

Praxisbuch Large Language Models

Sprache mit KI verarbeiten und generieren

DAS INHALTS- VERZEICHNIS

» Hier geht's
direkt
zum Buch

Vorwort	15
<hr/>	
Teil I: Die Funktionsweise von Sprachmodellen verstehen	25
1 Einführung in Large Language Models	27
Was ist Language AI?	28
Die jüngsten Entwicklungen im Bereich der Language AI	29
Darstellung von Sprache als ein Bag-of-Words	30
Verbesserte Vektordarstellungen mit Dense Vector Embeddings ...	32
Arten von Embeddings	34
Kontext mit Attention codieren und decodieren	35
Attention Is All You Need	38
Representation-Modelle – rein Encoder-basierte Modelle	
(»Encoder-only«)	41
Generative Modelle – rein Decoder-basierte Modelle	
(»Decoder-only«)	44
Das Jahr der generativen KI	46
Die sich wandelnde Definition von Large Language Models	48
Wie sich das Training von Large Language Models im Vergleich zu	
traditionellen Ansätzen unterscheidet	48
Anwendungsmöglichkeiten und Nutzen von Large Language Models ...	50
LLMs verantwortungsvoll entwickeln und nutzen	51
Limited Resources Are All You Need – LLMs auch ohne große	
Rechenressourcen trainieren und verwenden	52
Schnittstellen zur Nutzung von LLMs	53
Proprietäre, nicht frei zugängliche Modelle	53
Frei zugängliche Modelle	54
Open-Source-Frameworks	55
Ihren ersten Text mit einem LLM generieren	56
Zusammenfassung	59

2	Tokens und Embeddings	61
	Tokenisierung bei LLMs	62
	Wie Tokenizer die Eingaben für das Sprachmodell aufbereiten	62
	LLMs herunterladen und ausführen	63
	Welche Faktoren sind bei der Tokenisierung entscheidend?	67
	Tokenisierung auf Wort-, Wortteil-, Zeichen- und Byte-Ebene	68
	Ein Vergleich verschiedener trainierter Tokenizer	70
	Faktoren, die darüber entscheiden, wie sich ein Tokenizer verhält ...	78
	Token-Embeddings	80
	Sprachmodelle halten Embeddings für das Vokabular ihres	
	Tokenizer vor	81
	Kontextualisierte Word-Embeddings mit Sprachmodellen	
	erstellen	82
	Text-Embeddings (für Sätze oder ganze Dokumente)	85
	Wie Word-Embeddings jenseits von LLMs genutzt werden können	86
	Vortrainierte Word-Embeddings nutzen	86
	Der Word2vec-Algorithmus und Training mittels Contrastive	
	Learning	87
	Empfehlungssysteme aufbauen, die auf Embeddings basieren	91
	Songs mithilfe von Embeddings empfehlen	91
	Embedding-Modelle zur Empfehlung von Songs trainieren	92
	Zusammenfassung	94
3	Ein Blick ins Innere von Large Language Models	97
	Ein erster Überblick über Transformer-Modelle	98
	Die Ein- und Ausgaben eines Transformer-basierten LLM	98
	Die einzelnen Komponenten, die beim Forward-Pass durchlaufen	
	werden	100
	Auswahl eines einzelnen Tokens anhand einer	
	Wahrscheinlichkeitsverteilung (Sampling/Decodierung)	103
	Parallele Verarbeitung von Tokens und Kontextlänge	105
	Schnellere Generierung durch Zwischenspeichern von Schlüsseln	
	und Werten	107
	Ein Blick ins Innere des Transformer-Blocks	109
	Verbesserungen an der Transformer-Architektur	118
	Effizienterer Attention-Mechanismus	118
	Der Transformer-Block	122
	Positional-Embeddings (RoPE)	124
	Weitere Vorschläge und Verbesserungen im Hinblick auf die	
	Architektur	127
	Zusammenfassung	127

