

23.04.2026 | Netzwerkfrühstück mittelstandsforum-koeln-bonn.de

Inside KI

Florian Peschke | Cloud Platform Engineer mit Fokus GenAI @ SIGNAL IDUNA

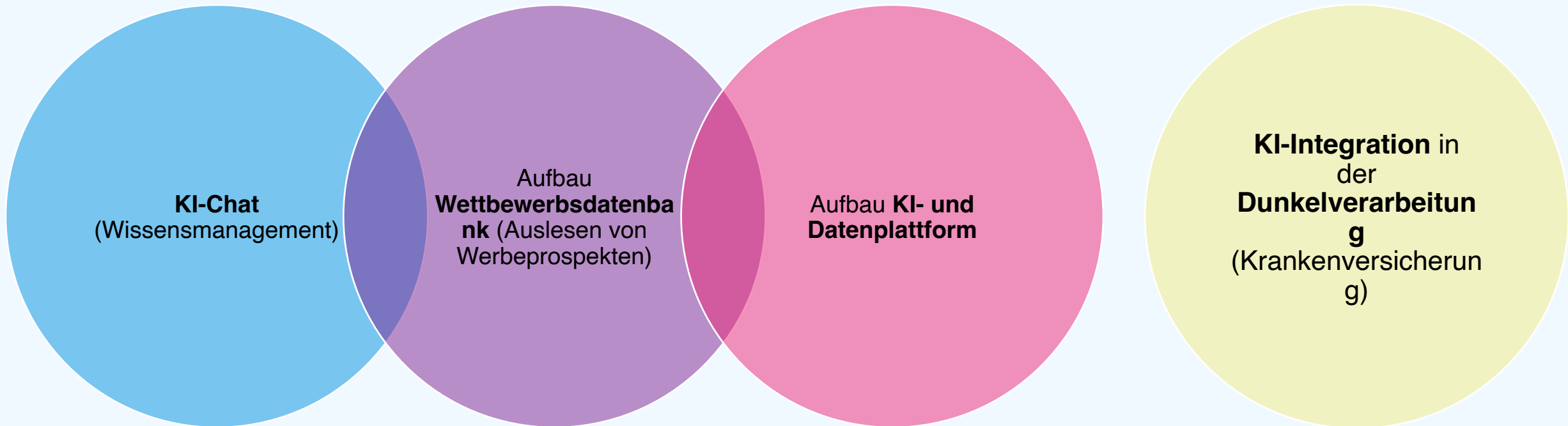
Die in dieser Präsentation geäußerten Ansichten sind ausschließlich meine eigenen und spiegeln nicht notwendigerweise die Position meines Arbeitgebers wider.

Inhalt

- 01** Was mache ich und wie arbeite ich?
- 02** Was ist KI?
- 03** Wie können Sie KI nutzen?
- 04** Meine Meinung & Einschätzung
- 05** Wie Mehrwert schaffen?

