Einführung in Large Language Models und Prompt Engineering

Zentrale Fragen für diesen Vortrag

1. Was sind Large Language Models?
2. Wie können wir Large Language Models benutzen?
3. Wie können wir Prompt Engineering einsetzen, um bessere Ergebnisse zu erzielen?

Was sind Large Language Models?

Ein **Large Language Model** (LLM) ist eine Art von **künstlicher Intelligenz** (KI), die Text verarbeiten und erzeugen kann.

LLMs sind eine noch relativ **junge und wenig erforschte**[1] Technologie, haben aber das Potenzial, unseren Umgang mit Computern **grundlegend zu verändern**.

[1] GPT-2 wurde 2019 veröffentlicht,

Wie funktionieren Large Language Models?

LLMs werden mit einer Technik namens **Deep Learning** trainiert.

Deep Learning ist eine Art des maschinellen Lernens, die künstliche neuronale Netze verwendet, um aus Daten zu lernen.

Im Fall von LLMs werden die neuronalen Netze mit riesigen Datensätzen bestehend aus Text und Code trainiert.

Die neuronalen Netze lernen dadurch die Beziehungen zwischen **Wörtern und Phrasen**. Dadurch können sie das "richtige" **nächste Wort** in einem Satz vorhersagen.

Wie benutzt man Large Language Models?

LLMs sind vor allem ein **Werkzeug**, mit dem man Text generieren kann.

Die Art und die Qualität des Texts hängt von der **Knowledge Base** des Models ab.

Kategorien von Aufgaben, die LLMs aktuell lösen können:

- Brainstorming
- Chatbots (z.B. ChatGPT)
- Codegenerierung (z.B. Github Copilot)
- Maschinelle Übersetzung (z.B. DeepL Translate)
- Textzusammenfassung

Beispiele aus unserem Arbeitsalltag:

— Battle Cards und One Pager
— Outlines von Präsentationen
— Pair Programming
— Produktideen sammeln
— `tl;dr` von technischen Artikeln

Was sind Grenzen von Large Language Models?

Spoiler: LLMs sind nicht perfekt und können Fehler machen.

LLMs schaffen **keine Fakten**, sie generieren Text.

LLMs neigen dazu, zu "**halluzinieren**", d.h. Informationen zu erfinden, die plausibel klingen, aber nicht auf Fakten beruhen.

"If hallucination is a problem, you're using the wrong tool."

LLMs haben einen **Stichtag**, der ihre aktuellsten Trainingsinformationen widerspiegelt:

— GPT-3.5: September 2021

— Github Copilot: Oktober 2021[2]

— PaLM 2: Februar 2023

— GPT-4: April 2023

[2] Copilot wird regelmäßig für "wichtige" Repositories aktualisiert.

LLMs haben eine starke **Zufallskomponente**: Selbst wenn man die gleichen Prompts benutzt, ist das Ergebnis sehr wahrscheinlich ein anderes.

Das unterscheidet sich von Programmiersprachen, die in der Regel ein deterministisches Ergebnis liefern.

Das Training von LLMs ist sehr rechen- und **ressourcenintensiv**.[3]

Daher werden LLMs auf spezialisierter Hardware (z.B. GPUs oder TPUs) trainiert, die vor allem Hyperscalern zur Verfügung stehen.

[3] 'Thirsty' AI: Training ChatGPT Required Enough Water to Fill a Nuclear Reactor's Cooling Tower, "500 ml Wasser pro 20 Fragen"

Wie können wir Prompt Engineering einsetzen, um bessere Ergebnisse zu erzielen?

Was ist Prompt Engineering?

Was sind Prompts? 😅

Prompts sind die "**Befehle**", die man an ein LLM schickt.

Zum Beispiel:

— "Fasse folgenden Text zusammen:"
— "Wer ist die Bundeskanzlerin von Deutschland?"
— "Erstelle eine Liste von Geschenkideen."

