Eigene Sprachmodelle

Nutzung und Feintuning

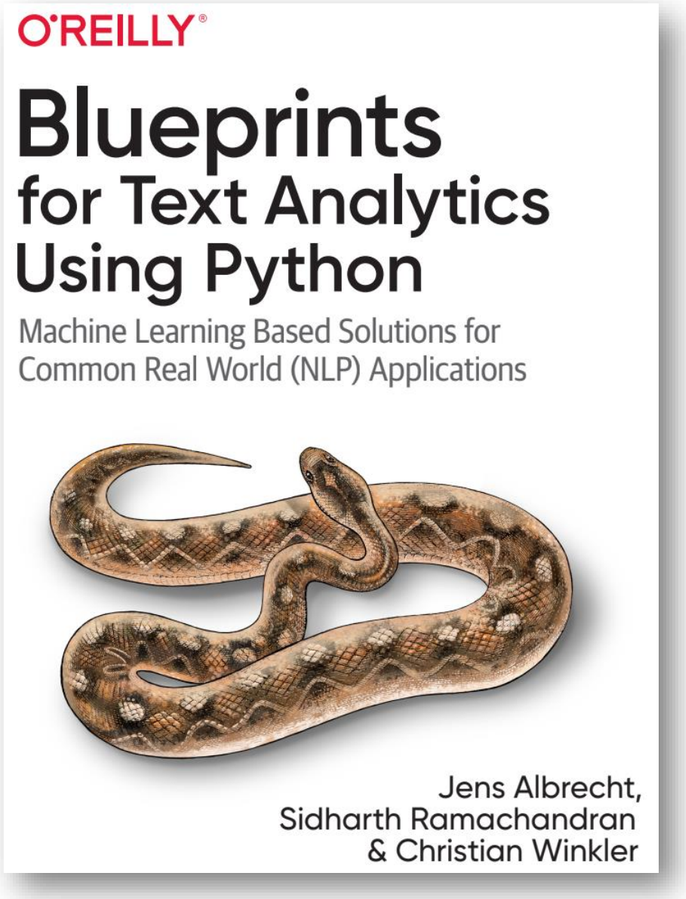
Christian Winkler, datanizing GmbH

KI Navigator, Workshop

https://github.com/datanizing/ki-navigator

Vorstellung und Ziele

Christian Winkler



christian.winkler@datanizing.com

christian.winkler@th-nuernberg.de

Agenda

1. Einführung und Motivation
2. Sprachmodelle: BERT
3. Feintuning von BERT
4. Semantische Ähnlichkeit
5. Generative Modelle
6. Modelle quantisieren
7. Feintuning von LLMs
8. Frontends für LLMs
9. Zusammenfassung und Ausblick
10. Feedback

Einführung und Motivation

Bisher bei der Textanalyse…

Vektorisierung mit Wörtern als Features

- Reihenfolge nicht beachtet
- Zusammenhänge nicht betrachtet
- Semantik geht verloren

Vektorisierung mit N-Grammen als Features

- Reihenfolge beachtet
- Zusammenhänge in Form von Tupeln
- Abstraktion in Form von Semantik fehlt

Worte jeweils einzelne Entitäten

Kontext entscheidet über Semantik!

Wiederholung – die Document-Term-Matrix

Textdaten müssen vektorisiert werden

- Tokenisieren und Stoppworte eliminieren
- Vokabular bestimmen
- Wörter in Dokumenten zählen

Optimierung

- TF/IDF

Unzulänglichkeiten

- Reihenfolge
- Semantik

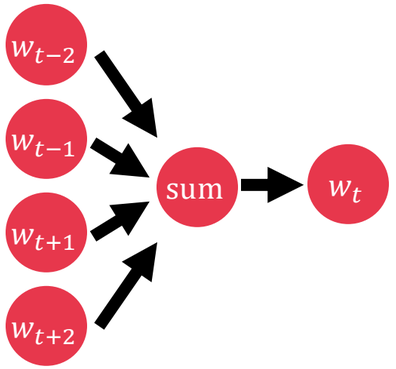
	looking	cheap	flight	where	should	stay	thanks	answer	nearest	train	station	car	airport
Looking for cheap flight?	1	1	1	0	0	0	0	0	0	0	0	0	0
Where should I stay?	0	0	0	1	1	1	0	0	0	0	0	0	0
Thanks for your answer	0	0	0	0	0	0	1	1	0	0	0	0	0
Nearest train station	0	0	0	0	0	0	0	0	1	1	1	0	0
Looking for a car	1	0	0	0	0	0	0	0	0	0	0	1	0
Train to airport	0	0	0	0	0	0	0	0	0	1	0	0	1

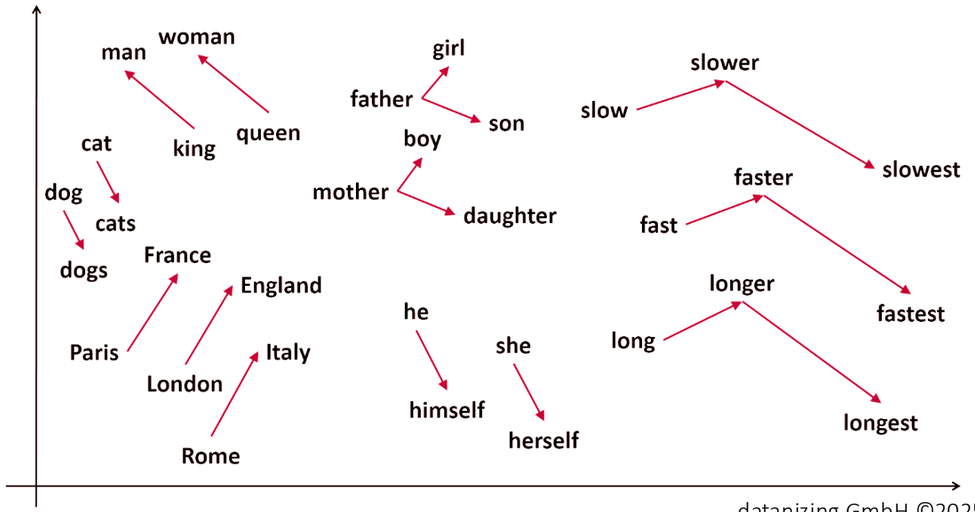
Verbesserung: Wortmodelle wie word2vec

"You shall know a word by the company it keeps."

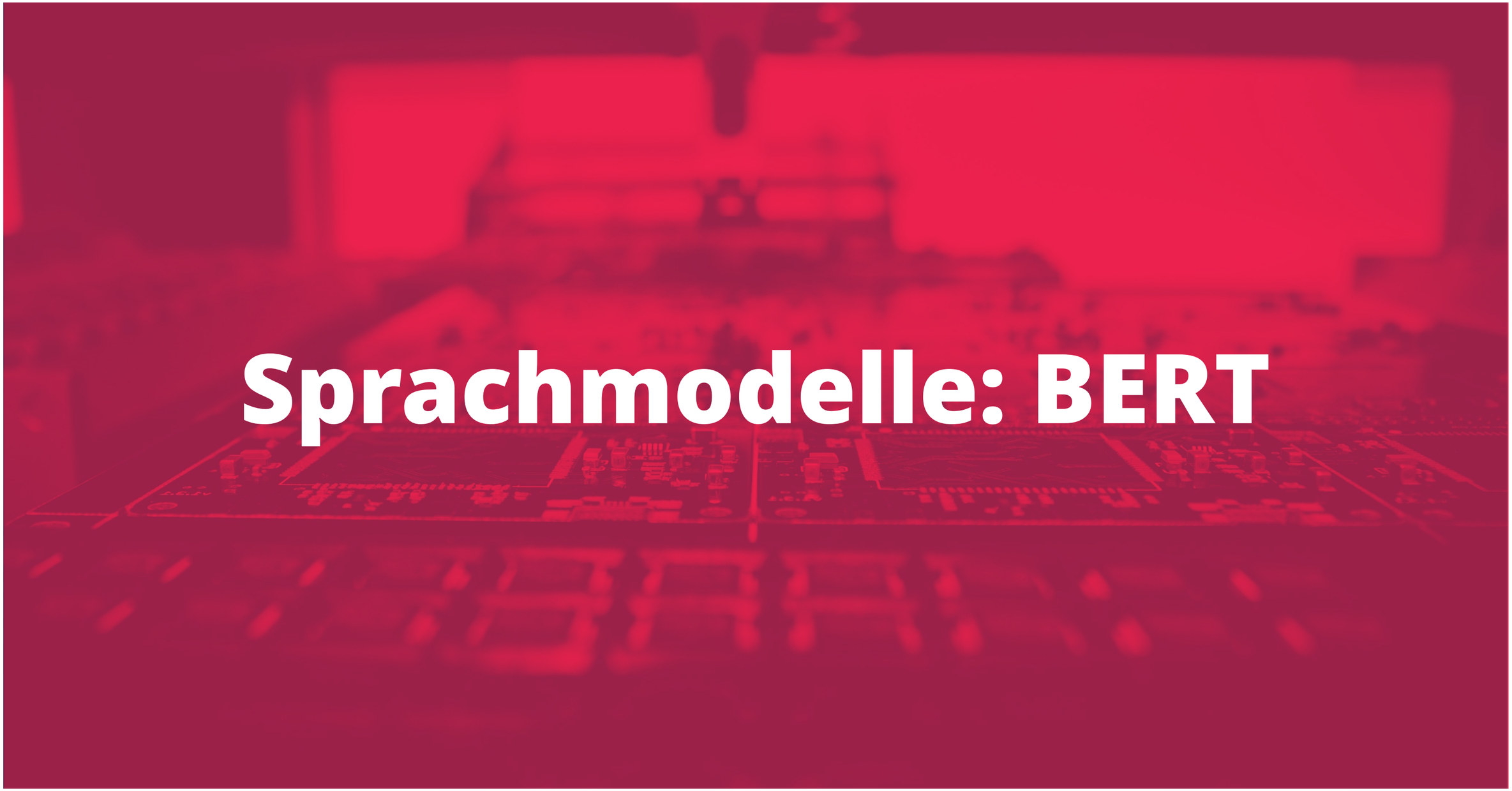
Beispiel: Was ist "tezgüino"? Was ähnelt "tezgüino"?

A bottle of _____ is on the table.
Everybody likes _____.
Don’t have _____ before you drive.
We make _____ out of corn.