01 Was mache ich und wie arbeite ich?

Projekte aus der Praxis



Wie arbeite ich

The screenshot shows a code editor with a project structure on the left and two Python files open in the main editor. The project structure includes folders like 'benchmarks', 'github', 'venv', 'packages', 'playground', and 'src'. The 'src' folder contains subfolders like 'agents', 'ai', 'conversations', 'protocols', 'responses', 'common', 'standard', 'streaming', 'config', 'exceptions', 'infra', 'cache', 'db', 'pubsub', 'storage', 'vector', 'memory', 'observability', 'conventions', 'logging', 'monitoring', 'tracing', 'attributes', 'common.py', 'noop.py', 'opentelemetry.py', 'typing.py', and 'utils'. The 'pubsub' folder contains 'bus.py' and 'topic.py'. The 'ai' folder contains 'conversations', 'protocols', 'responses', 'common', 'standard', 'streaming', 'config', 'exceptions', 'infra', 'cache', 'db', 'pubsub', 'storage', 'vector', 'memory', 'observability', 'conventions', 'logging', 'monitoring', 'tracing', 'attributes', 'common.py', 'noop.py', 'opentelemetry.py', 'typing.py', and 'utils'. The 'responses' folder contains 'common', 'standard', and 'streaming'. The 'common' folder contains 'common.py'. The 'standard' folder contains 'standard.py'. The 'streaming' folder contains 'streaming.py'. The 'config' folder contains 'config.py'. The 'exceptions' folder contains 'exceptions.py'. The 'infra' folder contains 'infra.py'. The 'cache' folder contains 'cache.py'. The 'db' folder contains 'db.py'. The 'pubsub' folder contains 'pubsub.py'. The 'storage' folder contains 'storage.py'. The 'vector' folder contains 'vector.py'. The 'memory' folder contains 'memory.py'. The 'observability' folder contains 'observability.py'. The 'conventions' folder contains 'conventions.py'. The 'logging' folder contains 'logging.py'. The 'monitoring' folder contains 'monitoring.py'. The 'tracing' folder contains 'tracing.py'. The 'attributes' folder contains 'attributes.py'. The 'common.py' folder contains 'common.py'. The 'noop.py' folder contains 'noop.py'. The 'opentelemetry.py' folder contains 'opentelemetry.py'. The 'typing.py' folder contains 'typing.py'. The 'utils' folder contains 'utils.py'. The 'bus.py' file is open in the main editor, showing a class definition for 'ServiceBus' and several asynchronous methods. The 'topic.py' file is also open, showing class definitions for 'TextResponseTextDeltaEvent', 'TextResponseSummaryTextDeltaEvent', and 'TextResponseOutputItemDoneEvent'. The code is written in Python and uses various annotations like 'Final', 'Lock', and 'Logger'.

```
class ServiceBus[T](metaclass=SingletonMeta):
    """Service bus implementation."""

    def __init__(
        self, max_queue_size_per_topic: int = 128, max_topics: int = 1000
    ) -> None:
        self._max_queue_size_per_topic: Final[int] = max_queue_size_per_topic
        self._max_topics: Final[int] = max_topics
        self._topics: Final[dict[TopicName, Topic[T]]] = {}
        self._topics_lock: Final[Lock] = Lock()
        self._handlers_to_topics: Final[defaultdict[HandlerId, set[TopicName]]] = (
            defaultdict(set)
        )
        self._pattern_handlers: Final[defaultdict[Pattern, list[Handler[T]]]] = (
            defaultdict(list)
        )
        self._handlers_lock: Final[Lock] = Lock()
        self._logger: Final[LoggingProtocol] = get_logger("ServiceBus")

    async def subscribe(
        self, handler: Handler[T], topic_name: str
    ) -> None:
        """Subscribe a handler to a topic..."""
        validated_topic = validate_and_normalize_topic(topic_name)

        try:
            if is_pattern(validated_topic):
                await self._handle_pattern_subscription(
                    handler=handler, pattern=validated_topic
                )
            else:
                await self._handle_topic_subscription(
                    handler=handler, topic_name=validated_topic
                )
        except Exception as e:
            self._logger.error(
                msg=f"Failed to subscribe handler {handler} to {validated_topic}: {e}",
                exc_info=e,
            )
            raise

    async def _handle_pattern_subscription(
        self, handler: Handler[T], pattern: str
    ) -> None:
        """Handle subscription to a pattern by registering it and subscribing to matching topics."""
        await self._register_pattern_handler(pattern=pattern, handler=handler)
        await self._subscribe_pattern_to_existing_topics(
            handler=handler, pattern=pattern
        )

    async def _register_pattern_handler(
        self, pattern: str, handler: Handler[T]
    ) -> None:
        from typing import Literal

        from attrs import define

        from atta.ai.responses.common import OutputItem
        from atta.ai.responses.standard.text import TextResponse
        from atta.ai.responses.streaming.field import _semcov_status_field
        from atta.ai.responses.types import ResponseStatus

        # Event type constants
        TEXT_DELTA_EVENT = "response.output_text.delta"
        SUMMARY_TEXT_DELTA_EVENT = "response.reasoning_summary_text.delta"
        OUTPUT_ITEM_DONE_EVENT = "response.output_item.done"
        RESPONSE_COMPLETED_EVENT = "response.completed"

        @define(slots=True, kw_only=True)
        class TextResponseTextDeltaEvent:
            """Stream event for text delta."""

            delta: str
            logprobs: dict[str, float] | None = None
            type: Literal["response.output_text.delta"] = TEXT_DELTA_EVENT # type: ignore[assignment]
            status: ResponseStatus = _semcov_status_field("in_progress")

        @define(slots=True, kw_only=True)
        class TextResponseSummaryTextDeltaEvent:
            """Stream event for summary text delta."""

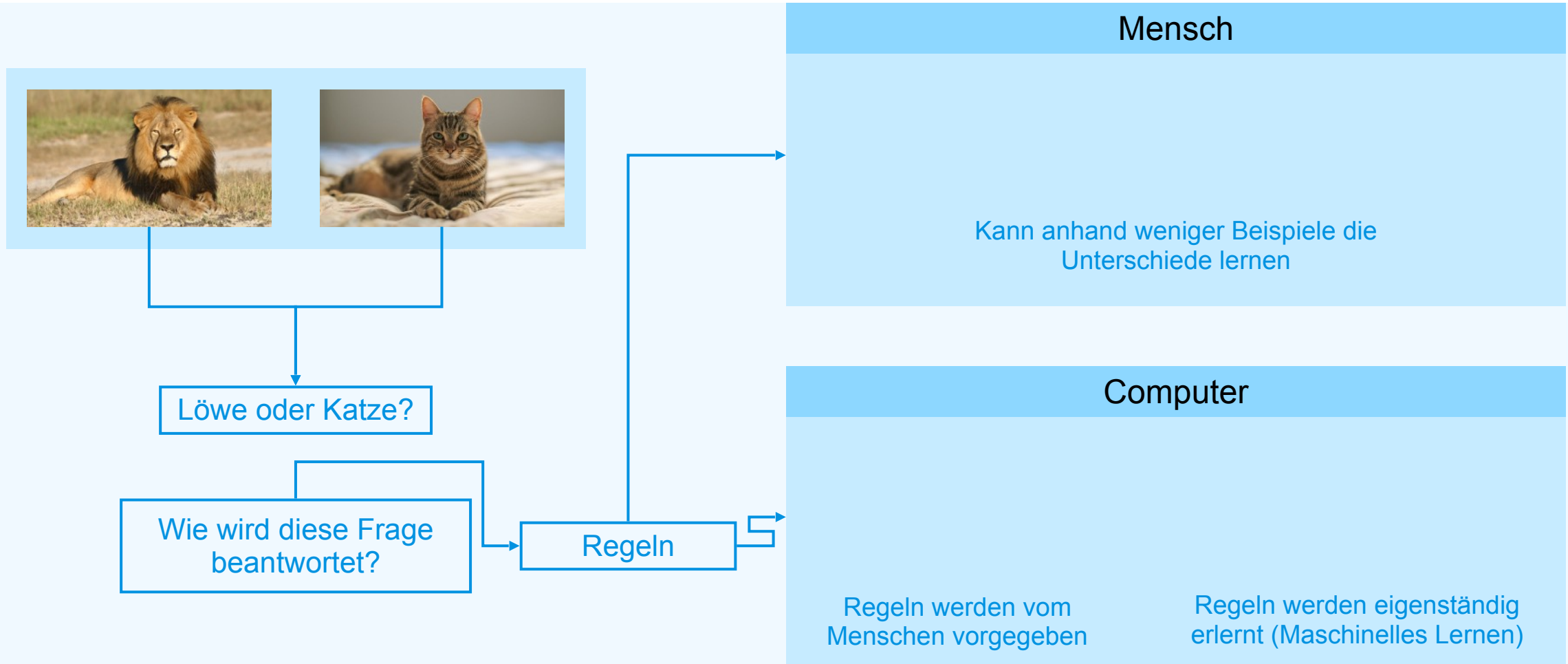
            delta: str
            type: Literal["response.reasoning_summary_text.delta"] = SUMMARY_TEXT_DELTA_EVENT # type: ignore[assignment]
            status: ResponseStatus = _semcov_status_field("in_progress")

        @define(slots=True, kw_only=True)
        class TextResponseOutputItemDoneEvent:
            """Stream event for output item done."""

            item: OutputItem
            type: Literal["response.output_item.done"] = OUTPUT_ITEM_DONE_EVENT # type: ignore[assignment]
            status: ResponseStatus = _semcov_status_field("in_progress")
```

