M17 - Modellauswahl und Evaluation

Stand: 02.2025

1 Modellübersicht (Ausschnitt)

Tier	Modelle	Eigenschaften	Anwendungsfälle
Tier 3	- **GPT-4o mini** - Gemini 1.5 Flash - Gemini 2.0 Flash-Lite - OpenChat 3.5 - Gemma 7B - **o3-mini**	- Preiswert - Schnell - Kosteneffizient - Hohe Intelligenz für die Größe	- Zusammenfassung vieler Dateien - Analyse von Podcast-Transkripten - Schnelle Textgenerierung - Lokale Verarbeitung auf Geräten - Prototyping und Experimente
Tier 2	- GPT-4 - GPT-4 Turbo - Claude 3.5 Sonnet - Gemini 1.5 Pro - Gemini 2.0 Pro - o3-mini-high	- Gutes Preis-Leistungs-Verhältnis - Vielseitig einsetzbar	- Programmierung - Funktionsaufrufe - Tool-Nutzung - E-Mail-Bearbeitung
Tier 1	- **GPT-4o** - Claude 3.5 Opus - Gemini 1.0 Ultra - o1 - DeepSeek R1	- Höchste Intelligenz - Langsamer - Teurer - Fortschrittliche Reasoning-Fähigkeiten	- Tiefes Verständnis - Kontextbezogenes Denken - Generierung von Anwendungsfällen - Komplexe Reasoning-Aufgaben

Tier 3: Die Arbeitspferde

Tier 3-Modelle sind die kostengünstigen und schnellen "Arbeitspferde" unter den LLMs. Sie bieten ein ausgewogenes Verhältnis zwischen Leistung und Effizienz:

- **Vorteile**: Preiswert, schnell, ressourceneffizient
- **Anwendungen**: Ideal für Aufgaben, die hohe Geschwindigkeit und niedrige Kosten erfordern, wie Datenzusammenfassungen oder Transkriptanalysen
- **Beispiele**: GPT-4o mini, Gemini 1.5 Flash, o3-mini

Tier 2: Die Allrounder

Tier 2-Modelle stellen die mittlere Stufe dar und bieten ein gutes Gleichgewicht zwischen Intelligenz und Preis:

- **Vorteile**: Vielseitig einsetzbar, gutes Preis-Leistungs-Verhältnis
- **Anwendungen**: Geeignet für ein breites Spektrum von Aufgaben wie Programmierung, Funktionsaufrufe und E-Mail-Bearbeitung
- **Beispiele**: GPT-4, Claude 3.5 Sonnet, Gemini 1.5 Pro

Tier 1: Die Spitzenreiter

Tier 1-Modelle repräsentieren die Spitze der KI-Technologie mit höchster Intelligenz und fortschrittlichen Fähigkeiten:

- **Vorteile**: Höchste Intelligenz, fortschrittliche Reasoning-Fähigkeiten
- **Anwendungen**: Ideal für komplexe Aufgaben, die tiefes Verständnis und kontextbezogenes Denken erfordern
- **Beispiele**: GPT-4o, Claude 3.5 Opus, o1, DeepSeek R1

Diese Einteilung hilft Nutzern, das am besten geeignete Modell für ihre spezifischen Anforderungen auszuwählen, basierend auf der benötigten Leistung, Geschwindigkeit und den Kosten.

> **Tip #1 Modellauswahl**
>
> Wählen Sie das richtige LLM-Modell basierend auf Ihren spezifischen Anforderungen. Für schnelle und kosteneffiziente Aufgaben wie Datenzusammenfassungen sind Tier 3-Modelle ideal. Für vielseitige Anwendungen wie Programmierung und E-Mail-Bearbeitung eignen sich Tier 2-Modelle. Für komplexe Aufgaben, die tiefes Verständnis erfordern, sollten Sie auf Tier 1-Modelle zurückgreifen. Dies hilft, die Effizienz zu maximieren und die Kosten zu minimieren.