Teil II: Vortrainierte Sprachmodelle verwenden	129
4 Textklassifikation	131
Sentimentanalyse von Spielfilmrezensionen	132
Texte mit Representation-Modellen klassifizieren	133
Ein geeignetes Modell auswählen	135
Ein aufgabenspezifisches Modell verwenden	136
Texte mit Embedding-Modellen klassifizieren	140
Überwachte Klassifikation	141
Was aber, wenn Ihnen keine gelabelten Daten zur Verfügung stehen?	143
Texte mit generativen Modellen klassifizieren	147
Texte mit T5-(Text-to-Text-Transfer-Transformer-)Modellen klassifizieren	148
Texte mit ChatGPT klassifizieren	152
Zusammenfassung	155
5 Clustering von Texten und Topic Modeling	157
ArXiv-Artikel aus dem Forschungsbereich Computation and Language	158
Die bewährte Pipeline beim Text-Clustering	159
Dokumente in Embeddings umwandeln	159
Die Dimensionierung der Embeddings verringern	160
Die dimensionsreduzierten Embeddings zu Clustern zusammenfassen	163
Die gebildeten Cluster inspizieren	164
Vom Clustern von Texten hin zum Topic Modeling	167
BERTopic: ein modulares Topic-Modeling-Framework	168
Der modulare Aufbau des BERTopic-Frameworks	172
Einen besonderen Baustein hinzufügen	176
Ein zusätzlicher Baustein zur Textgenerierung	181
Zusammenfassung	185
6 Prompt Engineering	187
Textgenerierungsmodelle verwenden	187
Ein geeignetes Textgenerierungsmodell wählen	187
Textgenerierungsmodelle laden	188
Einfluss auf die Ausgabe eines Modells nehmen	190
Einführung in das Prompt Engineering	193
Die grundlegenden Elemente eines Prompts	194
Prompts formulieren, die Anweisungen enthalten (Instruction-based Prompting)	196

Fortgeschrittene Prompt-Engineering-Techniken	198
Prompts komplexer gestalten	198
In-Context Learning – Beispiele bereitstellen	201
Prompt Chaining – Aufgaben in mehrere Teilaspekte aufteilen	203
Logisches Schließen mit generativen Modellen (Reasoning)	205
Chain-of-Thought – erst nachdenken, dann antworten	206
Self-Consistency – Auswahl aus mehreren Antwortmöglichkeiten	209
Tree-of-Thought – über Zwischenschritte zur besten Antwort gelangen	210
Modellausgaben validieren	213
Beispiele bereitstellen	214
Vorgaben machen und Modellausgaben beschränken	216
Zusammenfassung	220
7 Fortgeschrittene Techniken und Tools im Bereich der Textgenerierung	221
Optimierungen hinsichtlich der Verwendung von Modellen – quantisierte Modelle mit dem LangChain-Framework laden	222
Chains – die Anwendungsmöglichkeiten von LLMs noch erweitern	225
Prompt-Templates mit einem LLM verketteten	225
Mehrere Prompts miteinander verketteten	228
Einen Speicher bereitstellen – LLMs ermöglichen, sich an Gespräche zu erinnern	231
Conversation Buffer – LLMs den gesamten Gesprächsverlauf bereitstellen	233
Windowed Conversation Buffer – LLMs einen Teil des Gesprächsverlaufs bereitstellen	235
Conversation Summary – LLMs eine Zusammenfassung des Gesprächsverlaufs bereitstellen	236
Agenten – ein aus mehreren LLMs bestehendes System entwickeln	240
Die treibende Kraft hinter Agenten – Schritt-für-Schritt- Reasoning	242
ReAct im LangChain-Framework verwenden	244
Zusammenfassung	247
8 Semantische Suche und Retrieval-Augmented Generation	249
Einführung in semantische Such- und RAG-Systeme	250
Semantische Suche auf Basis von LLMs	252
Dense-Retrieval-Systeme	252
Reranking-Systeme	264
Metriken zur Evaluierung von Retrieval-Systemen	268
Retrieval-Augmented Generation (RAG)	273
Suchsysteme zu RAG-Systemen erweitern	274
Ein Beispiel für eine auf Fakten basierende Generierung (Grounded Generation) mit einem gemanagten LLM	276