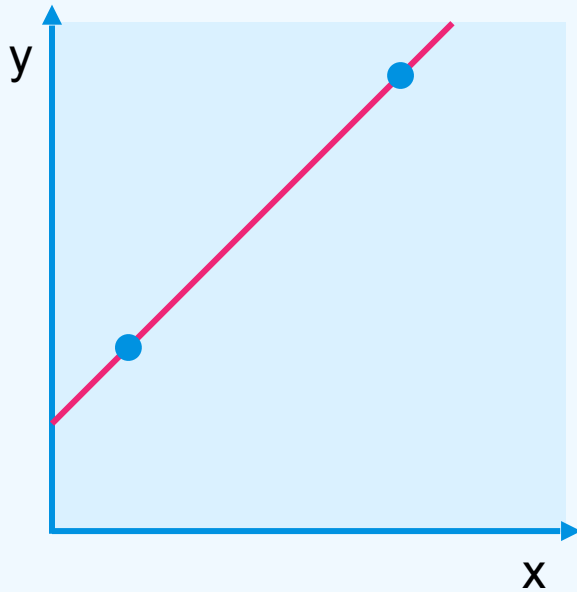
02 Was ist Künstliche Intelligenz (KI)?

Was ist der Kern von Künstlicher Intelligenz?



Mathematik ist die Sprache der KI

Schulmathematik



$$y = mx + b$$

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

Neuronale Netze

$$f_1(\mathbf{x}) = \sigma_1(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)$$

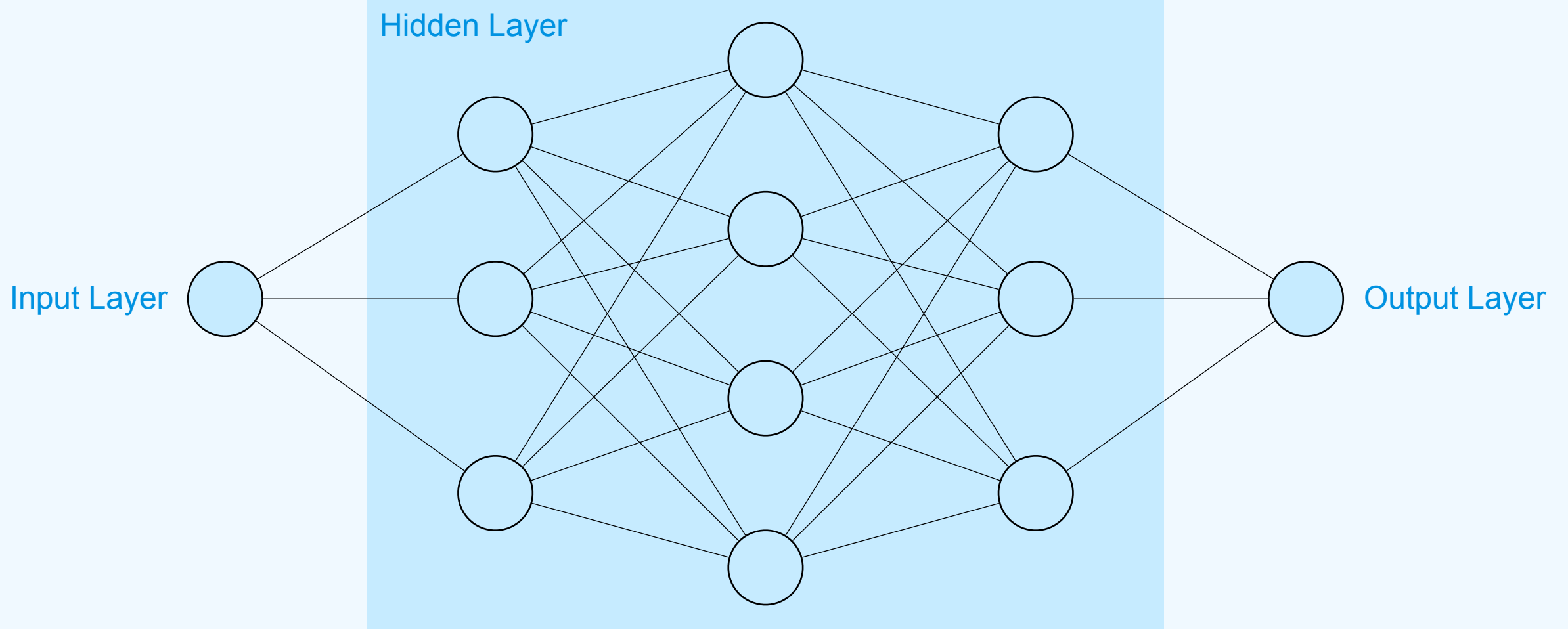
$$f_2(\mathbf{x}) = \sigma_2(\mathbf{W}_2f_1(\mathbf{x}) + \mathbf{b}_2)$$