EISENSTEIN, JACOB: *Natural Language Processing*. Georgia Tech, 2018. Ch. 14



Sprachmodelle: BERT

Grundfunktionsweise BERT: Transformer-Modell mit Attention

BERT basiert auf einem sog. Transformer-Modell

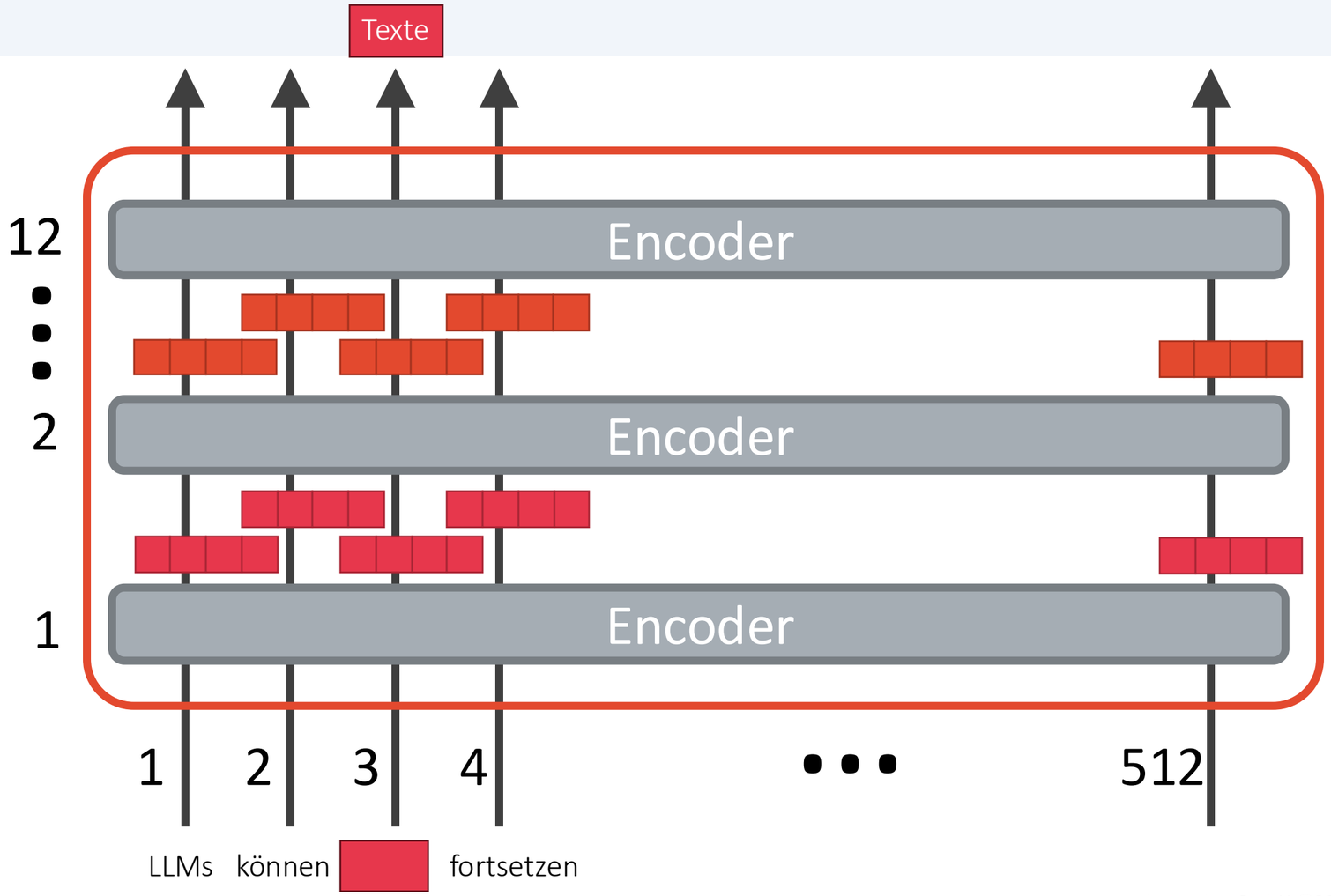
- Vorher verwendet für Übersetzungen
- Sog seq2seq-Verfahren

BERT nutzt nur den Encoder

- Encoder wird kaskadiert
- Unterschiedliche Layer modellieren die Kontextualisierung

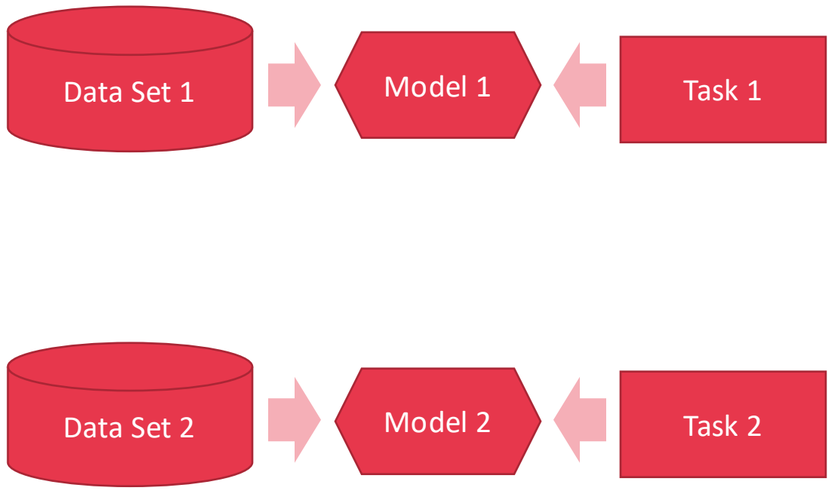
BERT ist ein sehr komplexes Modell

- Base: 12 Layer, 12 Attention Heads, 110 Millionen Parameter
- Large: 24 Layer, 16 Attention Heads, 340 Millionen Parameter
- Sehr, sehr aufwändiges Training
  ➔ Transfer Learning



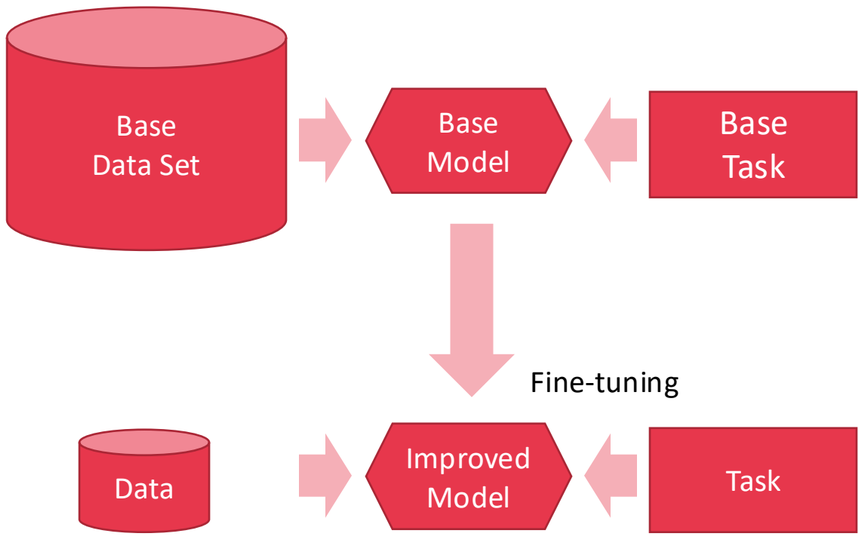
Funktionsweise von Transfer Learning

Klassisches ML



Ein Modell wird für genau eine Aufgabe
bei Null beginnend überwacht trainiert.
Es werden immer viele Trainingsdaten benötigt.

Transfer Learning



Ein Basis-Modell, dass auf einem großen Datensatz
(unüberwacht!) trainiert wurde, wird mit wenig Daten auf eine
spezifische Aufgabe angepasst.

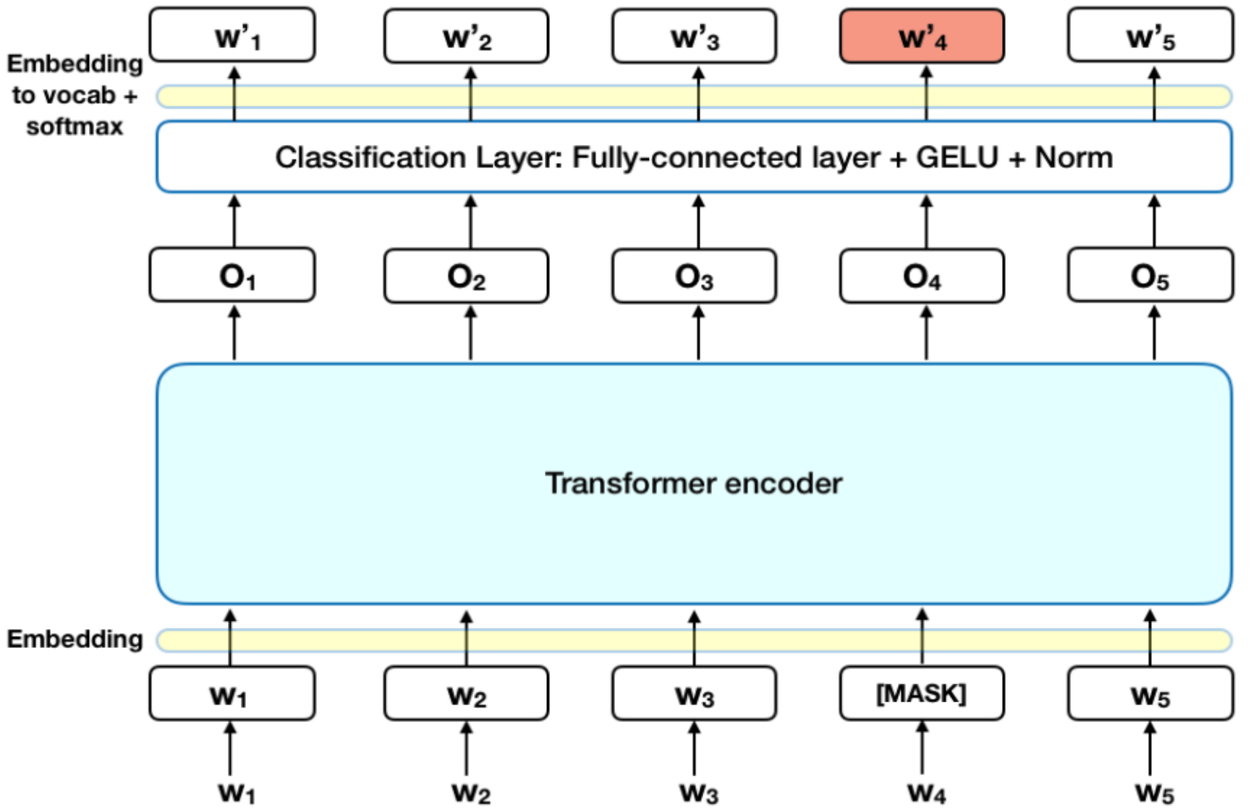
Training auf Vorhersage “versteckter” Worte (Maskierung)

Ich hebe Geld bei der Bank ab.

Auf der Bank hebe ich Geld ab.

Problem: Sprache ist vorwärts und rückwärts kontextualisiert

- Mögliche Lösung durch direktionales Modell
- Bei BERT durch Transformer und Masking



Quelle: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

Was sind *eigene Sprachmodelle*?

Kein Cloud-Zwang

- Ablauffähig auf lokalen Computern („on premise“)
- Parameter liegen vor
- GPU ist oft hilfreich, aber keinesfalls Zwang

Vorteile: Freiheit und mehr Möglichkeiten

- Datenschutz und Datensicherheit
- Experimente möglich
- Anpassung an eigene Bedürfnisse
- Finetuning

Feintuning von BERT

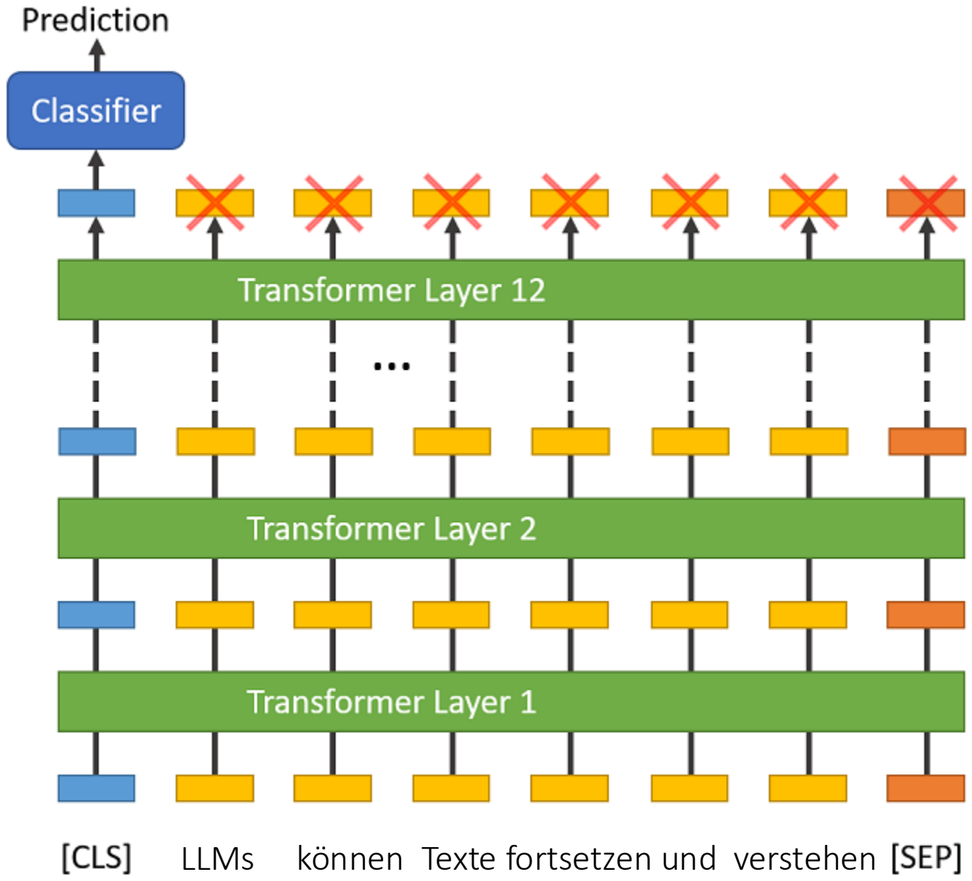
Finetuning für Klassifikationsaufgaben

Grober Ablauf des Trainingsprozesses:

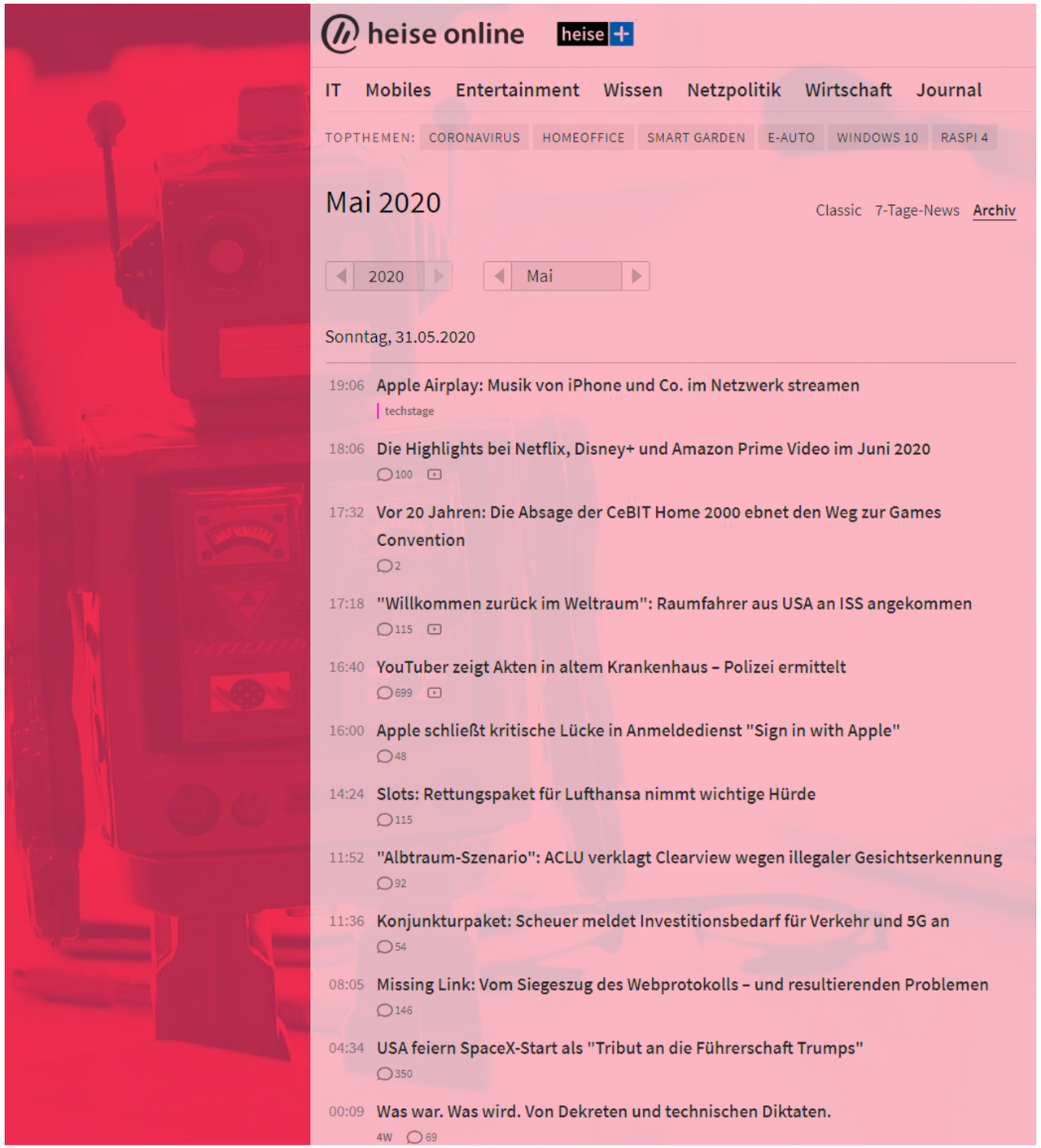
- Embedding der gesamten Sätze berechnen
- Berechnete Embeddings sind hoch kontextualisiert
- Sehr viel (alle?) Information steckt daher im kontextualisierten Embedding des ersten Tokens (der gar nicht zum Satz gehört)
- Embedding-Vektor von [CLS] wird durch Classifier verarbeitet
- Iteration mit Anpassung der Gewichte im letzten Layer

Klassifikationsprozess:

- Kontextualisierung des gesamten Satzes
- Klassifikation nur mit (kontextualisiertem) [CLS]



Quelle: https://mccormickml.com/2019/07/22/BERT-fine-tuning/



Unser Use Case

Heise Newsticker

“Instanz” für Tech- und Computer-News im deutschsprachigen Raun

"Willk
115

YouTu
699

Apple
48

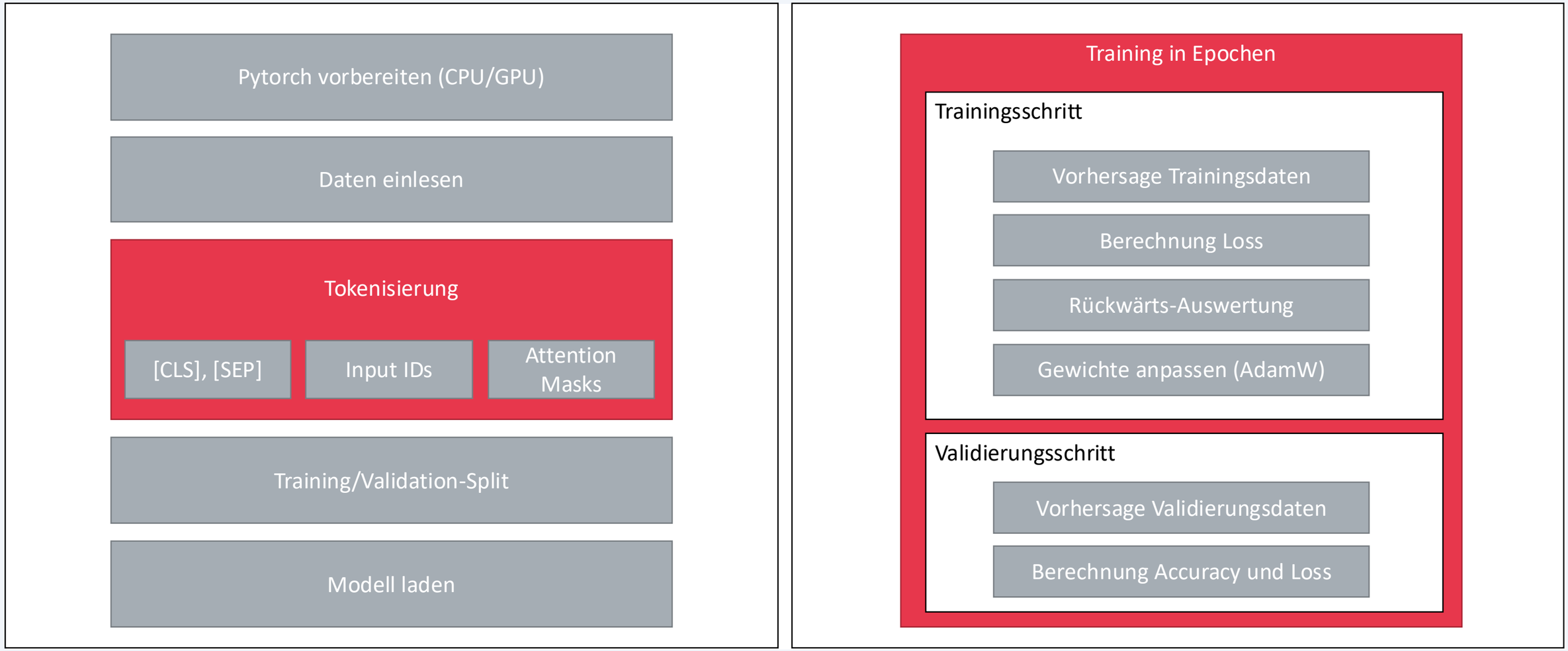
Optimierung

Welche Artikel werden besonders gern kommentiert?

Höhere Reichweite

Mehr Umsatz

Ablauf: Finetuning von BERT



Interaktiv / Jupyter

Semantische Ähnlichkeit

Ist BERT dafür ausreichend?

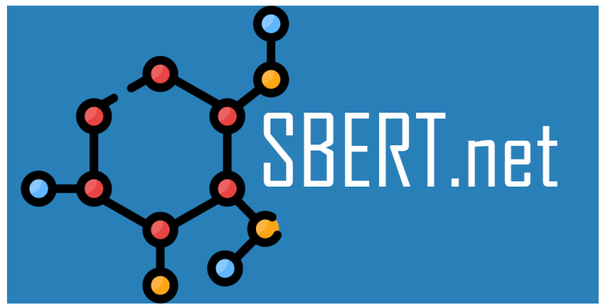
Fast, aber noch nicht ganz

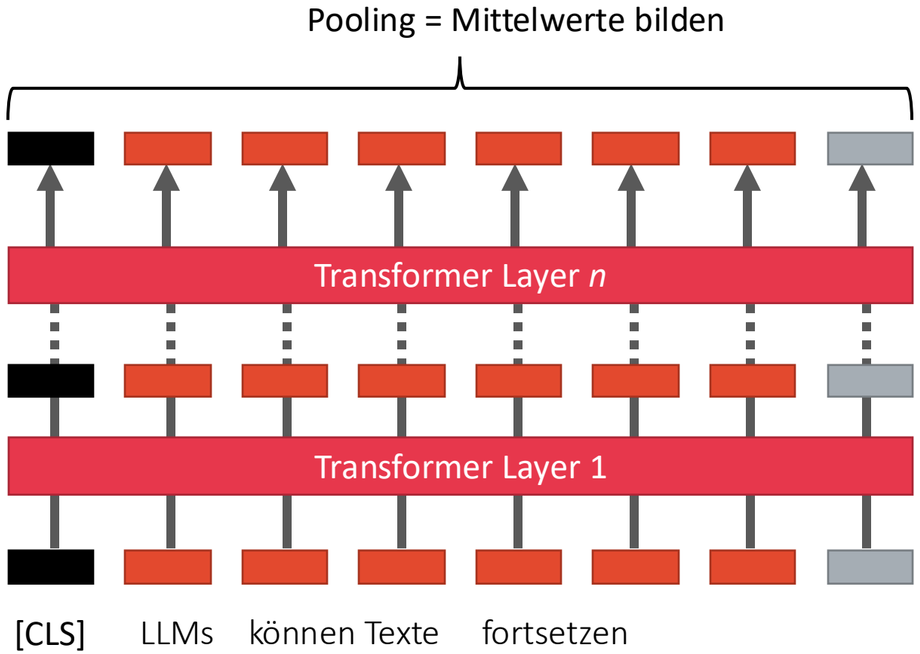
Modell ist darauf angepasst, fehlende Wörter zu erraten

Für Ähnlichkeit wird ein etwas modifiziertes Modell benötigt

Zum Glück ist das bereits verfügbar

Modell heißt SBERT (https://sbert.net) und kommt sogar aus Deutschland





Ablauf

Daten laden

Satz-Fragmente berechnen

Embedding für jeden Satz berechnen

Embeddings speichern

Sätze speichern

Search-Embedding berechnen

Ähnlichste Sätze finden

Nach Ähnlichkeit sortieren

Finetuning eigener Embedding-Modelle

Warum sollte man das machen?

