Generative KI-Systeme entwickeln

KI-Engineering für die Praxis – vom Prompt Engineering bis zu RAG und Agenten

DAS VORWORT

Vorwort

Als ChatGPT herauskam, war ich, wie viele meiner Kollegen und Kolleginnen, verwirrt. Für mich überraschend war weder die Größe des Modells noch seine Fähigkeiten. Seit mehr als einem Jahrzehnt wusste die KI-Community, dass ein Modell besser wird, wenn man es größer macht. Im Jahr 2012 stellten die Autoren von AlexNet in ihrem bahnbrechenden Artikel fest, dass »... alle unsere Experimente zeigen, dass sich unsere Ergebnisse allein dadurch verbessern lassen, dass wir auf schnellere GPUs und die Verfügbarkeit größerer Datensätze warten.« (*https://oreil.ly/XG3mv*).[1, 2]

Überraschend für mich war die schiere Menge an Anwendungen, die dieser Fähigkeitsschub ermöglichte. Ich dachte, eine kleine Verbesserung in den Metriken der Modellqualität würde auch nur zu einer kleinen Verbesserung bei den Anwendungen führen. Stattdessen führte es zu einer Explosion neuer Möglichkeiten.

Diese neuen KI-Fertigkeiten steigerten nicht nur den Bedarf an KI-Anwendungen, sie senkten auch die Einstiegshürde bei der Entwicklung. Es ist so einfach geworden, mit dem Bauen von KI-Anwendungen loszulegen. Es ist sogar möglich, eine Anwendung zu erstellen, ohne eine einzige Zeile Code zu schreiben. Dieser Wechsel hat die KI von einer spezialisierten Disziplin in ein leistungsfähiges Entwicklungstool transformiert, das jeder nutzen kann.

Obwohl der Einsatz von KI heutzutage als neu erscheint, baut er auf Techniken auf, die es schon eine ganze Weile gibt. Artikel über die Modellierung von Sprache erschienen schon in den 1950er-Jahren. Anwendungen mit *Retrieval-Augmented Generation* (RAG) bauen schon auf Retrieval-Technologien auf, die die Grundlage für Systeme zum Suchen und Empfehlen bildeten, als der Begriff RAG noch gar nicht geprägt war. Die Best Practices zum Deployen klassischer Anwendungen zum maschinellen Lernen – systematische Experimente, rigorose Evaluation, fortlaufende Opti-

1 Ilya Sutskever, einer der Autoren des AlexNet-Artikels, begründete später OpenAI mit und setzte dieses Wissen dann mit GPT-Modellen in die Realität um.

2 Selbst mein kleines Projekt aus dem Jahr 2017 (*https://x.com/chipro/status/937384141791698944*), bei dem ein Sprachmodell zum Bewerten der Qualität von Übersetzungen genutzt wurde, kam zu dem Schluss, dass wir »ein besseres Sprachmodell brauchten«.

mierung für schnellere und günstigere Modelle – haben weiterhin Gültigkeit, wenn Sie mit Anwendungen arbeiten, die Foundation Models nutzen.

Die Vertrautheit und die Einfachheit vieler KI-Engineering-Techniken können die Menschen zu der irrigen Annahme führen, es gäbe bei der KI-Entwicklung nichts Neues zu beachten. Auch wenn viele Prinzipien für das Bauen von KI-Anwendungen bestehen bleiben, sorgen die Größe und die verbesserten Fähigkeiten von KI-Modellen für Möglichkeiten und Herausforderungen, für die neue Lösungen erforderlich sind.

Dieses Buch behandelt den kompletten Prozess der Verwendung von Foundation Models zum Lösen realer Probleme vom Anfang bis zum Ende. Es kommen erprobte Techniken aus anderen Entwicklungsbereichen zum Einsatz, aber auch Techniken, die erst mit Foundation Models entstanden sind.