$$f_3(\mathbf{x}) = \sigma_3(\mathbf{W}_3f_2(\mathbf{x}) + \mathbf{b}_3)$$

$$\mathbf{NN}(\mathbf{x}) = f_3(\mathbf{x})$$

KI ist vor allem Mathematik und keine Hexerei

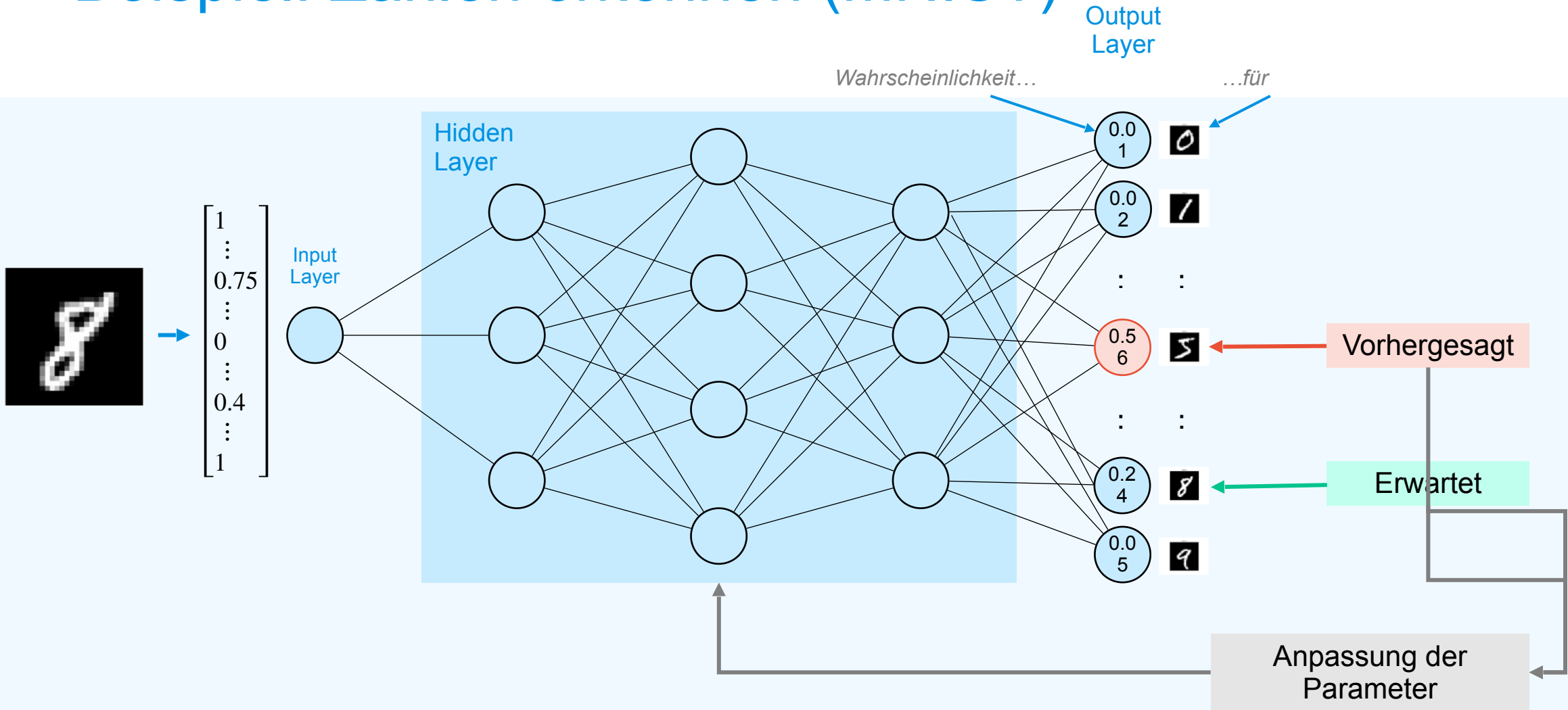
Was sind neuronale Netze?



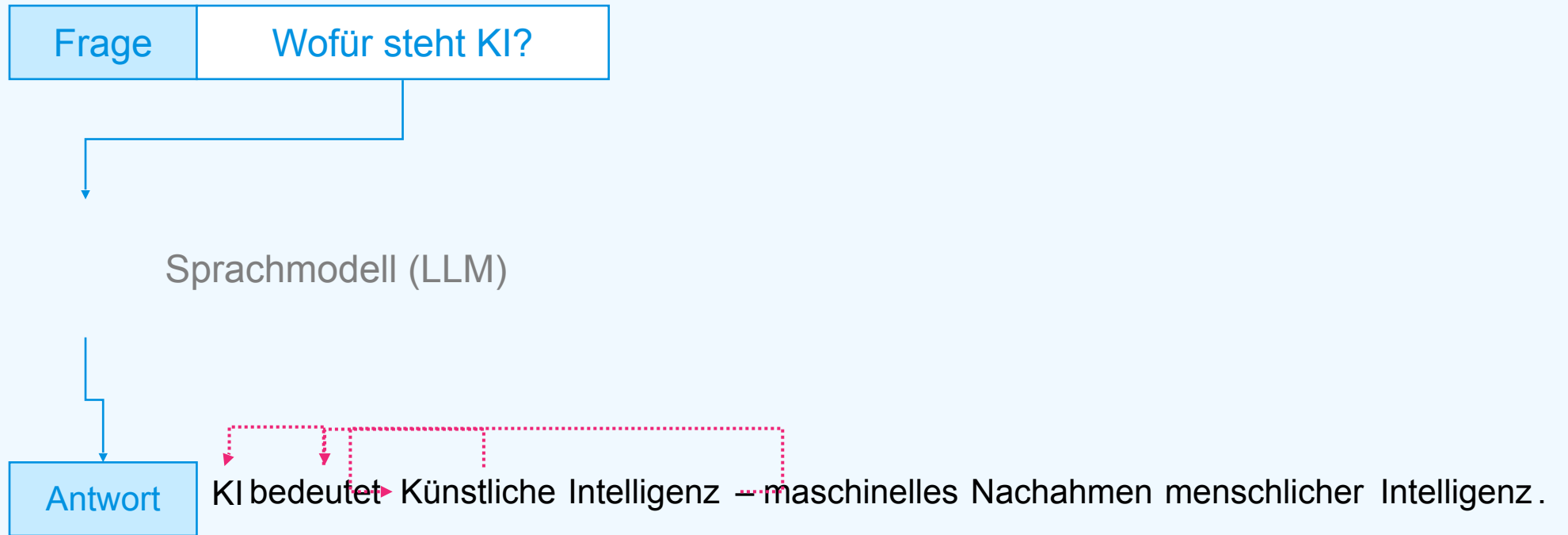
Beispiel: Zahlen erkennen (MNIST)



Beispiel: Zahlen erkennen (MNIST)



Wie funktionieren Sprachmodelle?



Zahlen und Fakten zum Training von LLMs



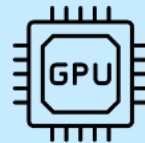
Training der Modelle kostet mehrere Millionen Dollar (teilweise über 100 Mio.)