- (Bessere) Abbildung domänenspezifischen Vokabulars
- Bessere Abbildung von Ähnlichkeiten

Wie geht das?

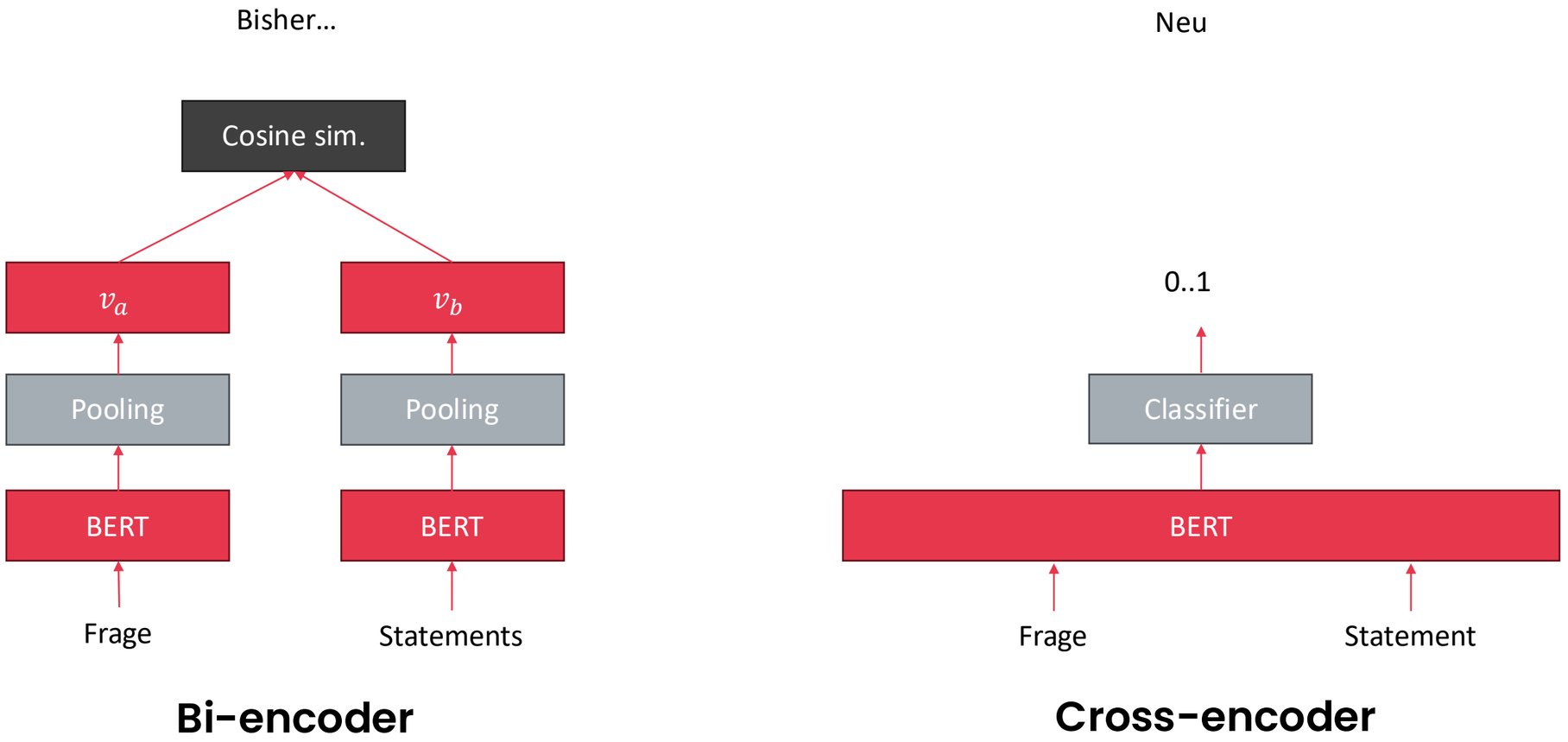
- Erzeugung von ähnlichen (oder weniger ähnlichen) Textfragmenten
- Annotation mit Score
- Finetuning mit BERT oder SBERT

Ist das nicht extrem aufwändig?

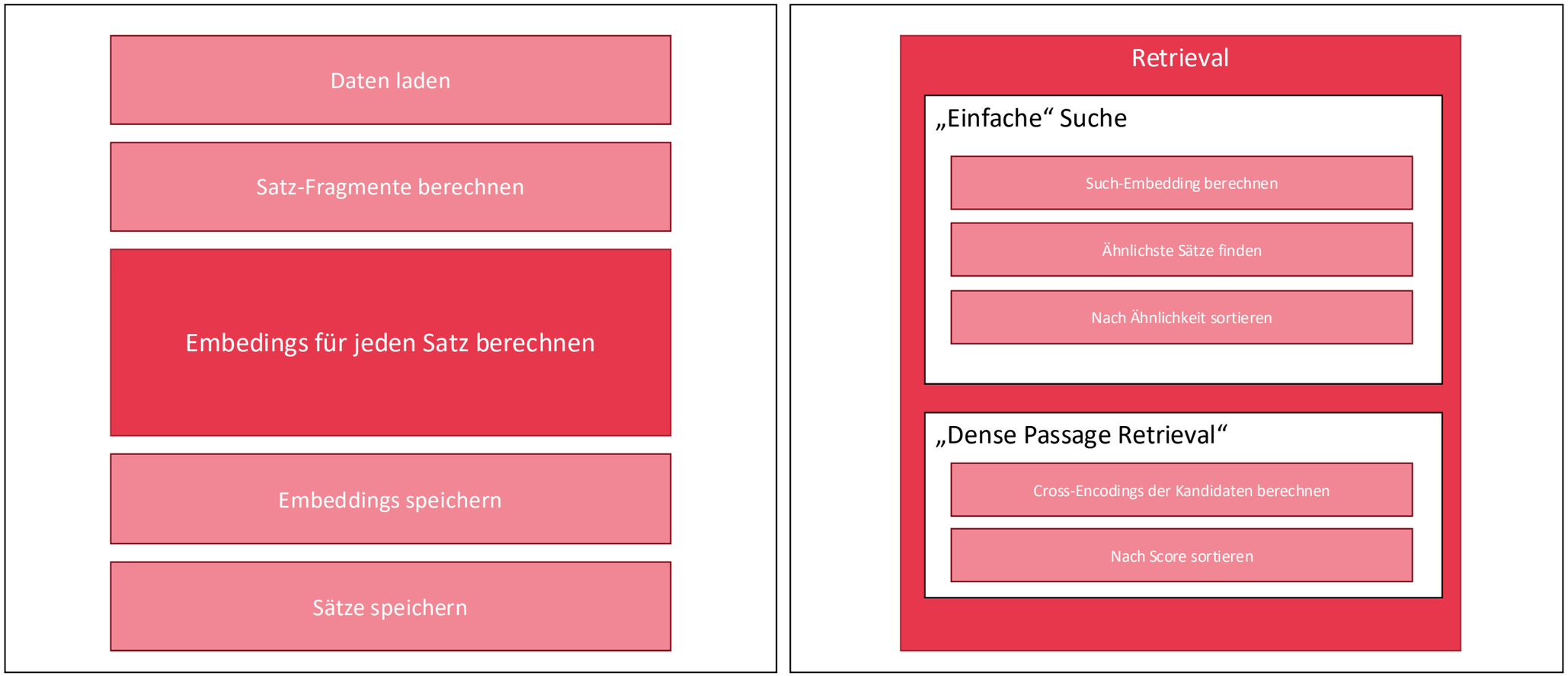
- Text lässt sich generativ erzeugen, Verbesserungen relativ schnell erkennbar
- Mit GPU keine langen Wartezeiten

Optimierung durch Cross-Encoder

Funktionsweise Cross-Encoder



Verbesserung der bisherigen Lösung



Nutzung zum Information Retrieval

Beste Ergebnisse für Ähnlichkeiten von Sätzen

- Codierung des Satzes (z.B. der Frage) notwendig **inkl. gleichzeitiger Codierung** der Referenz (des Ergebnisses)
- Sehr viele Codierungen
- Sehr gut, aber sehr langsam

Grundidee der Sentence Transformers

- Unabhängig Codierung
- Nutzung des Ähnlichkeitsmaßes

Weitere Optimierung durch Cross-Encoder

- Vorher Verkleinerung der Ergebnismenge
- Wartezeit verkürzen

Tausende Aussagen

➔ Tausende Codierungen für jede Suche notwendig (!)

Tausende Codierungen einmal notwendig

➔ Für jede Suche eine weitere Codierung

Tausende Codierungen einmal notwendig

➔ Für jede Suche eine weitere Codierung
➔ Für den Filter weitere 200 Codierungen
➔ Deutlich verbesserte Ergebnisse

Finetuning eigener Cross-Encoder

Warum sollte man das machen?

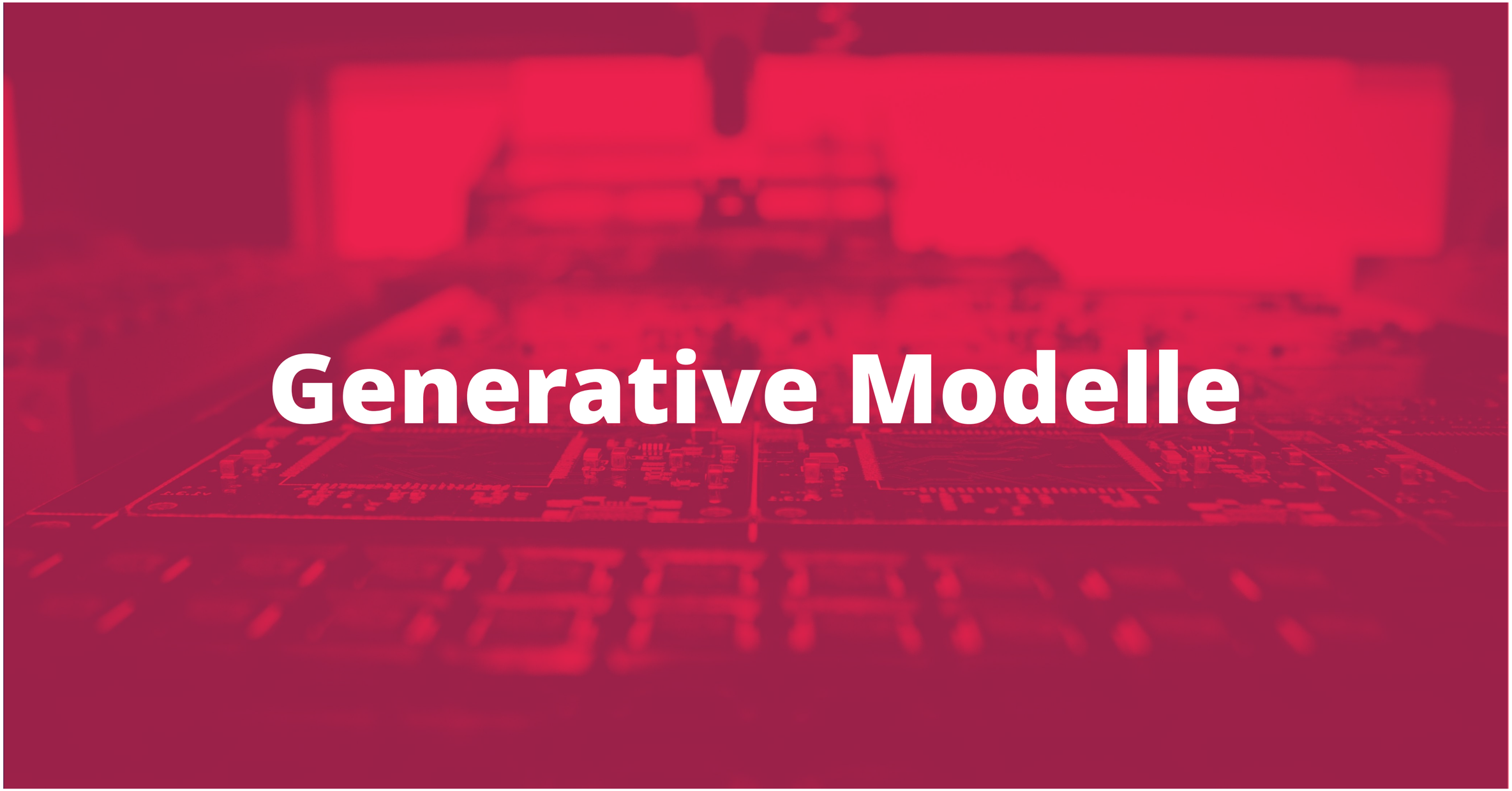
- (Noch) bessere Abbildung von Ähnlichkeiten
- Besseres Ranking

Wie geht das?