Es macht absolut Sinn, in verschiedenen Phasen eines KI-Projekts unterschiedliche Tools oder Modelle einzusetzen. Dies wird oft als "KI-Orchestrierung" oder "Model Cascade" bezeichnet und kann die Stärken verschiedener Systeme optimal nutzen.

2 Modellkaskade

Eine **Modellkaskade** bezeichnet eine Strategie in der GenAI, bei der mehrere Modelle nacheinander ausgeführt werden. Damit können die Stärken unterschiedlicher Modelle korrespondierend zu den Herausforderungen der Aufgabenstellung zur Lösung eingebracht werden.

Beispielhaft:

Problemdefinition und Recherche

- **Tools**: Perplexity AI, Elicit, Consensus
- **Vorteile**: Diese Tools sind auf Informationsrecherche und -zusammenfassung spezialisiert und können aktuelle Quellen einbeziehen

Datenanalyse und Exploration

- **Tools**: Claude Opus/Sonnet, GPT-4o
- **Vorteile**: Stärkere Reasoning-Fähigkeiten für komplexe Analysen und Mustererkennung

Lösungsentwicklung und Prototyping

- **Tools**: Langchain, LlamaIndex für den Aufbau von Workflows
- **Code-Generierung**: GitHub Copilot, Claude für komplexe Code-Generierung

Implementierung und Optimierung

- **Spezialisierte Modelle**: Für spezifische Aufgaben wie Bild-/Audioanalyse
- **Feinabstimmung**: Kleinere, auf die Aufgabe spezialisierte Modelle

Beispiel einer Modellkaskade

```
from langchain import PromptTemplate, LLMChain
from langchain.llms import OpenAI
import requests

# Phase 1: Recherche mit Perplexity API (hypothetisch)
def research_phase(query):
    perplexity_response = requests.post(
        "https://api.perplexity.ai/search",
        json={"query": query}
    ).json()

    return perplexity_response["answer"]

# Phase 2: Analyse mit Claude
def analysis_phase(research_results):
```

```
    analysis_prompt = PromptTemplate(
        input_variables=["research"],
        template="Analysiere folgende Rechercheergebnisse und identifiziere
die Hauptprobleme:\n{research}"
    )

    analysis_chain = LLMChain(
        llm=OpenAI(model="claude-3-opus-20240229"),
        prompt=analysis_prompt
    )

    return analysis_chain.run(research=research_results)

# Phase 3: Lösungsentwicklung mit spezialisiertem Modell
def solution_phase(problem_analysis):
    solution_prompt = PromptTemplate(
        input_variables=["analysis"],
        template="Entwickle konkrete Lösungsansätze für diese
Probleme:\n{analysis}"
    )

    solution_chain = LLMChain(
        llm=OpenAI(model="gpt-4o-mini"),
        prompt=solution_prompt
    )

    return solution_chain.run(analysis=problem_analysis)

# Vollständiger Workflow
def ai_problem_solving_pipeline(query):
    research_results = research_phase(query)
    analysis_results = analysis_phase(research_results)
    solution = solution_phase(analysis_results)

    return {
        "research": research_results,
        "analysis": analysis_results,
        "solution": solution
    }
```

Vorteile dieses Ansatzes:

1. **Kosteneffizienz**: Teure, leistungsstarke Modelle nur dort einsetzen, wo sie wirklich benötigt werden
2. **Spezialisierung**: Jedes Tool für seine Stärken nutzen
3. **Redundanzprüfung**: Mehrere Modelle können Ergebnisse validieren
4. **Skalierbarkeit**: Einfachere Anpassung an unterschiedliche Problemklassen

Diese Orchestrierung von verschiedenen KI-Werkzeugen wird mit fortschreitender Entwicklung des Feldes immer wichtiger, da sich Modelle zunehmend auf bestimmte Anwendungsfälle spezialisieren.

Tip #2 Modellkaskade

Nutze unterschiedliche KI-Tools für verschiedene Phasen des Projekts. Dies maximiert Effizienz und Qualität, da jedes Tool seine Stärken optimal einbringen kann und spart Kosten, da teure Modelle nur dort eingesetzt werden, wo sie wirklich nötig sind.