Betrieb verursacht operative Kosten von zehn- bis hundert Millionen US-Dollar pro Tag



Eine Nvidia A100 Grafikkarte (GPU) kostet rund 10.000 USD



Das Training benötigt mehrere Tausend GPUs im Parallelbetrieb über teilweise 100 Tage



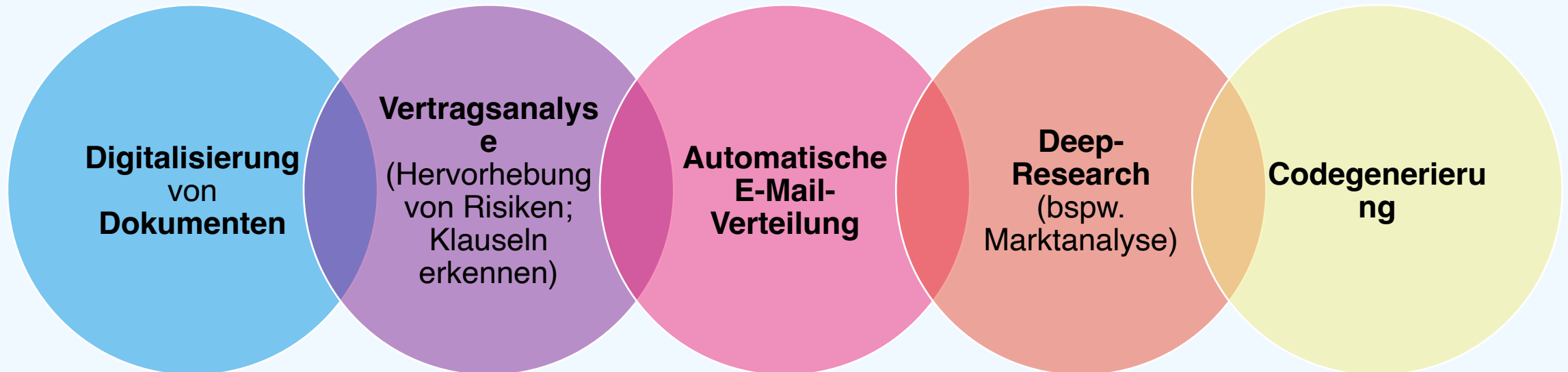
191 Mio. USD kostete das Training von Googles Gemini Ultra



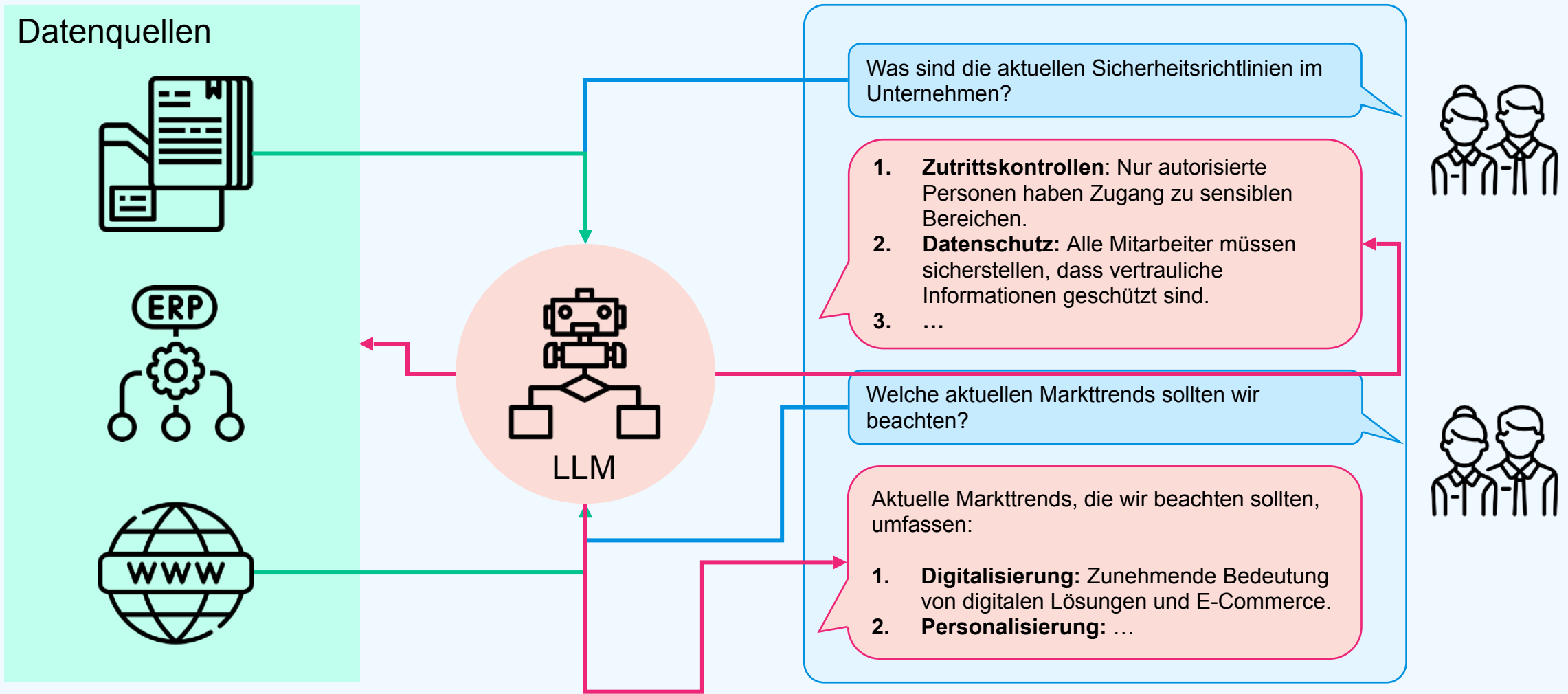
Der Trainingsdatensatz für ein Modelltraining hat teilweise 10 Billionen Token