- Nutzung der gleichen Daten wie beim Finetuning der Embeddings
- Finetuning erfolgt ebenso mit BERT oder SBERT

Ist das nicht extrem aufwändig?

- Vorteil: Ähnlichkeitsdaten schon vorhanden
- Mit GPU leicht zu bewältigen



Generative Modelle

Grundfunktionsweise: Transformer-Modell mit Attention

GPT basiert auf einem sog. Transformer-Modell

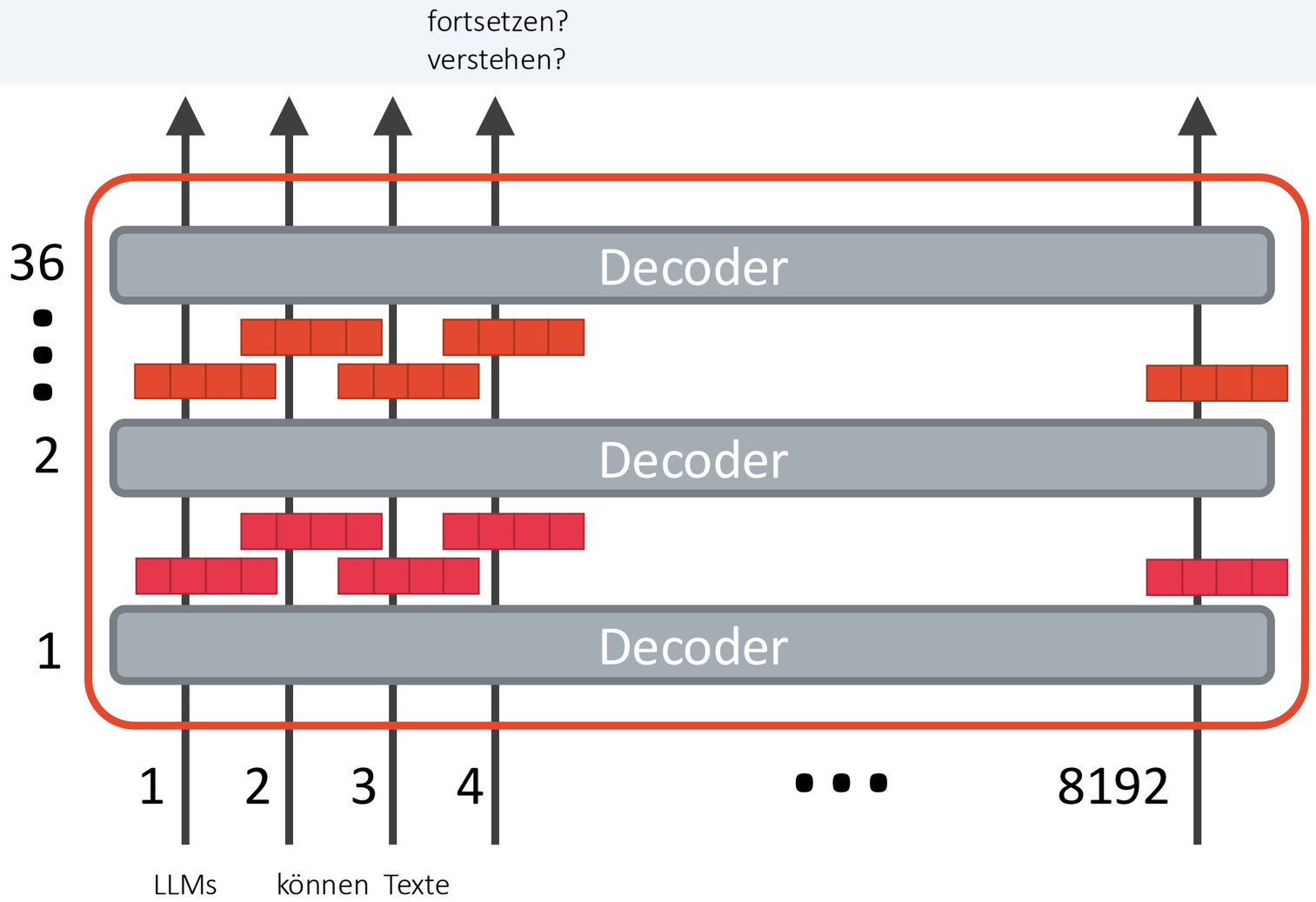
- Vorher verwendet für Übersetzungen
- Sog seq2seq-Verfahren (Encoder/Decoder)

GPT nutzt nur den Decoder

- Decoder wird kaskadiert
- Unterschiedliche Layer modellieren die Kontextualisierung (nur nach vorne)
- Training auf Vorhersage des nächsten Wortes (autoregressiv)

GPT hat sehr viele Parameter

- Ständiges Wachstum
- Training noch viel aufwändiger als bei BERT
- Kann nur mit extremem Hardware-Aufwand bewältigt werden



Wachst

GPT-4

Wachstum der Sprachmodelle

Unendliches Wachstum?

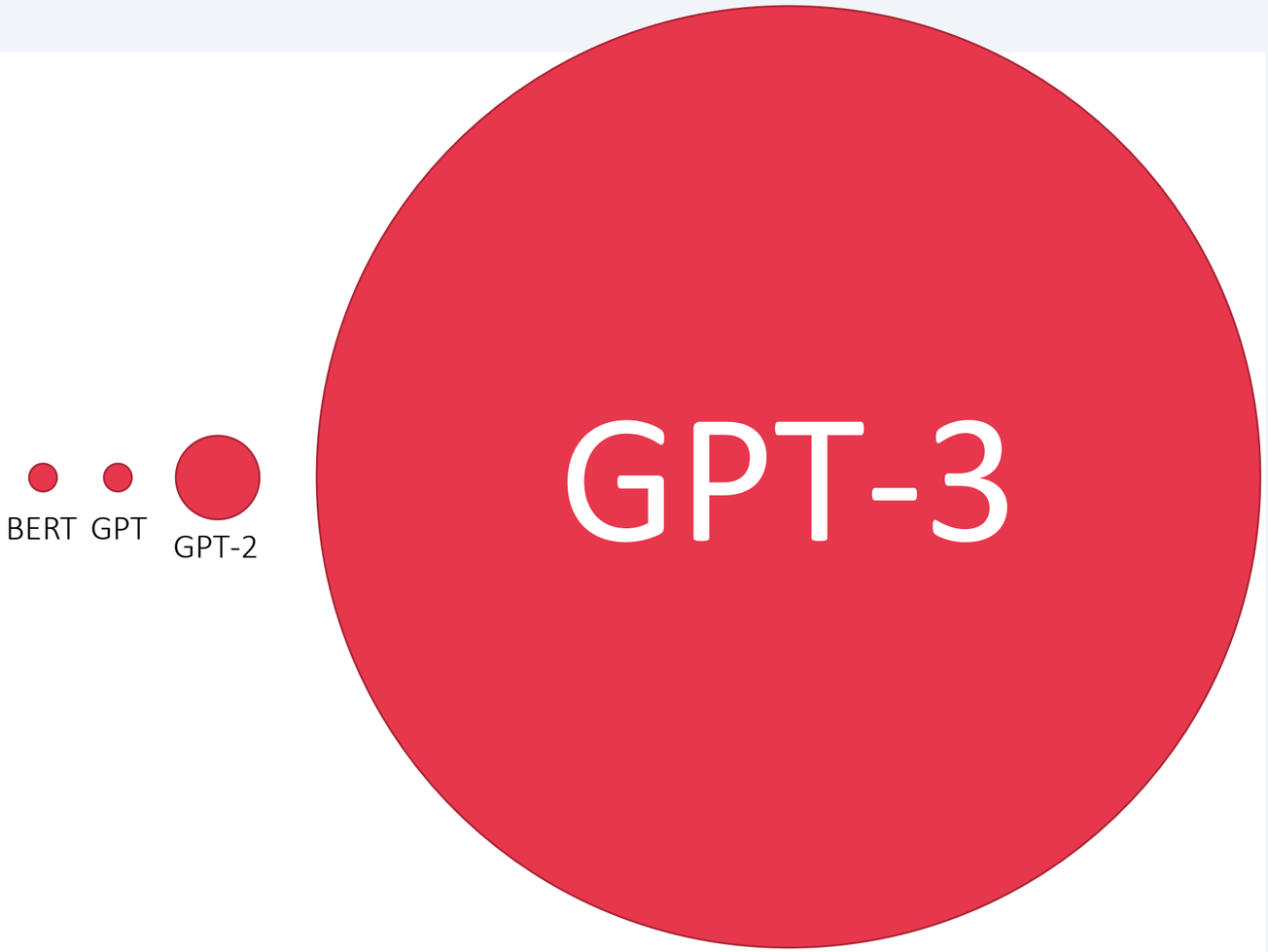
- Rechenkapazität wächst immer weiter und schneller
- Trainingsmenge (Text) in nahezu beliebiger Menge verfügbar (aber vielleicht irgendwann auch ausgeschöpft)
- Geld offenbar gar kein Problem

Aufgaben werden immer schwieriger

- Von Sentiment-Analyse zu wissenschaftlichen Arbeiten

Aber: wenig strukturell neue Ideen

- Dünn besetzte Modelle (sparse)
- Distillation (besonders bei BERT, jetzt auch bei GPT)