Ich wollte dieses Buch schreiben, weil ich etwas lernen wollte – und ich habe sehr viel gelernt. Ich habe aus den Projekten gelernt, an denen ich arbeitete, aus den gelesenen Artikeln und von den Personen, mit denen ich Gespräche geführt habe. Während des Schreibens dieses Buchs habe ich Notizen aus über 100 Unterhaltungen und Interviews verwendet, unter anderen mit Forschenden aus großen KI-Labs (OpenAI, Google, Anthropic ...), Personen aus der Framework-Entwicklung (NVIDIA, Meta, Hugging Face, Anyscale, LangChain, LlamaIndex ...), Führungskräften und KI-/Datenverantwortlichen in Firmen unterschiedlichster Größe, Produktmanagern, Community-Forschenden sowie unabhängigen Anwendungsentwicklerinnen und -entwicklern (siehe den Abschnitt »Danksagung« auf Seite 21).

Insbesondere von Leserinnen und Lesern früherer Versionen des Manuskripts habe ich gelernt. Sie stellten meine Annahmen auf die Probe, zeigten mir andere Perspektiven und konfrontierten mich mit neuen Problemen und Ansätzen. Einige Abschnitte dieses Buchs haben zudem Tausende von Kommentaren aus der Community erhalten, nachdem ich sie in meinem Blog geteilt habe (*https://huyenchip.com/blog/*). Dort fand ich oft andere Sichtweisen oder aber auch die Bestätigung von Hypothesen.

Ich hoffe, dass bei mir dieser Lernprozess nicht endet, nachdem Sie das Buch in Ihren Händen hatten – schließlich haben auch Sie Erfahrungen und Perspektiven, die einmalig sind. Ich freue mich sehr über Feedback zu diesem Buch – über X (*https://x.com/chipro*), LinkedIn (*https://www.linkedin-com/in/chiphuyen*) oder per E-Mail an *hi@huyenchip.com*.

Worum es in diesem Buch geht

Dieses Buch bietet einen Rahmen für das Anpassen von Foundation Models – zu denen sowohl *Large Language Models* (LLMs) als auch *Large Multimodal Models* (LMMs) gehören – an spezifische Anwendungen.

Es gibt viele verschiedene Wege, um eine Anwendung zu bauen. In diesem Buch wird eine Reihe von Lösungen umrissen, und auch Fragen werden vorgestellt, die Sie

sich stellen können, um die beste Lösung für Ihre Anforderungen zu finden. Zu diesen vielen Fragen, bei deren Beantwortung Ihnen dieses Buch helfen kann, gehören unter anderem:

- Sollte ich diese KI-Anwendung bauen?
- Wie evaluiere ich meine Anwendung? Kann ich KI nutzen, um KI-Ausgaben zu bewerten?
- Was führt zu Halluzinationen? Wie erkenne und vermeide ich Halluzinationen?
- Was sind die Best Practices für das Prompt Engineering?
- Wie funktioniert RAG? Was für Strategien kommen zum Einsatz?
- Was ist ein Agent? Wie baue und evaluiere ich einen Agenten?
- Wann optimiere ich ein Modell? Wann tue ich das nicht?
- Wie viele Daten brauche ich? Wie bewerte ich die Qualität meiner Daten?
- Wie mache ich mein Modell schneller, günstiger und sicher?
- Wie erschaffe ich eine Feedback-Schleife, um meine Anwendung fortlaufend zu verbessern?

Dieses Buch wird Ihnen auch dabei helfen, sich in der ausufernden KI-Landschaft zurechtzufinden: Sie erfahren etwas über die Arten von KI-Modellen, über Evaluations-Benchmarks und über eine anscheinend unendliche Zahl von Anwendungsfällen und Applikationsmustern.

Der Inhalt dieses Buchs wird durch Fallstudien illustriert, an denen ich zum größeren Teil selbst gearbeitet habe und die durch viele Referenzen ergänzt und durch Expertinnen und Experten unterschiedlichsten Hintergrunds umfassend begutachtet wurden. Auch wenn dieses Buch zwei Jahre zum Entstehen brauchte, greift es auf meine Erfahrungen bei der Arbeit mit Sprachmodellen und ML-Systemen aus dem letzten Jahrzehnt zurück.