03 Wie können Sie KI nutzen?

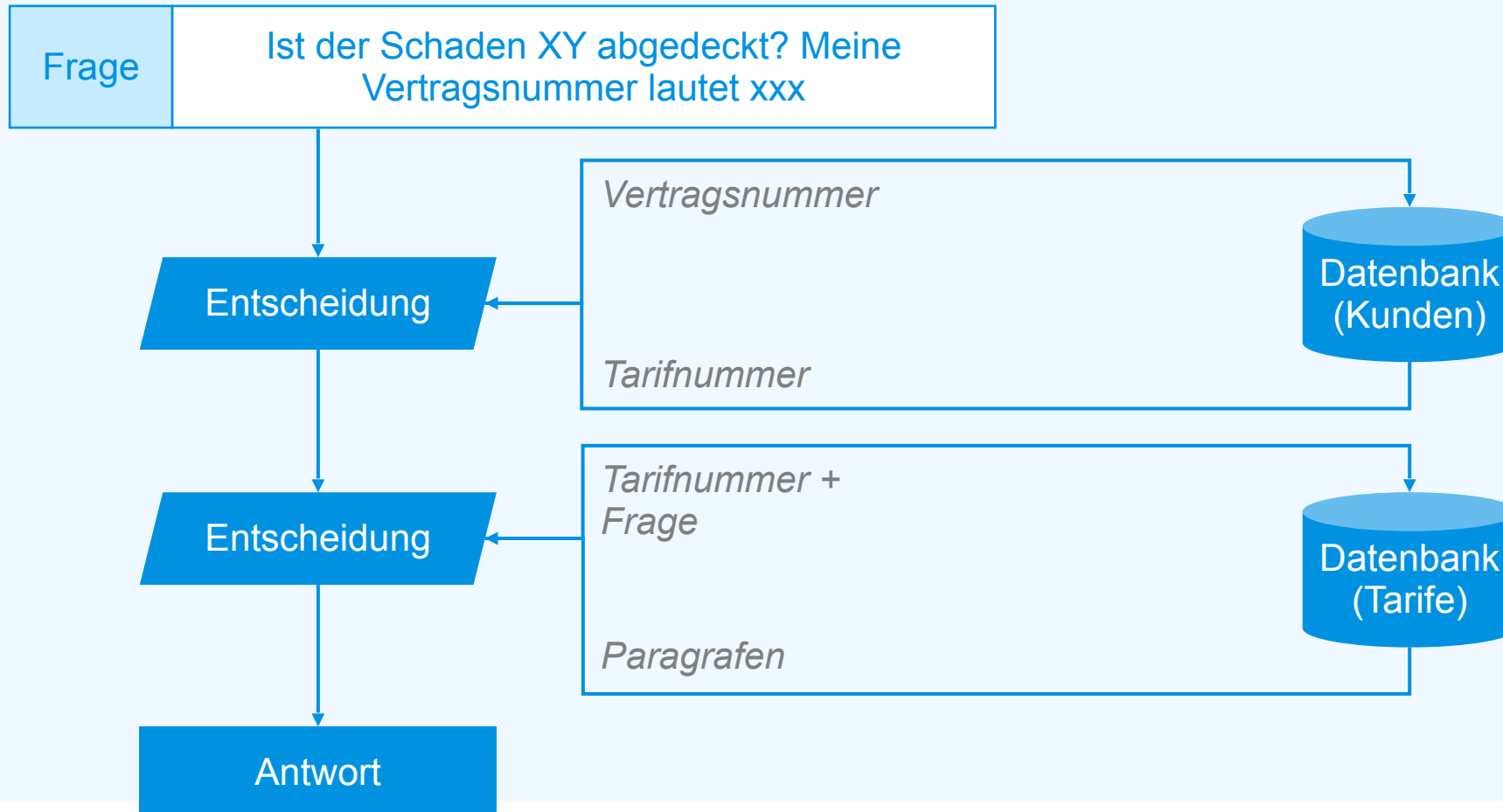
Beispiele als Anregung



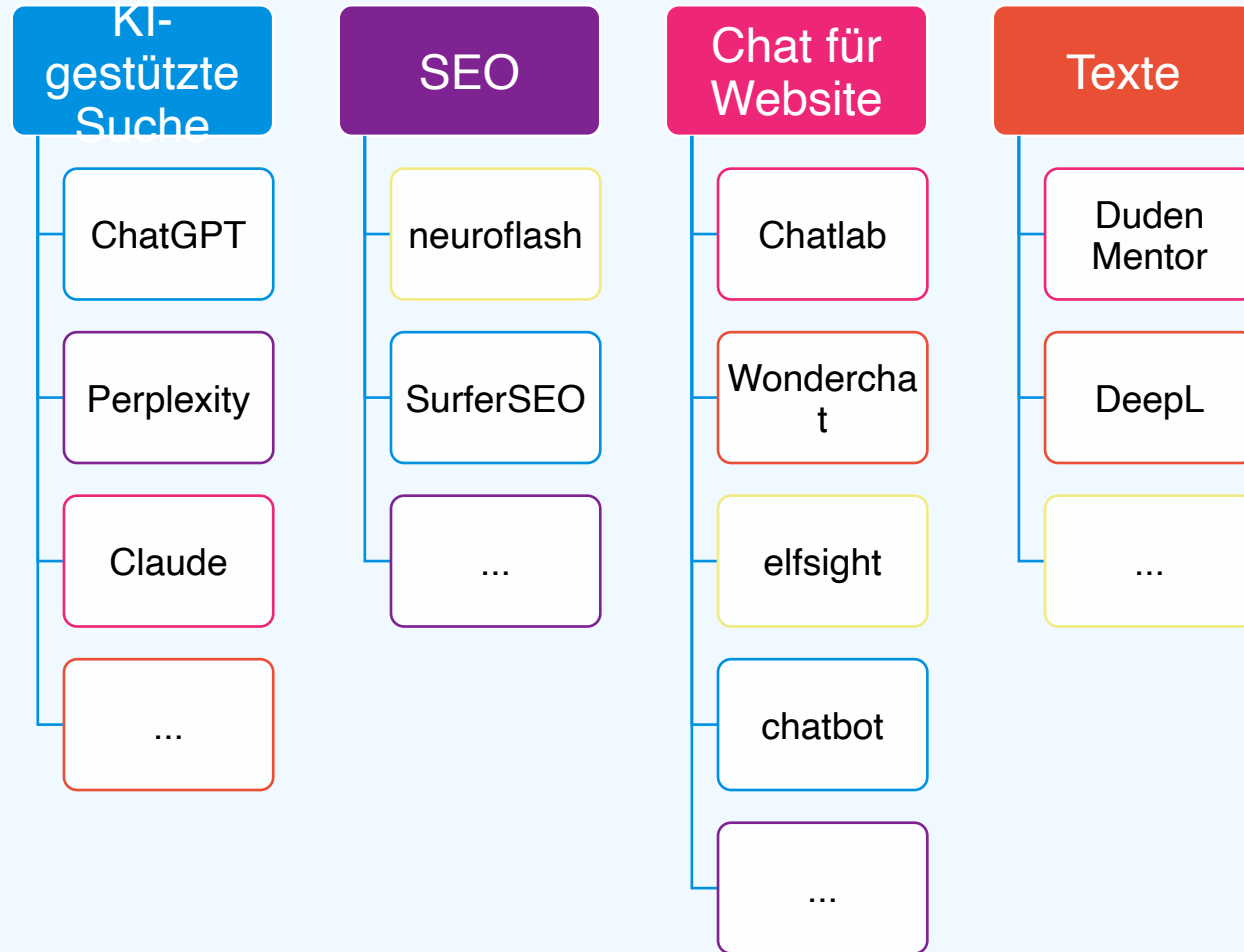
Sprachmodelle (LLM) für das Wissensmanagement