Prompt Engineering ist der Versuch, diese Befehle durch Ausprobieren und Erfahrung so zu schreiben, dass sie **bessere Ergebnisse** liefern.

Dabei kommt "besser" ganz auf die Situation an.

Best Practices für Prompt Engineering

Starte mit einem klaren Ziel. Was soll das LLM tun?

Sei genau und eindeutig.

Je weniger Interpretationsspielraum das Prompt lässt, umso geringer die Wahrscheinlichkeit, dass das LLM den falschen Weg einschlägt.

Verwende die "richtige" Sprache.[4]

LLMs wurden auf einer Vielzahl von Texten trainiert. Wenn das Ergebnis ein Fachtext sein soll, verwende Fachbegriffe. Wenn das Ergebnis allgemein verständlich sein soll, benutze "Alltagssprache".

[4] Im Moment bedeutet das leider auch, die Prompts besser in Englisch zu formulieren und das Ergebnis zu übersetzen. 🇺🇸

Verwende den richtigen Tonfall.

Der Ton eines Prompts kann die Ausgabe des LLMs beeinflussen. Ist das Prompt formell geschrieben, wird das LLM wahrscheinlich eine formellere Antwort generieren.

Experimentiere mit verschiedenen Prompts.

Die besten Prompts für eine bestimmte Aufgabe hängen von der Art des LLMs und der jeweiligen Aufgabe ab.

Erzähle dem LLM, was wichtig für die Aufgabe ist.

LLMs ermöglichen die Einführung neuer Fakten, z. B. zu Ereignissen, die nach dem Stichtag stattgefunden haben, oder proprietäre Informationen, die nicht Teil des ursprünglichen Trainingsdatensatzes waren.

Einige nützliche Prompt-Patterns

Prompt-Patterns sind **wiederkehrende Muster** in Prompts, die sich mit der Zeit als **besonders effektiv** herausgestellt haben.

Actor-Pattern

```
Acting as a $ROLE, do $TASK.
```

Audience-Pattern

```
Assuming you are talking to $AUDIENCE, do $TASK.
```

Chain-of-thought-Pattern

```
Do $TASK and explain your reasoning step by step.
```

Normalerweise als Teil eines Prompts, bei dem Fact Checking wichtig ist.

Wait-for-Input-Pattern

```
Do $TASK. Wait for my input before answering.
```

Normalerweise in Kombination mit Pattern, die längere Prompts wie Beispiele oder zusätzlichen Kontext erfordern.

Template-Pattern

```
Format your output using the following pattern:

---
Q: <<QUESTION>>
A: <<ANSWER>>
---
```

List-Pattern

```
Create a list based on $TASK, each item formatted as ...
```

Normalerweise kombiniert mit dem Template-Pattern.

JSON-Output-Pattern

```
Return your answer formatted as JSON, with the following fields ...
```

Funktioniert auch mit anderen Formaten, wie z.B. Markdown oder YAML.

Weitere Patterns findet ihr hier:

A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT

Beispiel-Prompts

Es folgen ein paar Beispiel-Prompts aus einem Projekt, mit dem ich Battle Cards und One Pager für eins unserer Produkte (OEDIV Cloud Data Lake) erstellt habe.

Context

Act as a technical expert for the cloud data lake project working at OEDIV.

You are responsible for the data lake architecture and design.

Knowledge Base

My next prompt contains a description of the cloud data lake service.

Summarize the description **in your own words**.

Wait for the prompt to appear before typing your answer.

Service-Beschreibung

Create a list that describes a service offering based on the cloud data lake.

The list should include the following items: What does it do. What are the benefits. What are unique selling points.

Zentrale Fragen für diesen Vortrag … beantwortet?

✅ Was sind Large Language Models?
✅ Wie können wir Large Language Models benutzen?
✅ Wie können wir Prompt Engineering einsetzen?

Q & A