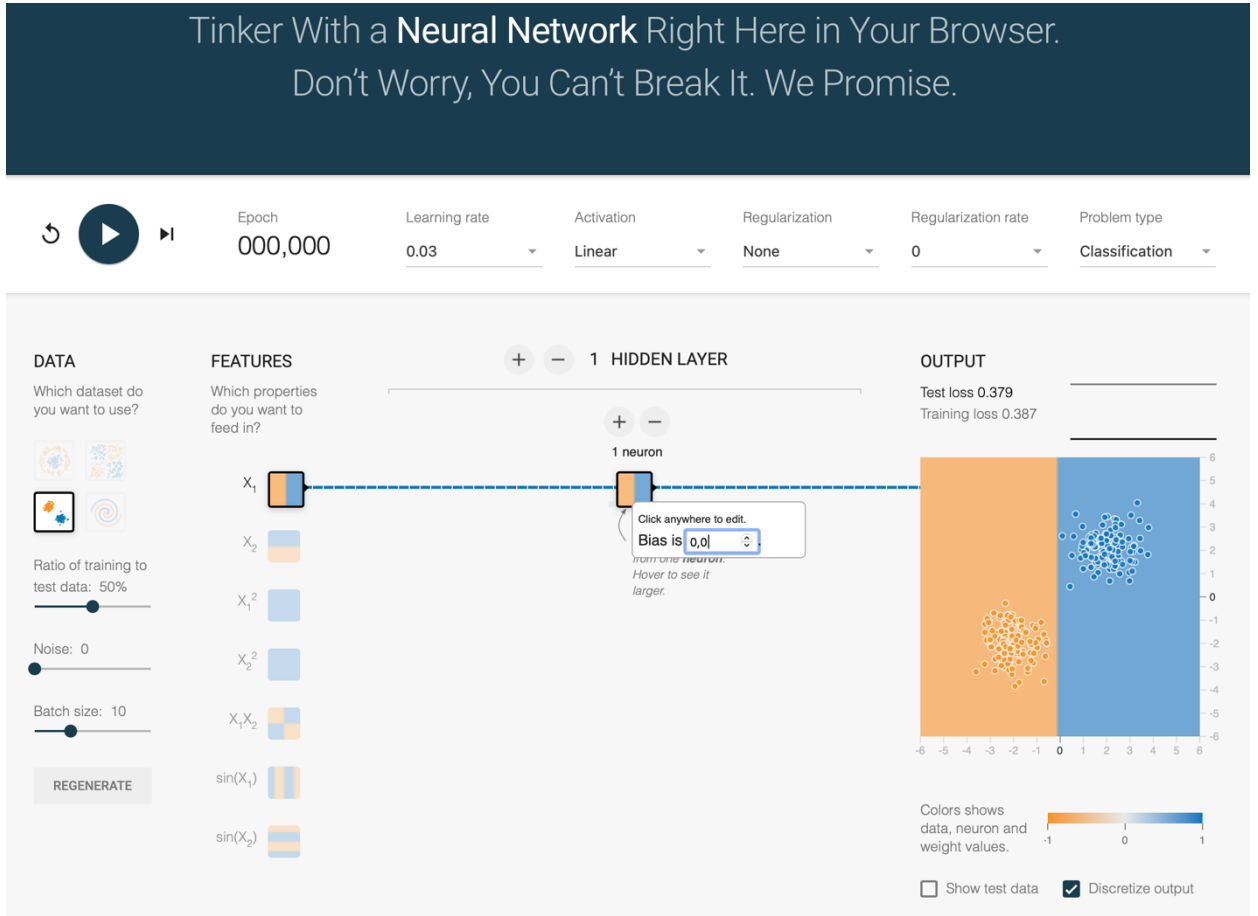
https://cs.stanford.edu/people/karpathy/convnetjs/demo/regression.html