Agenten – Kundenanfrage bearbeiten



KI-Tools



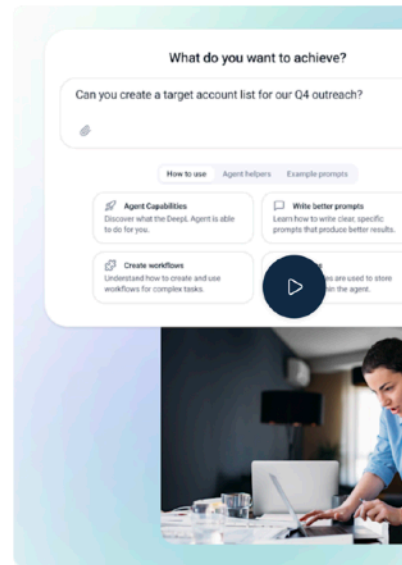
Agenten-Plattform



Begrüßen Sie Ihren neuen KI-Kollegen

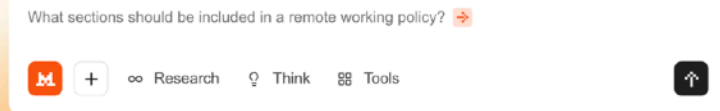
DeepL Agent kann die Wissensarbeit optimal unterstützen: Er übernimmt Routineaufgaben, damit sich Teams auf die Zusammenarbeit, auf stärkere Kundenbeziehungen und auf die Umsetzung neuer Ideen konzentrieren können.

[Sales-Team kontaktieren](#)



The AI assistant for your most meaningful work.

Frontier intelligence. Deep personalization. Full control.



[Try the Chat >](#) [Enterprise deployments >](#)

04 Meine Meinung & Einschätzung

Extreme Dimensionen



Stromverbrauch von GPT5 wird auf 1,5 Millionen US-Haushalte pro Tag geschätzt

Meine Meinung & Einschätzung

Menü | 🔍

Süddeutsche Zeitung

Jetzt abonnieren | Login

Meine SZ | SZ Plus | Nahost | USA | Ukraine | Politik | Wirtschaft | Meinung | Panorama | Sport | München | Kultur | Medien | B. >

Home > Wirtschaft > Technologie > Claude Mythos: Anthropic neues KI-Modell ist angeblich zu gefährlich für Öffentlichkeit

Claude Mythos von Anthropic

Ein KI-Modell, angeblich zu gefährlich für die Öffentlichkeit

8. April 2026, 15:15 Uhr | Lesezeit: 3 Min. | [6 Kommentare](#)

ARTIFICIAL INTELLIGENCE

OpenAI: Hey, We Also Have a New Tool That Is So Scarily Powerful We Can't Release It

Spooky season.

BY AJ DELLINGER | PUBLISHED APRIL 9, 2026, 3:35 PM ET

READING TIME 2 MINUTES

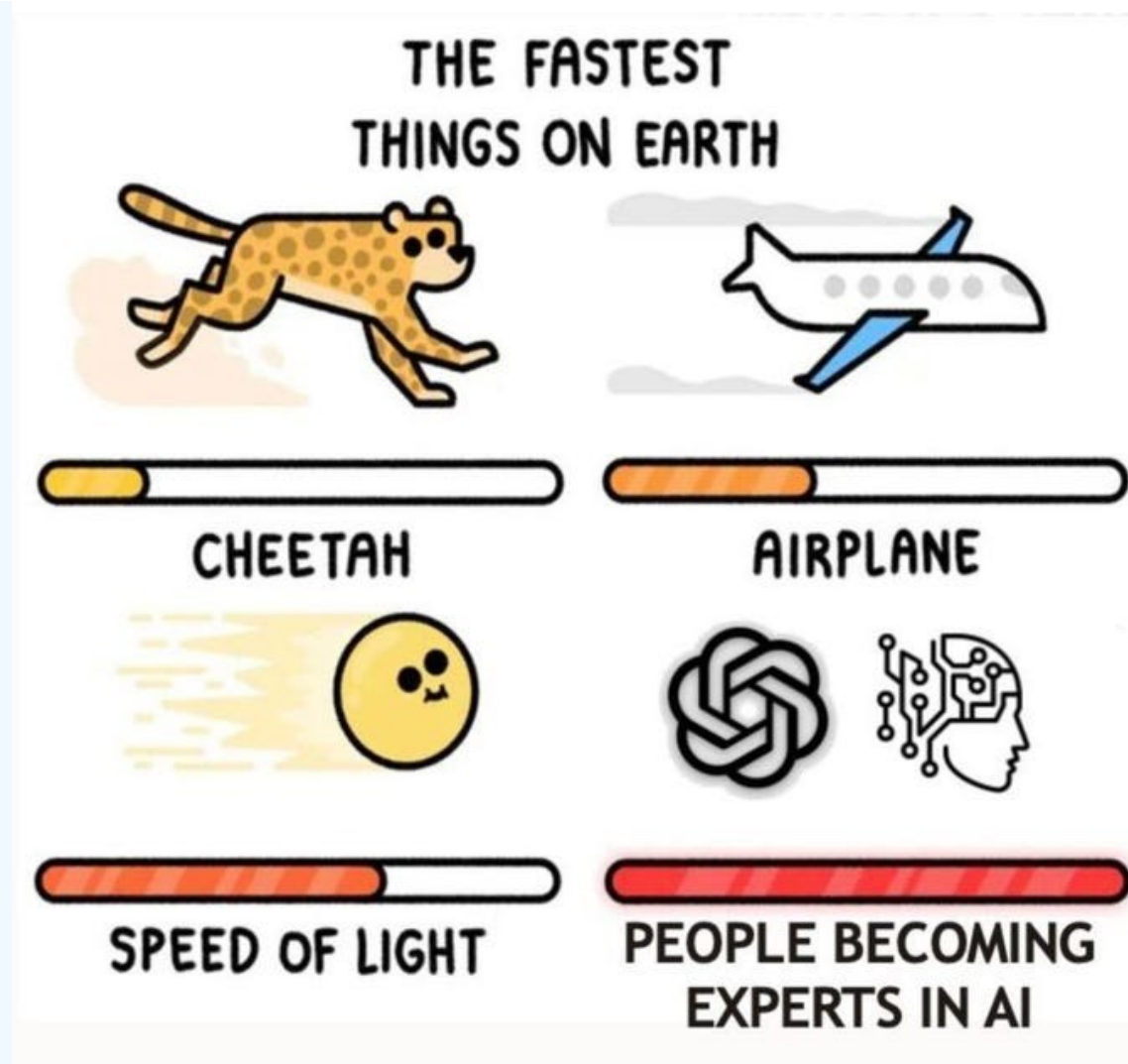
[READ LATER](#) | [COMMENTS \(22\)](#)