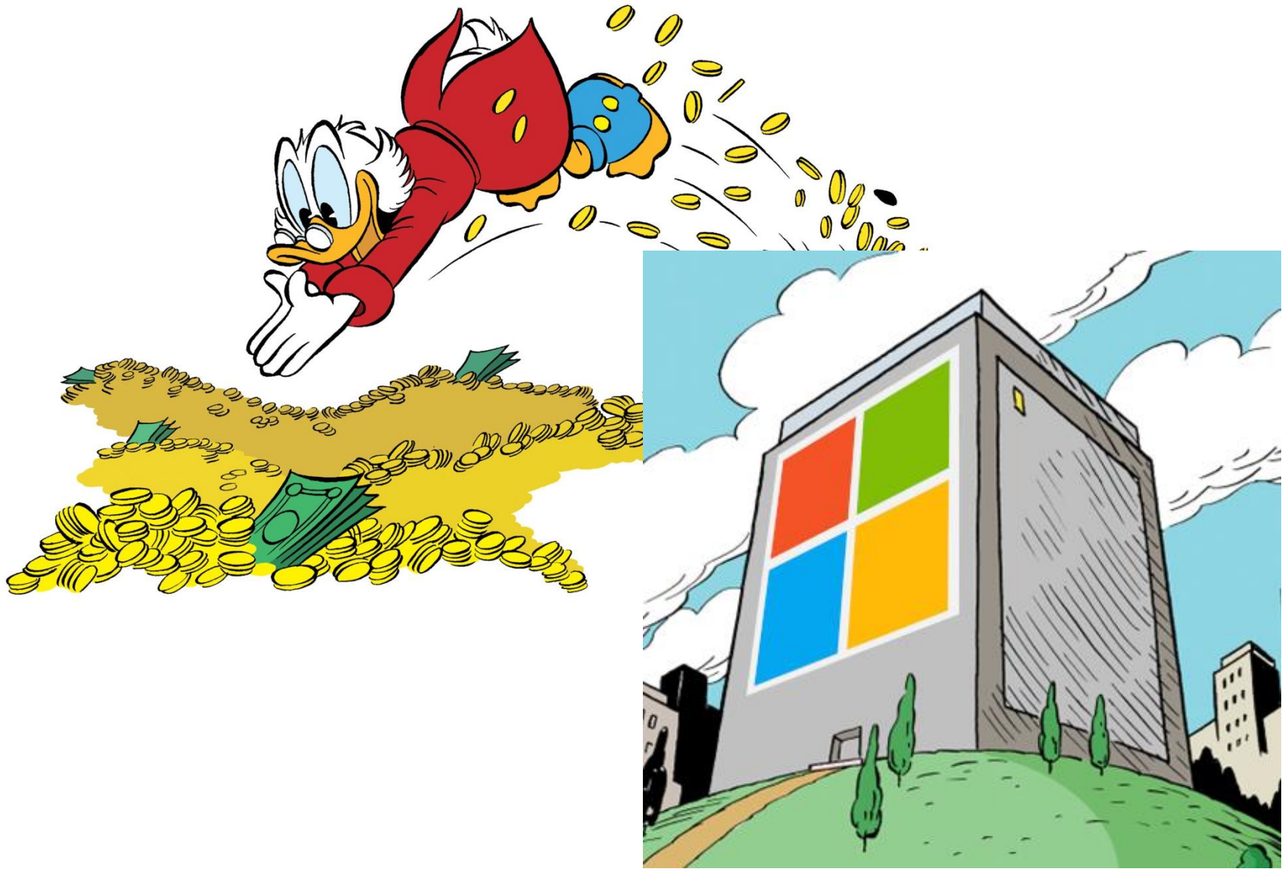
Welche Rechenleistung wird für ChatGPT benötigt und was kostet das?

Training:

- Sehr, sehr aufwändig
- Billionen von Token
- Kosten für GPT-4 angeblich über 100 Millionen Dollar[1]

Betrieb:

- Auch dafür werden Grafikkarten benötigt
- Bei ChatGPT wohl mehr als 700.000 Dollar pro Tag[2]
- Überwachung durch Menschen (mit minimalem Lohn!) notwendig[3]



Quelle: https://www.businessinsider.de/panorama/dagobert-ducks-geldspeicher-verlag-ueberrascht-mit-neubauplaenen-und-spricht-in-ausschreibung-gezielt-studierende-an/

[1] https://www.wired.com/story/openai-ceo-sam-altman-the-age-of-giant-ai-models-is-already-over/
[2] https://www.businessinsider.com/how-much-chatgpt-costs-openai-to-run-estimate-report-2023-4
[3] https://www.nbcnews.com/tech/innovation/openai-chatgpt-ai-jobs-contractors-talk-shadow-workforce-powers-rcna81892

Instruction Following

Basis-Training

- LLMs können Wörter vorhersagen
- Sehr gut zur Fortsetzung von Texten
- Enorm aufwändiges Basis-Training

Beantworten von Fragen?

- Finetuning
- Super-Struktur hinzufügen (Templates)
- Oftmals kombiniert mit Reinforcement Learning (RLHF mit Reward-Funktion)
- Bestimmte Antworten werden bevorzugt

```
<|system|>
You are a helpful travel assistant.<|end|>
<|user|>
I am going to Paris, what should I see?<|end|>
<|assistant|>
Paris, the capital of France, is known for its stunning
architecture, art museums, historical landmarks, and romantic
atmosphere. Here are some of the top attractions to see in
Paris:\n\n1. The Eiffel Tower: The iconic Eiffel Tower is one
of the most recognizable landmarks in the world and offers
breathtaking views of the city.\n2. The Louvre Museum: The
Louvre is one of the world's largest and most famous museums,
housing an impressive collection of art and artifacts,
including the Mona Lisa.\n3. Notre-Dame Cathedral: This
beautiful cathedral is one of the most famous landmarks in
Paris and is known for its Gothic architecture and stunning
stained glass windows.\n\nThese are just a few of the many
attractions that Paris has to offer. With so much to see and
do, it's no wonder that Paris is one of the most popular
tourist destinations in the world."<|end|>
```

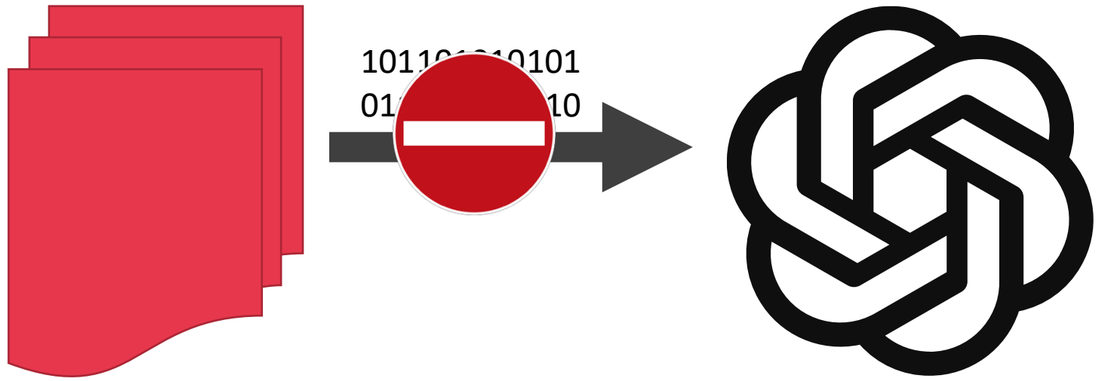
Läuft das auch auf *meiner* Hardware?

Kommt auf deine Hardware an…

- Evtl. sehr langsam auf einer CPU mit genügend RAM
- Akzeptable Performance erfordert mehrere A100 mit 80 GB

Warum es nicht geht:

- Modell ist nicht veröffentlicht
- Alle Daten werden nur bei OpenAI verarbeitet (Datenhoheit!)



CH Erkläre Heise Zeitschriftenverlag!

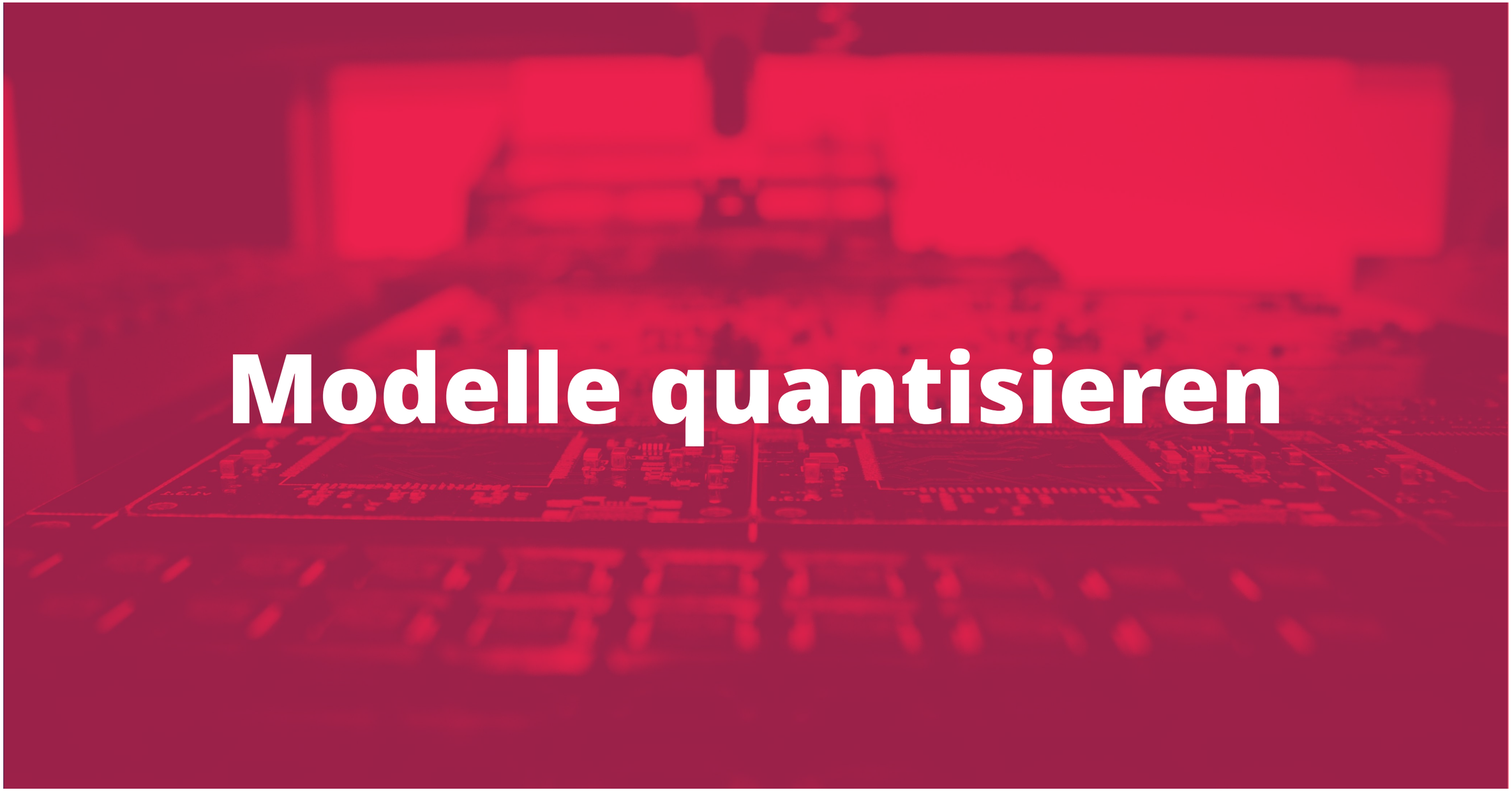
Der Heise Zeitschriftenverlag ist ein deutsches Verlagsunternehmen mit Sitz in Hannover, das sich auf die Publikation von IT- und Telekommunikationszeitschriften sowie Online-Portalen spezialisiert hat.

Gegründet wurde der Verlag 1949 von August Heise als "Heise RegioConcept". Seit den 1980er Jahren liegt der Schwerpunkt des Verlags auf IT- und Telekommunikationsthemen. Bekannte Zeitschriften aus dem Verlag sind beispielsweise die c't (Computerzeitschrift), iX (Magazin für professionelle Informationstechnik), Technology Review und Telepolis.

Neben den Print-Publikationen betreibt der Heise Verlag auch zahlreiche Online-Portale, darunter heise online, das zu den meistbesuchten deutschsprachigen IT-News-Portalen zählt.

Der Heise Zeitschriftenverlag ist heute Teil der Heise Gruppe, die auch IT-Veranstaltungen wie die CeBIT und die SecIT sowie den Verlag dpunkt.verlag und die Jobbörse heise jobs umfasst.