Wie schon mein früheres O'Reilly-Buch *Designing Machine Learning Systems* (DMLS) konzentriert sich dieses Buch auf die Grundlagen der KI-Entwicklung und nicht auf spezifische Tools oder APIs. Tools veralten schnell, aber die Grundlagen sollten länger gültig sein.[3]

Generative KI-Systeme entwickeln zusammen mit *Designing Machine Learning Systems* lesen

Das vorliegende Buch *Generative KI-Systeme entwickeln* (englisch *AI Engineering*, AIE) kann eine gute Ergänzung zu *Designing Machine Learning Systems* (DMLS) sein. DMLS konzentriert sich auf das Bauen von Anwendungen, die auf klassischen ML-Modellen aufbauen. Dafür sind mehr Annotationen an Datentabellen, Feature Engineering und Modelltraining erforderlich. AIE fokussiert sich darauf, Anwendungen zu bauen, die auf

3 Das Unterrichten eines Kurses zum Einsatz von TensorFlow im Jahr 2017 hat mir schmerzhaft gezeigt, wie schnell Tools und Tutorien veralten.

Foundation Models basieren – das beinhaltet notwendigerweise mehr Prompt Engineering, das Schaffen von Kontext und ein parametereffizientes Optimieren. Beide Bücher können für sich genutzt werden, und beide sind modular aufgebaut, sodass Sie sie unabhängig voneinander lesen können.

Da Foundation Models ML-Modelle sind, gibt es einige Konzepte, die für beide relevant sind. Ist ein Thema für AIE wichtig, wurde aber ausführlich in DMLS behandelt, werden Sie es trotzdem hier finden, wenn auch weniger umfangreich, dafür aber mit Verweisen auf relevanten Ressourcen.

Beachten Sie, dass viele Themen in DMLS, aber nicht in AIE behandelt werden – und umgekehrt. Das erste Kapitel dieses Buchs geht auch auf die Unterschiede zwischen klassischem ML Engineering und KI Engineering ein. Ein reales System beinhaltet oft sowohl klassische ML-Modelle als auch Foundation Models, daher ist meist Wissen über beides erforderlich.

Es ist oft eine Herausforderung, zu erkennen, ob etwas von Dauer sein wird. Ich verlasse mich da auf drei Kriterien. So bestimme ich für ein Problem, ob es das Ergebnis der grundlegenden Beschränkungen des Vorgehens von KI ist oder ob es mit besseren Modellen verschwinden wird. Ist ein Problem grundlegend, analysiere ich die Herausforderungen und mögliche Lösungen dazu. Ich fange gern erst mal einfach an, daher lege ich bei vielen Problemen mit der einfachsten Lösung los und steigere mich zu komplexeren Lösungen, wenn die Herausforderungen größer werden.

Dann tauche ich in ein umfassendes Netzwerk aus Forscherinnen und Entwicklern ein, die klüger sind als ich, und versuche, herauszufinden, was ihrer Meinung nach die wichtigsten Probleme und Lösungen sind.

Gelegentlich verlasse ich mich auch auf Lindy's Law (*https://en.wikipedia.org/wiki/Lindy_effect*), das besagt, dass die restliche zu erwartende Lebensspanne einer Technologie proportional zu ihrem Alter ist. Gibt es also etwas schon eine Weile, gehe ich davon aus, dass es auch noch länger vorhanden sein wird.

In dieses Buch habe ich aber vereinzelt auch Konzepte aufgenommen, die meiner Meinung nach nur temporär sind, weil sie für die Anwendungsentwicklung manchmal direkt Nutzen bringen oder weil sie einen interessanten Lösungsansatz illustrieren.

Was dieses Buch nicht ist

Dieses Buch ist kein Tutorial. Wenn es spezifische Tools erwähnt und Pseudocodeschnipsel enthält, um bestimmte Konzepte zu illustrieren, erfahren Sie dadurch nicht, wie Sie ein Tool verwenden. Stattdessen bietet es einen Rahmen für das Auswählen von Tools. Sie finden hier viele Erläuterungen zu den Vor- und Nachteilen verschiedener Lösungen und Fragen, die Sie sich stellen sollten, wenn Sie eine Lösung bewerten. Wollen Sie ein Tool einsetzen, ist es im Allgemeinen einfach, Tutorials dafür online zu finden. KI-Chatbots sind ebenfalls ziemlich gut darin, Ihnen beim Einstieg in verbreitete Tools zu helfen.