© Samuel Boivin/NurPhoto via Getty Images

Eigene Meinung – nicht die meines Arbeitgebers.

Meine Meinung & Einschätzung



Eigene Meinung – nicht die meines Arbeitgebers.

Meine Meinung & Einschätzung

The Washington Post Sign in

Tech Help Desk Artificial Intelligence Internet Culture

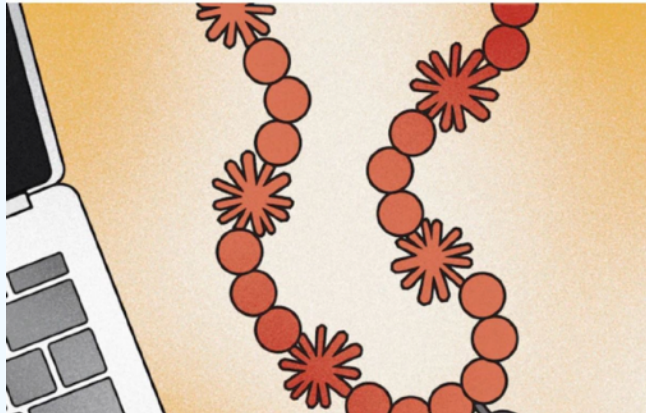
Can AI be a 'child of God'? Inside Anthropic's meeting with Christian leaders.

The artificial intelligence company asked religious leaders for guidance on building a moral chatbot.

April 11, 2026

7 min 436

Make us preferred on Google



Maria Sukhareva gefällt das.

Yann LeCun · 3.+
Executive Chairman, AMI Labs - Prof N...
9 Std. ·


Manufactured AI hysteria.

Übersetzung anzeigen

Subbarao... · 3.+
Prof at ASU (Former President of AAAI)
1 Tag(e) ·

tldr; on the latest "AI models lie, cheat and steal to protect other models from being deleted" manufactured hysteria.. (read this for a longer...mehr)

Übersetzung anzeigen



741 35 Kommentare · 30 Reposts

Eigene Meinung – nicht die meines Arbeitgebers.

05 Wie Mehrwert schaffen?

Mehrwert schaffen

Was?

Aufgaben, die sich **wiederholen**, **teuer** und **messbar** sind

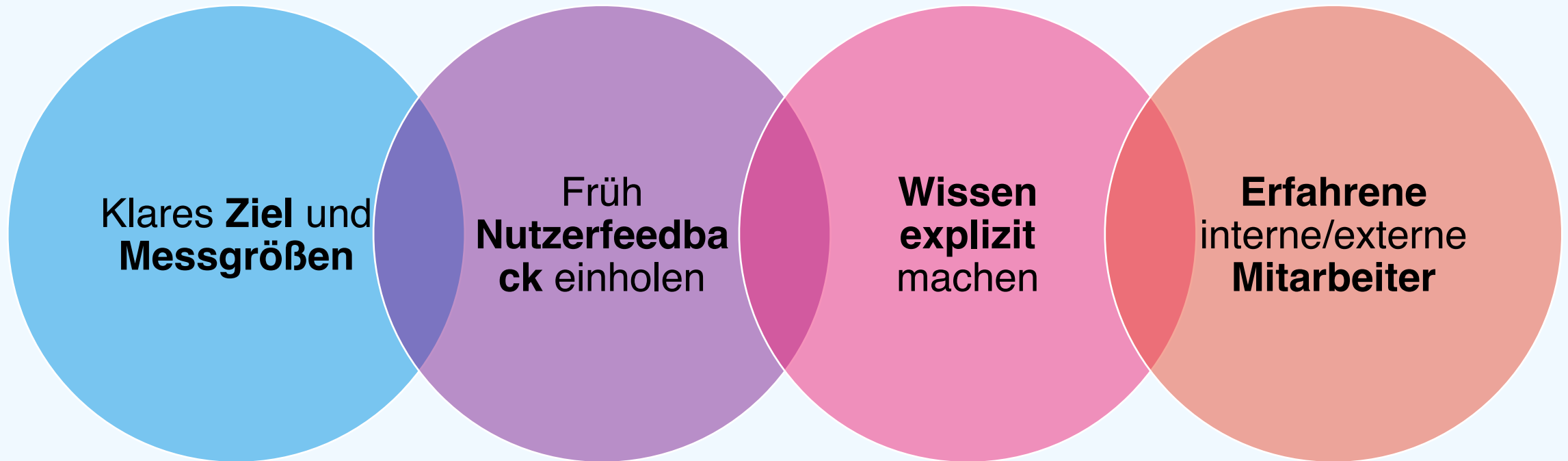
Wie?

KI-System **speichert Nutzerinformationen** (Memory) und **lernt** aus Fehlern und Nutzerfeedback

Wo?

Tiefe **Integration** in bestehende Systeme (**Daten** werden **zurückgeschrieben** und **Workflow** automatisch **ausgeführt**)

Was Sie beachten sollten



Mehrwert schaffen

Toy KI	Echte KI
Chat Tools	Workflow Engines (Agenten)
recherchiert und schlägt vor	führt autonom aus
vergisst (zustandslos)	lernt und speichert Kontext
ist generisch	ist konkret und domänen-spezifisch
bemisst sich an der Nutzung	bemisst sich an dem Nutzen