Interaktiv / Jupyter



Modelle quantisieren

Präzision von Fließkommazahlen

Bisher häufig verwendet

- Double precision: 64 bit → 8 bytes
- Single precision: 32 bit → 4 bytes

Besonders häufig bei LLMs

- 16 bit → 2 bytes (bfloat16)
- 8 bit bleiben für Exponent, aber nur 7 bit für Mantisse
- Präzision wird zugunsten des Wertebereichs geopfert

Für ein 70B Modell braucht man 140 GB RAM

- Kann man das weiter reduzieren?
- Präzision auf 8 bit reduzieren (int8) mit `bitsandbytes`
- Funktioniert, aber muss jedesmal berechnet werden (Speicher!)
- LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale, 2022

Type	Bits			
	Sign	Exponent	Significand	Total
Half (IEEE 754-2008)	1	5	10	16
Single	1	8	23	32
Double	1	11	52	64
x86 extended precision	1	15	64	80
Quad	1	15	112	128

Source: https://en.wikipedia.org/wiki/Floating-point_arithmetic

Platz sparen mit GPTQ

Kann man den RAM-Bedarf weiter reduzieren?

- Ja, durch weitere Reduzierung der Genauigkeit (z.B. zu int4)
- Funktioniert sehr gut für die *Evaluation* (aber nicht für das Training)
- Veröffentlichung in 2023:
  GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers
- Algorithmus ist viel schneller als alles Andere zuvor

Existierende Implentierungen zur Modell-Evaluation

- In Hugging Face Transformer integriert:
  https://huggingface.co/blog/gptq-integration
- Auch als separate Software nutzbar: llama.cpp

Bessere Quantisierung

- Idee: Dynamisch für unterschiedliche Layer
- AWQ: Activation-aware quantization
  - Viele Modelle bereits verfügbar
  - Braucht viel VRAM (Layer umschreiben)
- ExLlamaV2: mittlere Anzahl von Bits
  - Quantisierung muss oft selbst durchgeführt warden
  - Benötigt Datenset zum Abgleich
  - Sehr schnelle Inferenz mit geringem RAM-Bedarf
- llama.cpp: Ausführung auf CPU (gguf-Format)

Fertige Software für Inferenz

- TGI
- vLLM
- SGLang
- tabbyAPI

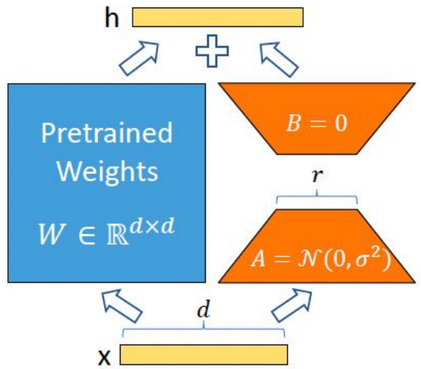
Interaktiv / Jupyter

Feintuning von LLMs

Werkzeugkasten

Grundsätzliche Problematik

- Modelle haben sehr (zu) viele Parameter
- Training, aber auch Finetuning sehr aufwändig
- Ziel: Anzahl der Parameter reduzieren



LoRA: Low Rank Adaptation

- Reduktion der trainierbaren Parameter
- “Einfrieren” de Originalgewichte
- Darstellung der (veränderten) Gewichte durch ein Produkt zweier niedrigdimensionaler Matrizen
- Sehr effizient und oft besser als “vollständiges” Feintuning

PEFT – “Parameter Efficient Tuning”

- Bibliothek von Hugging Face
- Praktisch überall verwendet
- Ermöglicht Feintuning auf “Consumer Hardware” (A100)

QLoRA

- Feintuning eines bereits quantisierten Modells
- Weitere Optimierung mit LoRA
- Deutlich verringerter Speicherbedarf

Deep Speed

deepspeed

- Optimierung für Instruction Following (Reinforcement Learning with Human Feedback, RLHF)
- Sehr hohe Beschleunigung in der Lernphase

Accelerate

- Verteilte Ausführung

Modelle, Datensätze und Trainingssoftware

Beispiele mit kleinen Modellen

- Qwen3-0.6B
- Lllama-3.2-1B
- SmolLM von Hugging Face
- Phi-3.5

Datensätze

- Häufig Instruction Following
- Aber nicht nur, auch Fortsetzung
- Kommt auf den Anwendungsfall an
- Unterschied ist nur in der Konstruktion der Texte zum Training

Umgebung

- Nicht immer ganz einfach mit Docker
- Spezielle Anpassungen für CUDA notwendig

Selbst schreiben

- Sehr viele Optionen, teilweise auch modellspezifisch
- Gibt die besten Einblicke

unsloth

- Besonders effizient, unterstützt QLoRA
- Jupyter-Notebooks verfügbar

axolotl

- Viele Modelle unterstützt und wird ständig erweitert
- Notebook oder CLI
- Nicht ganz einfach zu installieren (Enironment verwenden!)

Interaktiv / Jupyter

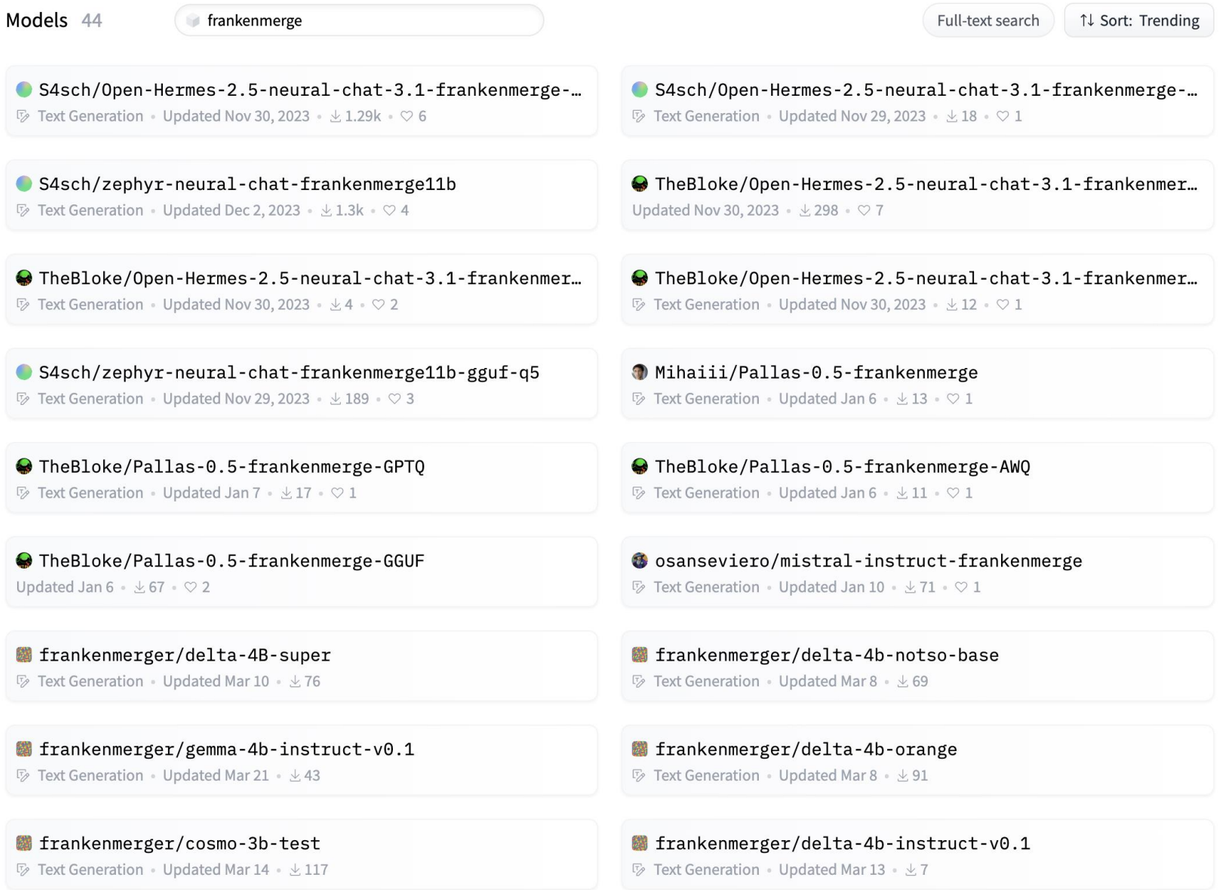
Eigene Modelle ohne Training?