Ein Beispiel für ein RAG-System, bei dem das Modell lokal betrieben wird	276
Fortgeschrittene Techniken im Bereich der RAG-Systeme	279
RAG-Systeme evaluieren	281
Zusammenfassung	282
9 Multimodale Large Language Models	283
Vision Transformer	284
Multimodale Embedding-Modelle	287
CLIP – Modelle, die eine Verbindung zwischen Texten und Bildern herstellen können	289
Wie werden bei CLIP multimodale Embeddings generiert?	289
OpenCLIP	292
CLIP mit der sentence-transformers-Bibliothek laden	296
Multimodale Textgenerierungsmodelle erstellen	296
BLIP-2 – Modelle erstellen, die auf Basis von Texten und Bildern logische Schlüsse ziehen können	297
Multimodale Eingabedaten aufbereiten	301
1. Anwendungsfall: Bildbeschriftungen erstellen	304
2. Anwendungsfall: Chatmodelle erstellen, die multimodale Prompts unterstützen	306
Zusammenfassung	309
<hr/>	
Teil III: Sprachmodelle trainieren und feintunen	311
10 Text-Embedding-Modelle erstellen	313
Embedding-Modelle	313
Was genau ist Contrastive Learning?	316
SBERT	318
Embedding-Modelle erstellen	320
Kontrastive Beispiele erstellen	321
Das Modell trainieren	322
Differenziertere Evaluierung	325
Verlustfunktionen	326
Embedding-Modelle feintunen	334
Embedding-Modelle mit gelabelten Datensätzen feintunen	335
Augmented SBERT	337
Embedding-Modelle mit ungelabelten Daten feintunen	342
Transformer-based Sequential Denoising Auto-Encoder (TSDAE) ...	342
TSDAE zur Domain Adaptation nutzen	346
Zusammenfassung	347

11 Representation-Modelle für die Klassifikation feintunen	349
Klassifikation mit gelabelten Daten	349
Ein vortrainiertes BERT-Modell feintunen	351
Schichten eines Modells einfrieren	354
Few-Shot-Klassifikation	359
SetFit – Modelle mit nur wenigen Trainingsbeispielen auf effiziente Weise feintunen	359
Modelle für die Few-Shot-Klassifikation feintunen	363
Bereits vortrainierte Modelle mittels Masked Language Modeling weiter vortrainieren	366
Named-Entity Recognition	371
Daten für die Named-Entity Recognition aufbereiten	373
Modelle für die Named-Entity Recognition feintunen	378
Zusammenfassung	379
12 Generative Modelle feintunen	381
Die drei Schritte beim Training von LLMs – Pretraining, Supervised Fine-Tuning und Preference Tuning	381
Supervised Fine-Tuning (SFT)	383
Vollständiges Feintuning (Full Fine-Tuning)	384
Parameter-efficient Fine-Tuning (PEFT)	385
Instruction Tuning mit dem QLoRA-Verfahren	393
Instruktionsdaten über ein Template bereitstellen	394
Modelle quantisieren	395
Die Konfiguration für das LoRA-Verfahren festlegen	396
Die Parameter für das Training festlegen	397
Das Modell trainieren	398
Gewichte des QLoRa-Modells mit denen des Basismodells zusammenführen	399
Generative Modelle evaluieren	400
Metriken auf Wortebene	400
Benchmarks	401
Leaderboards	402
Automatisierte Evaluierung	403
Evaluierung durch Menschen	403
Modelle darauf ausrichten, was Nutzer erwarten – Preference Tuning, Alignment und Reinforcement Learning from Human Feedback	405
Die Preference Evaluation mithilfe von Reward-Modellen automatisieren	407
Die Ein- und Ausgaben des Reward-Modells	407
Ein Reward-Modell trainieren	408
Preference Tuning ohne zusätzliches Training eines Reward-Modells	411

Preference Tuning mittels DPO	414
Templates für Präferenzdatensätze erstellen	414
Das Modell quantisieren	415
Parameter für das Training festlegen	416
Das Modell trainieren	416
Zusammenfassung	417
Schlussbemerkung	419
Index	421