Dieses Buch ist kein Buch zur ML-Theorie. Es erklärt nicht, was ein neuronales Netz ist oder wie Sie ein Modell von Grund auf bauen und trainieren. Es geht zwar auf viele theoretische Konzepte ein, die für das Buch direkt relevant sind, aber das Buch ist ein Praxisbuch, das sich darauf konzentriert, Ihnen beim Bauen erfolgreicher KI-Anwendungen zum Lösen realer Probleme zu helfen.

Es ist zwar möglich, ohne ML-Wissen Anwendungen zu bauen, die auf Foundation Models beruhen, aber Basiswissen über ML und Statistik kann dabei helfen, bessere Anwendungen zu erstellen und unnötige Probleme zu umgehen. Sie können dieses Buch ohne ML-Hintergrund lesen. Aber Sie werden beim Bauen von KI-Anwendungen effektiver sein, wenn Sie folgende Konzepte kennen:

- Wahrscheinlichkeitskonzepte wie Sampling, Determinismus und Verteilung.
- ML-Konzepte wie überwachtes und selbstüberwachtes Lernen, Log-Wahrscheinlichkeit, das Gradientenverfahren, Backpropagation, die Verlustfunktion und das Optimieren von Hyperparametern.
- Die verschiedenen Architekturen neuronaler Netze, unter anderem Feedforward, Rekurrent und Transformer.
- Metriken wie Originaltreue, F1, Präzision, Recall, Kosinus-Ähnlichkeit und Kreuzentropie.

Ist Ihnen all das noch nicht bekannt, müssen Sie sich keine Sorgen machen – in diesem Buch finden Sie entweder kurze Erläuterungen dazu oder Verweise auf Ressourcen, bei denen Sie sich schlaumachen können.

Für wen dieses Buch gemacht ist

Dieses Buch ist für alle, die mithilfe von Foundation Models reale Probleme lösen wollen. Es handelt sich um ein technisches Buch, daher richtet sich die Sprache an technische Rollen: KI-Entwickler, ML Engineers, Data Scientists, Entwicklungsmanagerinnen und an das technische Produktmanagement. Dieses Buch ist etwas für Sie, wenn Sie sich in einem der folgenden Szenarien wiederfinden:

- Sie bauen oder optimieren eine KI-Anwendung – entweder von Anfang an, oder Sie wollen die Demophase hinter sich lassen und sie produktiv nutzen können. Eventuell haben Sie Probleme mit Halluzinationen, mit der Sicherheit, der Latenz oder den Kosten und brauchen zielgerichtete Lösungen.
- Sie wollen den KI-Entwicklungsprozess Ihres Teams vereinfachen, ihn systematischer, schneller und zuverlässiger machen.
- Sie wollen verstehen, wie Ihre Organisation Foundation Models nutzen kann, um den Gewinn zu steigern, und wie Sie dafür ein Team aufbauen können.

Sie profitieren auch dann von dem Buch, wenn Sie zu einer der folgenden Gruppen gehören:

- Toolentwickler, die unterversorgte Bereiche der KI-Entwicklung erkennen wollen, um ihr Produkt dort platzieren zu können.

- Forscherinnen, die Anwendungsfälle für die KI besser verstehen wollen.
- Jobkandidaten, die sich darüber informieren wollen, welche Fähigkeiten notwendig sind, um eine Karriere als AI Engineer starten zu können.
- Alle, die die Fähigkeiten und Grenzen der KI und ihren Einfluss auf die verschiedenen Rollen besser verstehen möchten.

Ich durchschaue Dinge gern vollständig, daher gehen einige Abschnitte vielleicht aus technischer Sicht etwas weiter. Viele Leserinnen und Leser früherer Versionen des Manuskripts freuten sich zwar über die vielen Details, aber eventuell ist das nicht für jeden etwas. Ich werde Ihnen einen Überblick geben, bevor es zu technisch wird. Sie können den folgenden Abschnitt dann gern überspringen, wenn Ihnen der ein bisschen zu weit geht!