Training ist aufwändig

- Trainingsdaten werden benötigt
- Braucht GPUs
- Braucht Rechenzeit

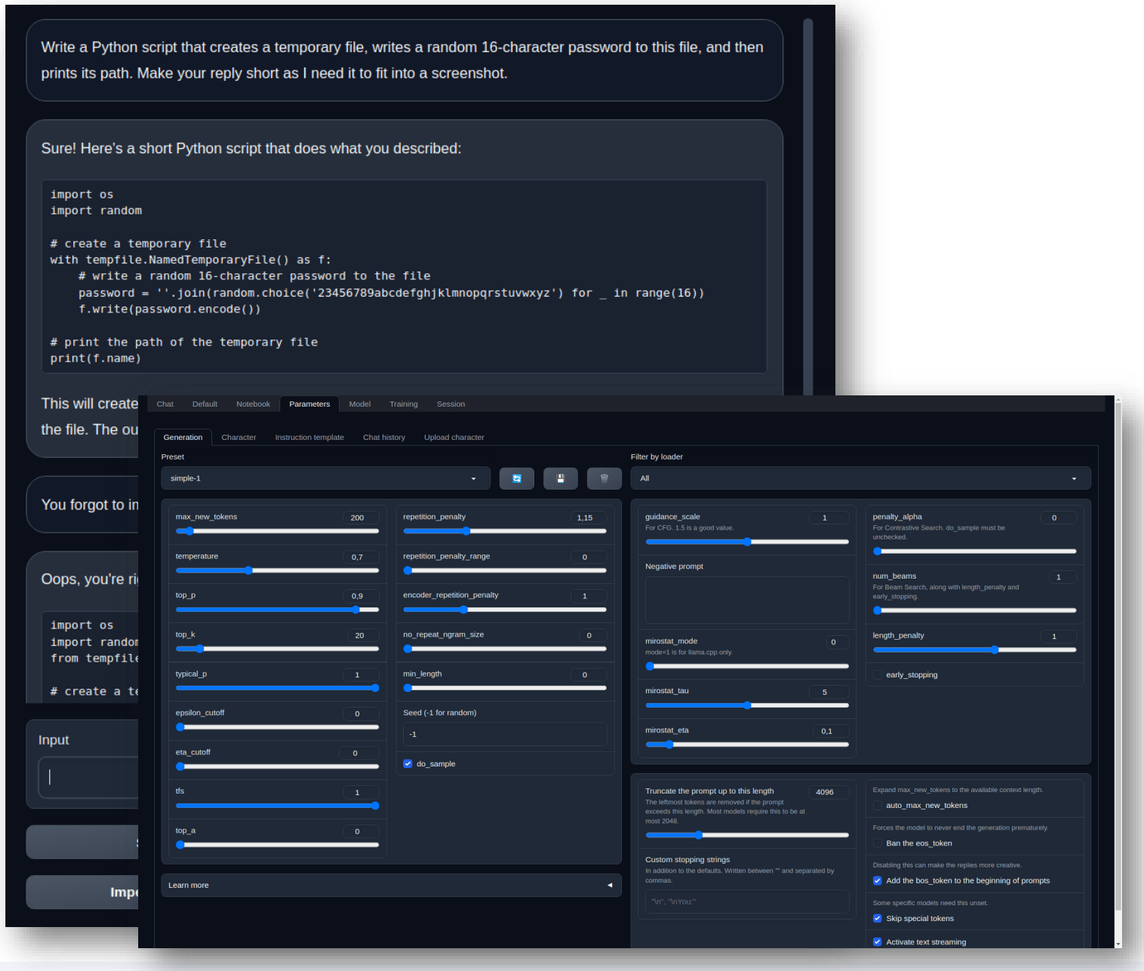
Alternative

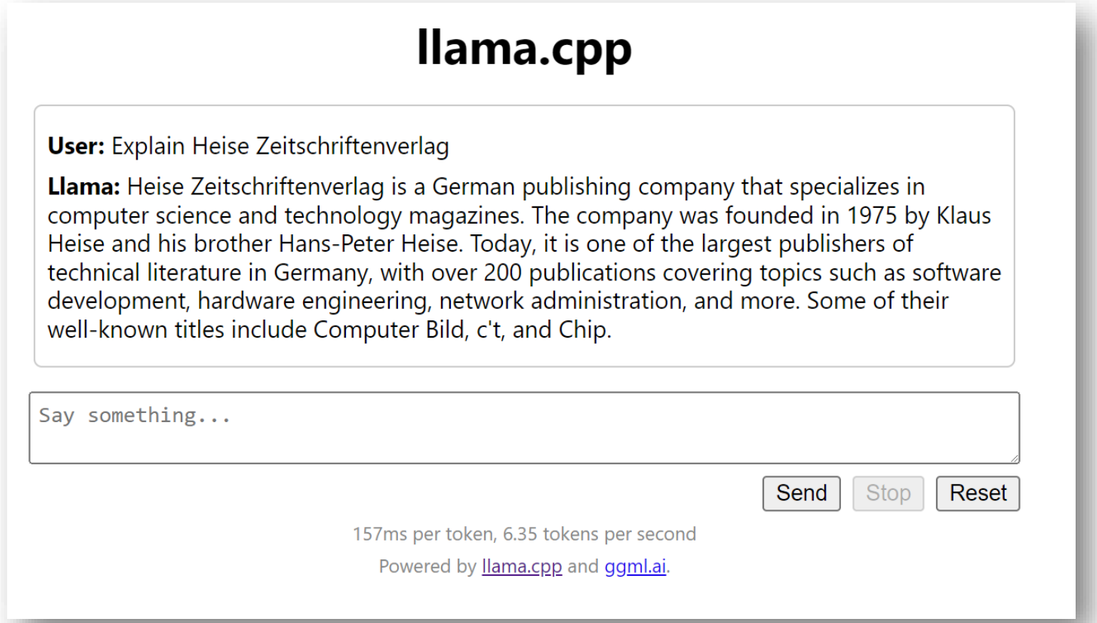
- Nutzung existierender Modelle und LoRas
- “Verschmelzung” der Modelle
    - Funktioniert überraschenderweise verhältnismäßig gut
    - “Frankenmerge”
    - Nicht mehr ganz so populär



Frontends für LLMs

Die Qual der Wahl…





Systematischer Überblick

	llama.cpp	koboldcpp	Text generation web UI	ollama	Open WebUI	LM Studio
Frontend	CLI, simple web	sophisticated web	Sophisticated web	CLI	sophisticated web	Own
Backend	llama.cpp	llama.cpp	various	llama.cpp	ollama, OpenAI API	llama.cpp, mlx
API	Python, OpenAI	OpenAI	OpenAI	Ollama, OpenAI	own	OpenAI
Models	unmanaged	unmanaged	unmanaged	managed	ollama, OpenAI API	managed

Notable mention: mistral.rs (CLI), Silly Tavern (Roleplay), tabbyAPI (Exllamav2)

Zusammenfassung und Ausblick

Warum so spannend?

Mehr und mehr Basismodelle stehen zur Verfügung

- Rege Entwicklung mit offenen Daten etc.
- Google: Open Source-Modelle werden den kommerziellen den Rang ablaufen[1]
- Domänenspezifisches Finetuning

MISTRAL AI_

Business Cases

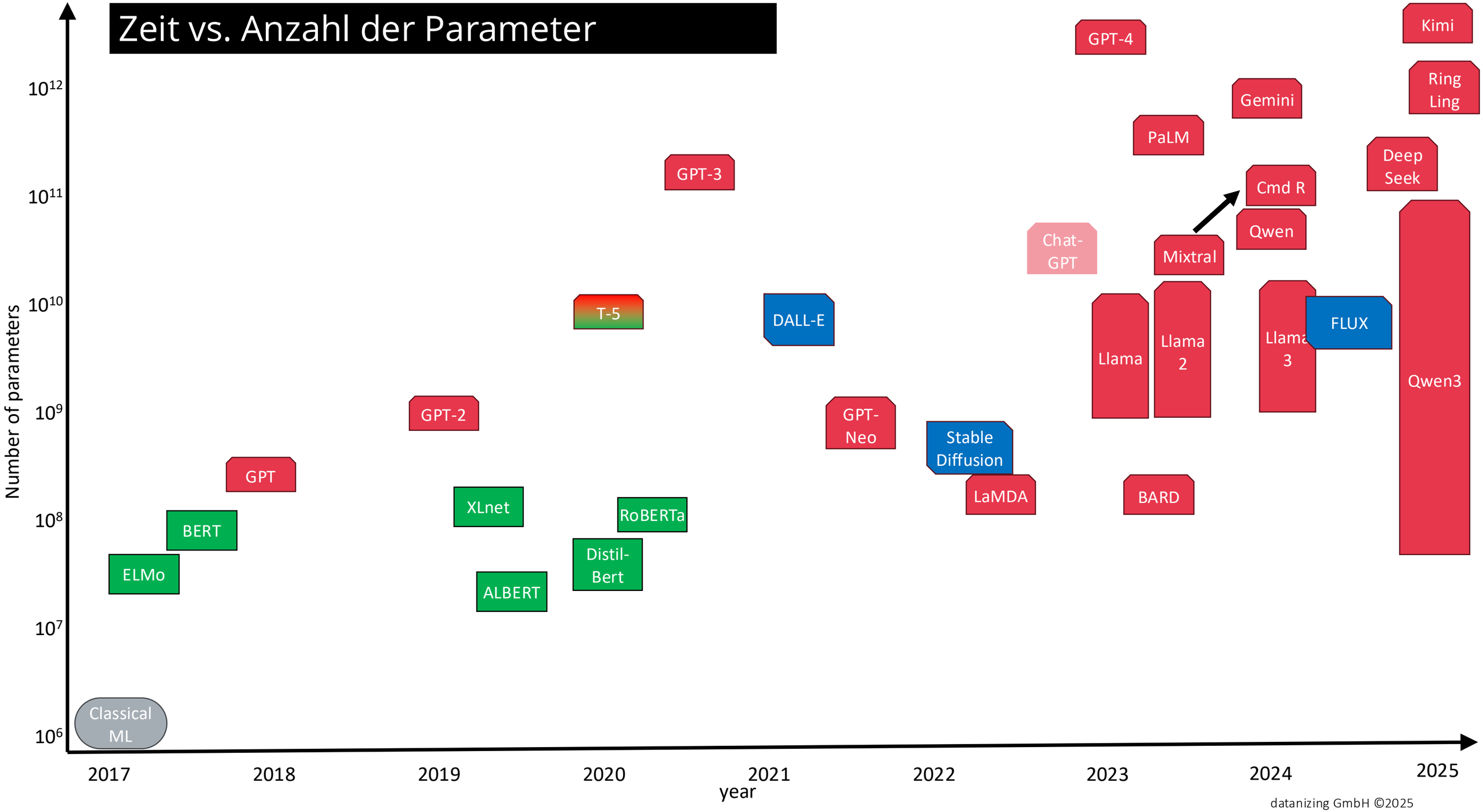
- Bestimmte Berufsbilder nicht mehr existent: Copywriter
- Viele Ideen fehen noch

Kombination der Verfahren

- Retrieval Augmented Generation
- Effizient und vermeidet viele Probleme
- Neue Form des Information Retrievals

[1] https://www.heise.de/news/Anonymer-Google-Entwickler-Open-Source-wird-Google-und-OpenAI-den-Rang-ablaufen-8989179.html

Zeit vs. Anzahl der Parameter



Aktuelle Entwicklungen

RAG

- Retrieval Augmented Generation

Spezialmodelle

- CodeLlama, Llemma, geometische Modelle etc.

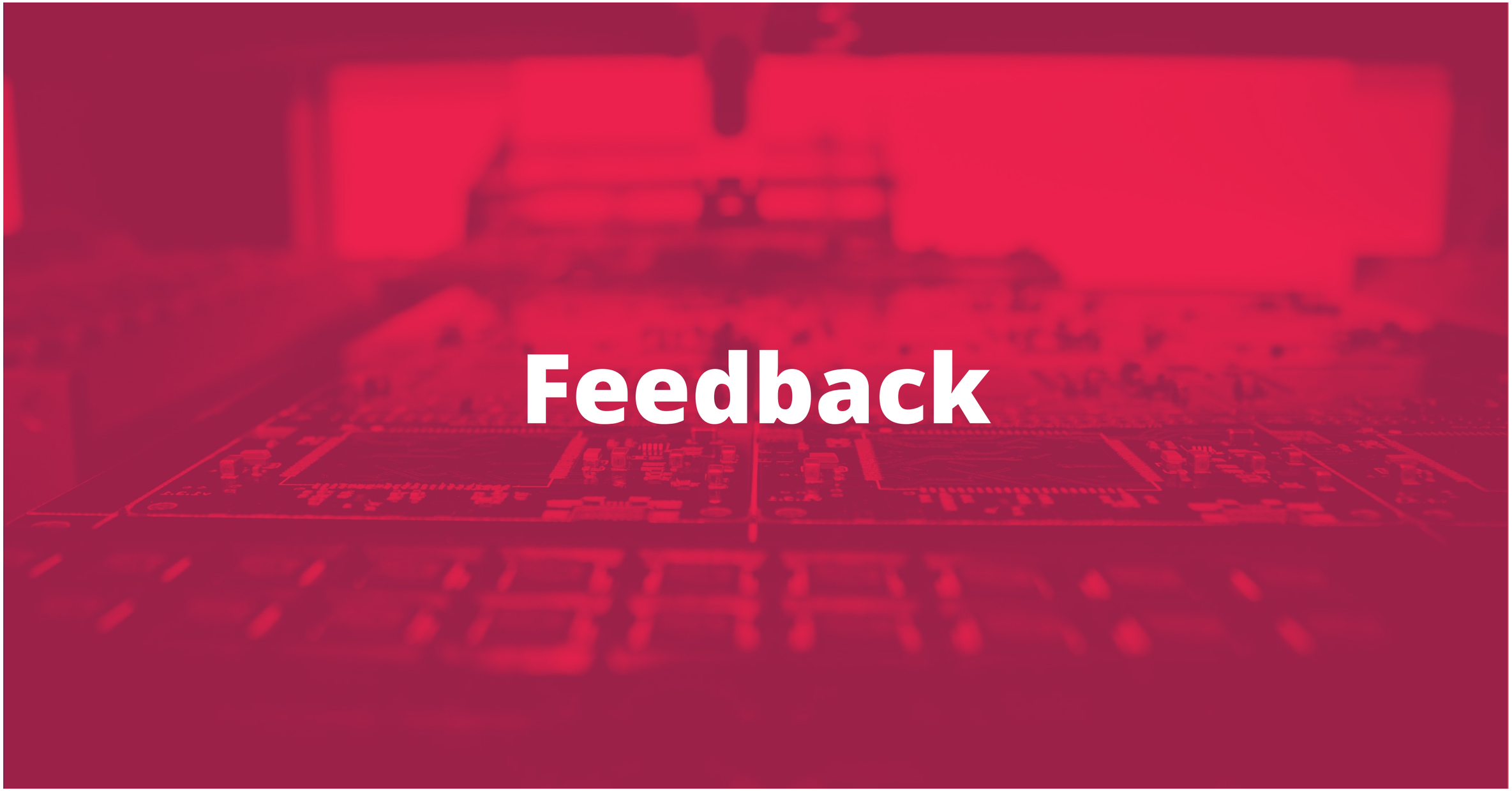
Performance von Open Weight-Modellen

- Mistral, Mixtral, Qwen, DeepSeek, Kimi K2,

Neue Methoden

- HQQ (Half quadratic quantization)
- Groq, Cerebras: Extrem schnelle Inferenz
- Pures Reinforcement Learning

Quelle: https://www.linkedin.com/posts/maxime-labonne_arena-elo-graph-updated-with-new-models-activity-7187062633735368705-u2jB



Feedback