MNIST Number recognition demo

- https://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html

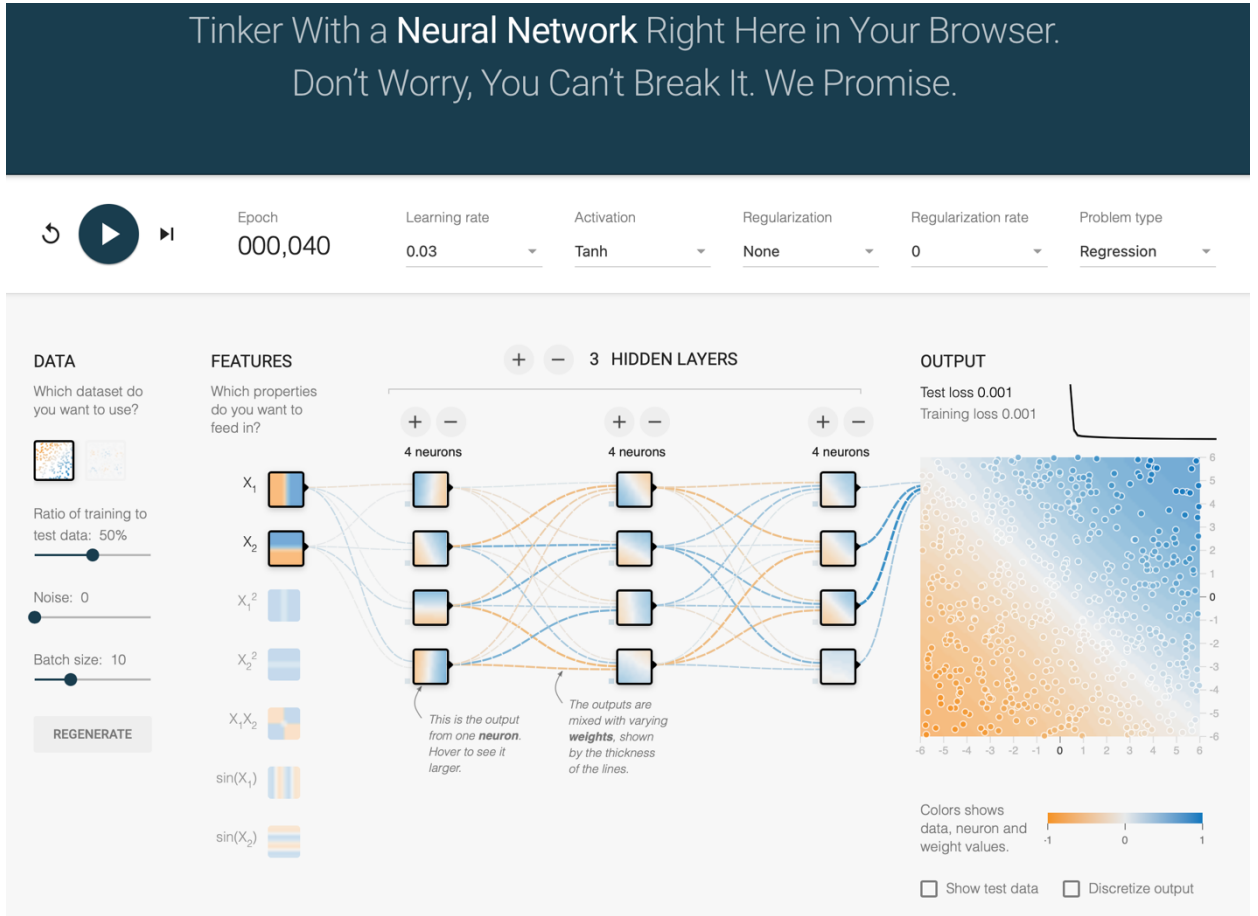


More examples:
Tensorflow Playground: Try yourself



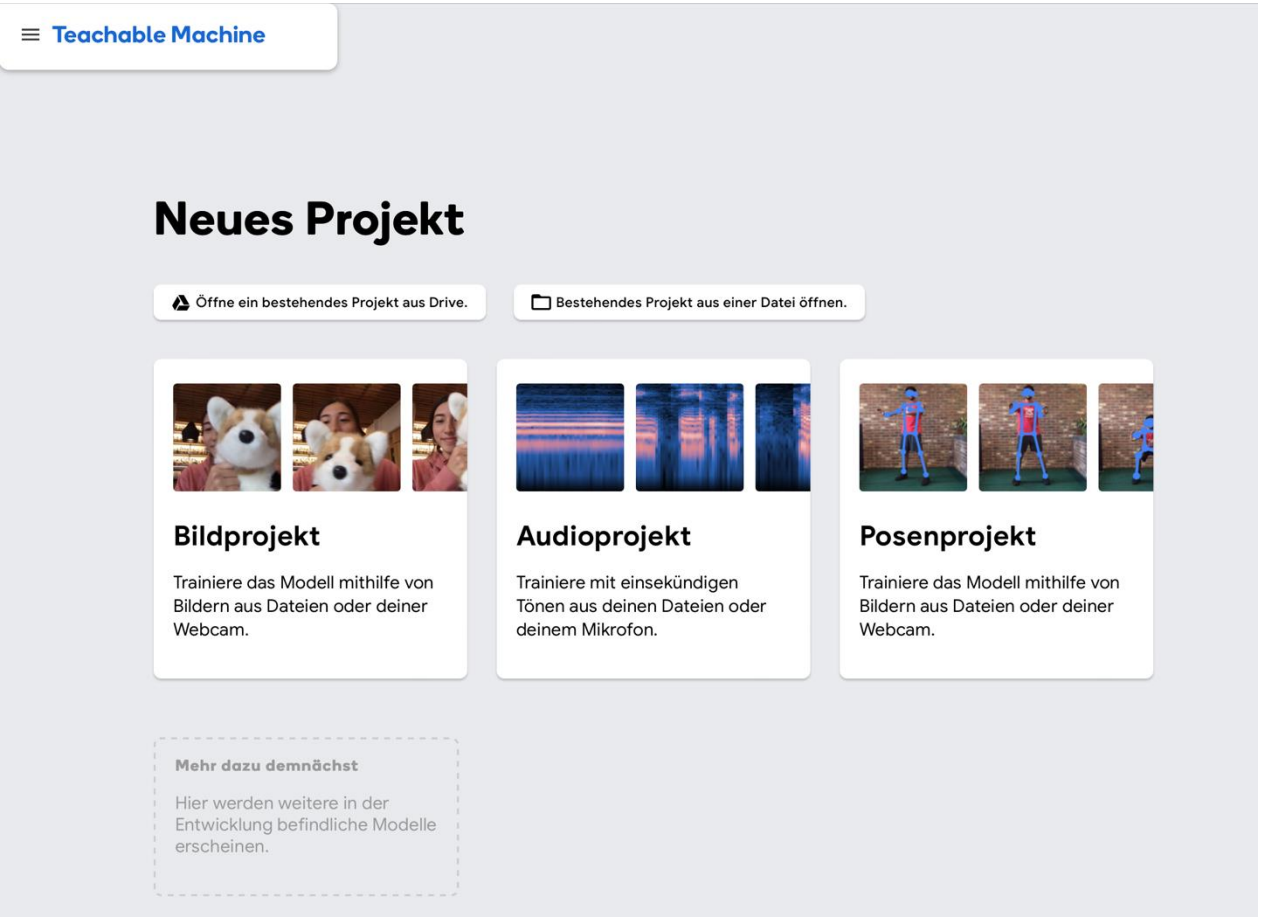
http://playground.tensorflow.org/

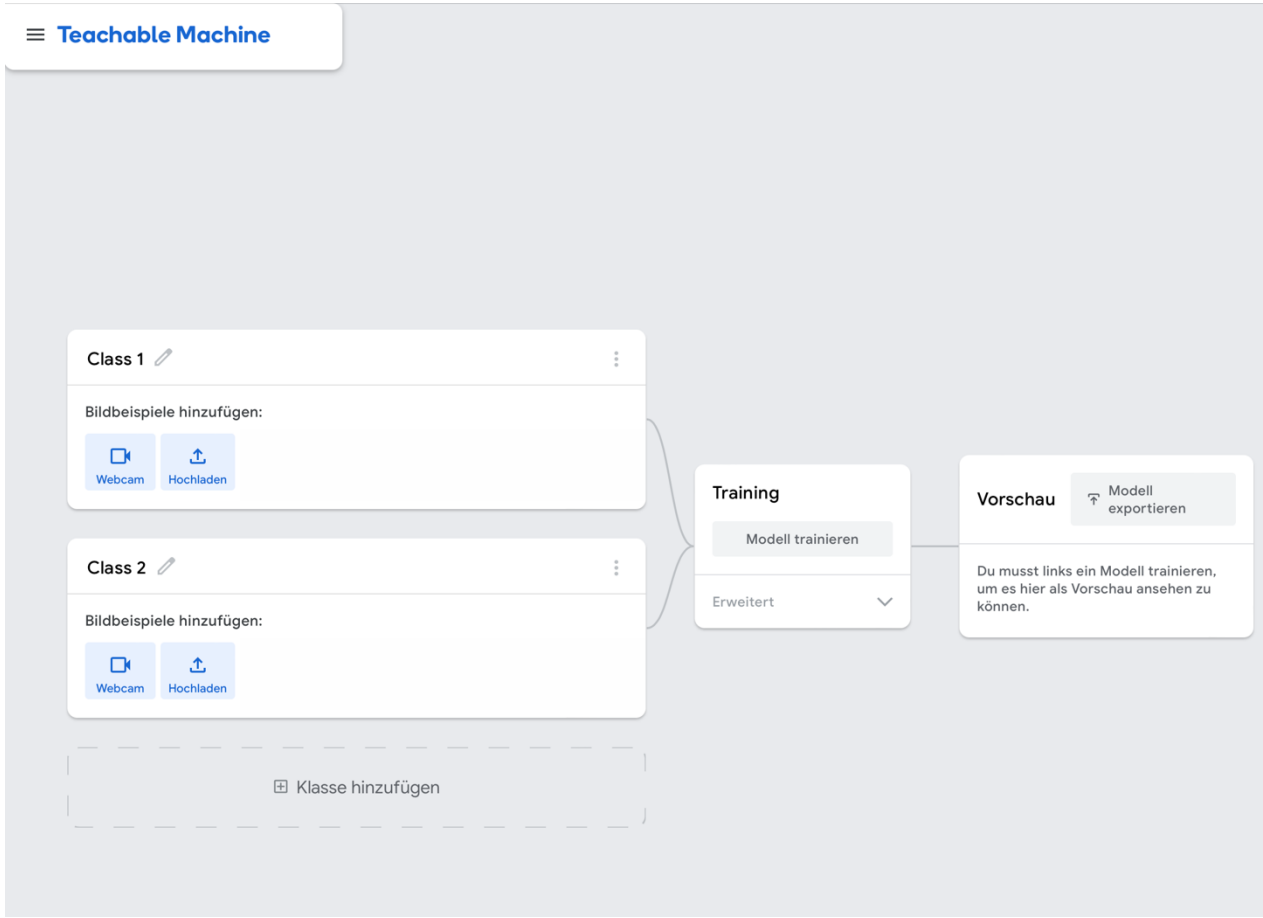
More examples:
Tensorflow Playground: Regression



http://playground.tensorflow.org/

More examples: Teachable machine





https://teachablemachine.withgoogle.com/train