Wie Sie sich im Buch zurechtfinden

Dieses Buch ist so strukturiert, dass es den typischen Prozess beim Entwickeln einer KI-Anwendung nachzeichnet. Im Folgenden beschreibe ich, wie dieser typische Prozess aussieht und wie jedes Kapitel in den Prozess passt. Weil das Buch modular aufgebaut ist, können Sie gern Abschnitte überspringen, mit denen Sie schon vertraut oder die für Sie nicht so relevant sind.

Bevor Sie sich dafür entscheiden, eine KI-Anwendung zu bauen, ist es erforderlich, zu verstehen, was zu diesem Prozess gehört, und sich folgende Fragen zu stellen: Ist diese Anwendung notwendig? Wird KI gebraucht? Muss ich die Anwendung selbst bauen? Das erste Kapitel des Buchs hilft Ihnen dabei, diese Fragen zu beantworten. Sie finden hier auch eine Reihe hilfreicher Anwendungsfälle, mit denen Sie ein Gefühl dafür bekommen können, was mit Foundation Models möglich ist.

Wissen im Bereich des maschinellen Lernens ist zwar nicht notwendig, um KI-Anwendungen zu bauen, aber es hilft, zu verstehen, wie ein Foundation Model hinter den Kulissen funktioniert, um das Beste aus ihm herauszuholen. In Kapitel 2 werden das Entstehen eines Foundation Model und die Designentscheidungen analysiert, die signifikanten Einfluss auf die darauf aufbauenden Anwendungen haben. Dazu gehören die Rezepte für die Trainingsdaten, die Modellarchitekturen und Größenordnungen und wie das Modell trainiert wird, damit es die menschlichen Anforderungen erfüllt. Dann berichte ich davon, wie ein Modell eine Antwort generiert. Das hilft dabei, scheinbar rätselhafte Verhaltensweisen wie Inkonsistenzen oder Halluzinationen zu erklären. Das Ändern der Generierungseinstellungen eines Modells ist zudem oft eine günstige und einfache Möglichkeit, dessen Performance deutlich zu verbessern.

Haben Sie sich dazu entschieden, eine Anwendung mit Foundation Models zu bauen, wird die Evaluation ein integraler Bestandteil jedes folgenden Schritts sein. Evaluation ist eine der schwersten – wenn nicht die schwerste – Herausforderung beim Entwickeln generativer KI-Systeme. Dieses Buch widmet ganze zwei Kapitel, die Kapitel 3 und 4, dem Erforschen verschiedener Evaluationsmethoden und deren

Einsatz beim Erstellen einer zuverlässigen und systematischen Evaluations-Pipeline für Ihre Anwendung.

Bei einer gegebenen Anfrage hängt die Qualität der Antwort des Modells von den folgenden Aspekten ab (abgesehen von den Generierungseinstellungen des Modells):

- von den Anweisungen dazu, wie sich das Modell verhalten soll,
- dem Kontext, den das Modell nutzen kann, um auf die Anfrage zu reagieren, sowie
- vom Modell selbst.

Die nächsten drei Kapitel des Buchs fokussieren sich darauf, wie man jeden dieser Aspekte optimiert, um die Performance eines Modells für eine Anwendung zu verbessern. In Kapitel 5 geht es um Prompt Engineering. Hier wird erklärt, was ein Prompt ist, warum Prompt Engineering funktioniert und welche Best Practices es dafür gibt. Außerdem geht es darum, wie böswillige Akteure Ihre Anwendung durch Prompt-Angriffe missbrauchen können und wie Sie Ihre Anwendung dagegen schützen.

In Kapitel 6 schauen wir uns an, warum der Kontext zum Generieren akkurater Antworten für ein Modell so wichtig ist. Dabei geht es vor allem um zwei relevante Anwendungs-Patterns für das Aufbauen eines Kontexts: RAG und agentenbasiert. Das RAG-Pattern ist besser verstanden, und es hat sich auch schon in Produktivumgebungen als erfolgreich erwiesen. Das agentenbasierte Pattern verspricht hingegen, leistungsfähiger zu sein, es ist aber komplexer und auch noch nicht ganz erforscht.

In Kapitel 7 geht es darum, wie Sie ein Modell an eine Anwendung anpassen, indem Sie es durch Optimieren verändern. Aufgrund der Größenordnung von Foundation Models ist das native Optimieren des Modells sehr speicherintensiv, und es wurden viele Techniken entwickelt, um mit weniger Speicherbedarf Modelle zu optimieren. Das Kapitel stellt unterschiedliche Ansätze beim Optimieren vor, ergänzt durch einen weiteren, eher experimentellen Ansatz – das Model Merging. Außerdem finden Sie hier einen mehr technisch ausgerichteten Abschnitt, der zeigt, wie Sie den Speicherbedarf eines Modells berechnen.

Aufgrund der Verfügbarkeit vieler Optimierungs-Frameworks ist der Optimierungsprozess selbst oft nicht kompliziert. Aber es ist schwierig, Daten dafür zu erhalten. Im nächsten Kapitel geht es daher um Daten – deren Beschaffen, Annotieren, Synthetisieren und Verarbeiten. Viele der Themen aus Kapitel 8 sind auch über das Optimieren hinaus relevant – unter anderem die Frage, was Datenqualität bedeutet und wie Sie die Qualität Ihrer Daten bewerten können.

Geht es in den Kapiteln 5 bis 8 um das Verbessern der Modellqualität, dreht es sich in Kapitel 9 darum, das Inferieren oder Schlussfolgern des Modells günstiger und schneller zu machen. Hier wird vorgestellt, wie Sie auf Ebene des Modells und des Inferenzservice Optimierungen vornehmen können. Nutzen Sie eine Modell-API – zum Beispiel weil jemand anderes Ihr Modell für Sie hostet –, wird diese API sehr wahrscheinlich die Inferenzoptimierung für Sie übernehmen. Aber wenn Sie Ihr Mo-

dell selbst hosten – entweder ein Open-Source-Modell oder ein selbst entwickeltes Modell –, werden Sie viele der in diesem Kapitel vorgestellten Techniken implementieren müssen.

Das letzte Kapitel des Buchs führt die verschiedenen im Buch vorgestellten Konzepte zusammen, um eine Anwendung vom Anfang bis zum Ende zu bauen. Der zweite Teil des Kapitels fokussiert sich mehr auf das Produkt. Hier geht es darum, wie Sie ein System zum Benutzer-Feedback designen, das Ihnen dabei hilft, nützliches Feedback zu erhalten und für eine gute User Experience zu sorgen.

Hinweis

In diesem Buch verwende ich oft das »wir«, mit dem ich mich auf Sie (als Leserin oder Leser) und mich beziehe. Das ist eine alte Angewohnheit aus meiner Zeit als Dozentin, in der ich das Schreiben als gemeinsame Lernerfahrung für die Autorin und die Leser betrachtet habe.

In diesem Buch genutzte Konventionen

Die folgenden typografischen Konventionen werden in diesem Buch genutzt:

Kursiv
: Für neue Begriffe, URLs, E-Mail-Adressen, Dateinamen und Dateierweiterungen.

`Nichtproportionalschrift`
: Für Programmlistings, aber auch für Codefragmente in Absätzen, wie zum Beispiel Variablen- oder Funktionsnamen, Datenbanken, Datentypen, Umgebungsvariablen, Anweisungen und Schlüsselwörter.

`Fette Nichtproportionalschrift`
: Für Befehle und anderen Text, der genau so vom Benutzer eingegeben werden sollte.

`Kursive Nichtproportionalschrift`
: Für Text, der vom Benutzer durch eigene Werte ersetzt werden sollte.

Dieses Symbol steht für einen Tipp oder Vorschlag.

Dieses Symbol steht für eine allgemeine Anmerkung.

Dieses Symbol steht für eine Warnung oder Vorsichtsmaßnahme.

Verwendung von Codebeispielen

Zusätzliches Material (Codebeispiele, Übungen und so weiter) finden Sie zum Herunterladen unter *https://github.com/chiphuyen/aie-book*. Das Repository enthält auch zusätzliche Ressourcen zum Entwickeln generativer KI-Systeme, unter anderem wichtige Artikel und nützliche Tools. Außerdem werden hier Themen behandelt, die den Rahmen des Buchs sprengen würden. Wer am Prozess des Schreibens dieses Buchs interessiert ist, findet in diesem GitHub-Repository zudem Informationen und Statistiken, die hinter die Kulissen schauen.

Dieses Buch soll Ihnen bei Ihrer Arbeit helfen. Ganz allgemein gilt: Wenn in diesem Buch Beispielcode angeboten wird, können Sie ihn in Ihren Programmen und Dokumentationen verwenden. Sie müssen sich dafür nicht unsere Erlaubnis einholen, es sei denn, Sie reproduzieren einen großen Teil des Codes. Schreiben Sie zum Beispiel ein Programm, das mehrere Teile des Codes aus diesem Buch nutzt, brauchen Sie keine Erlaubnis. Verkaufen oder vertreiben Sie Beispiele aus O'Reilly-Büchern, brauchen Sie eine Erlaubnis. Beantworten Sie eine Frage, indem Sie dieses Buch und Beispielcode daraus zitieren, benötigen Sie keine Erlaubnis. Binden Sie einen großen Anteil des Beispielcodes aus diesem Buch in die Dokumentation Ihres Produkts ein, brauchen Sie eine Erlaubnis.

Wir freuen uns über eine Erwähnung, verlangen sie aber nicht. Eine Erwähnung enthält üblicherweise Titel, Autor, Verlag und ISBN, zum Beispiel: »*Generative KI-Systeme entwickeln* von Chip Huyen (O'Reilly). Copyright 2026 Rheinwerk Verlag GmbH, ISBN 978-3-96009-276-6.«

Falls Sie befürchten, zu viele Codebeispiele zu verwenden oder die oben genannten Befugnisse zu überschreiten, kontaktieren Sie uns unter *kommentar@oreilly.de*.

Danksagung

Dieses Buch hätte in seiner Entstehung viel länger gebraucht, und viele wichtige Themen würden fehlen, wenn mir nicht so viele wunderbare Menschen dabei geholfen hätten.

Weil der Zeitrahmen für das Projekt knapp war – zwei Jahre für ein Buch mit 150.000 Wörtern, das so viele Themen abdeckt –, bin ich den technischen Reviewern dankbar, die sich ihre wertvolle Zeit genommen haben, um dieses Buch so schnell zu begutachten.

Luke Metz ist ein toller Gesprächspartner, der meine Annahmen überprüft und mich davon abgehalten hat, den falschen Weg einzuschlagen. Han-Chung Lee –

immer auf dem neuesten Stand bei KI und der Community-Entwicklung – wies mich auf Ressourcen hin, die ich übersehen hatte. Luke und Han waren die ersten, die meine Entwürfe zu sehen bekamen, bevor ich sie an die nächste Runde technischer Reviewer schickte, und ich stehe für immer in ihrer Schuld, weil sie meine Irrungen und Wirrungen ausgehalten haben.

Vittorio Cretella und Andrei Lopatenko haben schon KI-Innovationen in Fortune-500-Unternehmen geleitet und konnten mir sehr wertvolles Feedback liefern, das tiefes technisches Wissen und Einblicke aus der Führungsebene kombinierte. Vicki Reyzelman hat mir dabei geholfen, meine Inhalte zu erden und sie für Lesende mit Softwareentwicklungshintergrund relevant zu halten.

Eugene Yan, ein guter Freund und ausgezeichneter angewandter Forscher, unterstützte mich technisch und emotional. Shawn Wang (swyx) lieferte einen wichtigen Stimmungscheck, der mir dabei half, mehr Vertrauen in mein Buch zu haben. Sanyam Bhutani, einer der besten Lernenden und einer der bescheidensten Menschen, die ich kenne, gab mir nicht nur sorgsam geschriebenes schriftliches Feedback, er nahm auch Videos auf, um sein Feedback zu erläutern.

Kyle Kranen ist ein ausgezeichneter Tutor, der seine Kollegen interviewt hat und mir eine wunderbare Zusammenfassung ihres Optimierungsprozesses zukommen ließ, die einen Leitfaden für das Optimierungskapitel lieferte. Mark Saroufim, ein neugieriger Geist, der immer seine Finger am Puls der interessantesten Probleme hat, lieferte mir tolle Ressourcen zur Effizienz. Sowohl Kyles als auch Marks Feedback waren entscheidend für die Kapitel 7 und 9.

Kittipat »Bot« Kampa hat nicht nur viele meiner Fragen beantwortet, sondern mir auch detailliert visualisiert, wie er über KI-Plattformen denkt. Ich freue mich über Denys Linkovs systematischen Ansatz zur Evaluation und zur Plattformentwicklung. Chetan Tekur lieferte tolle Beispiele, die mir dabei halfen, KI-Anwendungs-Patterns zu strukturieren. Auch möchte ich Shengzhi (Alex) Li und Hien Luu für ihr durchdachtes Feedback zu meinem Entwurf zur KI-Architektur danken.

Aileen Bui ist ein Schatz – sie hat wunderbares Feedback und Beispiele aus Sicht einer Produktmanagerin beigetragen. Todor Markov gab mir einen direkt umsetzbaren Rat für das Kapitel zu RAG und Agenten. Und Tal Kachman ist in letzter Minute eingesprungen, um das Kapitel zu Optimierungen ins Ziel zu bringen.

Es gibt so viele wunderbare Menschen, deren Gesellschaft und Gespräche mir Ideen für den Inhalt dieses Buchs lieferten. Ich habe mein Bestes versucht, alle Namen mit aufzunehmen, aber aufgrund der inhärenten Fehlerhaftigkeit des menschlichen Gehirns sind zweifelsohne viele nicht aufgeführt. Wenn ich vergessen habe, einen Namen zu erwähnen, dann liegt das nicht daran, dass ich den Beitrag nicht geschätzt habe. Bringen Sie sich bitte in Erinnerung, sodass ich das so schnell wie möglich nachholen kann!

Andrew Francis, Anish Nag, Anthony Galczak, Anton Bacaj, Balázs Galambosi, Charles Frye, Charles Packer, Chris Brousseau, Eric Hartford, Goku Mohandas, Hamel Husain, Harpreet Sahota, Hassan El Mghari, Huu Nguyen, Jeremy Howard,

Jesse Silver, John Cook, Juan Pablo Bottaro, Kyle Gallatin, Lance Martin, Lucio Dery, Matt Ross, Maxime Labonne, Miles Brundage, Nathan Lambert, Omar Khattab, Phong Nguyen, Purnendu Mukherjee, Sam Reiswig, Sebastian Raschka, Shahul ES, Sharif Shameem, Soumith Chintala, Teknium, Tim Dettmers, Undi95, Val Andrei Fajardo, Vern Liang, Victor Sanh, Wing Lian, Xiquan Cui, Ying Sheng und Kristofer.

Ich möchte auch all den Leserinnen und Lesern früher Versionen des Manuskripts danken, die mir Feedback gegeben haben. Douglas Bailley ist ein Super-Leser, der so viel wertvolles Feedback lieferte. Dank auch an Nutan Sahoo, die einen eleganten Weg für das Erläutern von Perplexität vorgeschlagen hat.

Ich habe aus den Onlinediskussionen mit so vielen Menschen sehr viel gelernt. Vielen Dank an alle, die meine Fragen beantwortet, meine Posts kommentiert oder mir eine E-Mail mit ihren Gedanken geschickt haben.

Natürlich hätte sich das Buch nicht ohne das Team von O'Reilly umsetzen lassen, insbesondere mit meinen Development Editors (Melissa Potter, Corbin Collins, Jill Leonard) und meinem Production Editor (Elizabeth Kelly). Liz Wheeler ist die scharfsichtigste Korrektorin, mit der ich je zusammengearbeitet habe. Nicole Butterfield war die Kraft, die das Buch von der Idee bis zum finalen Produkt begleitet hat.

Dieses Buch ist letztendlich eine Kumulation unschätzbarer Lektionen, die ich im Laufe meiner Karriere gelernt habe. Diese Lektionen verdanke ich meinen außerordentlich kompetenten und geduldigen aktuellen und ehemaligen Kolleginnen und Kollegen. Von jeder Person, mit der ich zusammengearbeitet habe, lernte ich etwas Neues über ML und dessen Einsatz.