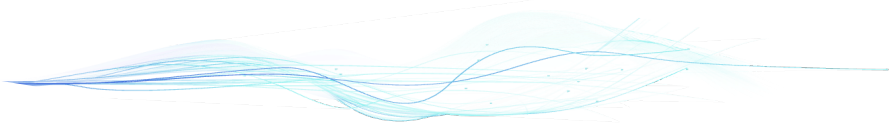
CONTECO

Workshop



Generative KI Masterclass: Ein eigenes LLM bauen und trainieren

Generative KI Masterclass: Ein eigenes LLM bauen und trainieren

Kursdauer
5 Tage

Zielgruppe
Erfahrene Entwickler:innen, Data Scientists und Machine-Learning-Expert:innen, die **tiefer in die Architektur und das Training großer Sprachmodelle** eintauchen möchten. Der Fokus liegt auf **Modellarchitektur, Datenauswahl, Tokenisierung, Training, Evaluierung und Optimierung**.

Schulungsziel
Diese Masterclass vermittelt ein tiefgehendes Verständnis darüber, **wie LLMs aufgebaut, trainiert und optimiert werden**. Die Teilnehmenden lernen, **eigene Transformer-Modelle zu entwickeln**, die richtigen **Datensätze aufzubereiten**, Tokenizer zu trainieren und den gesamten Trainingsprozess von LLMs nachzuvollziehen. Neben dem Training von Modellen wird auch **Feintuning, Evaluierung und Optimierung** behandelt. Nach dem Kurs sind die Teilnehmenden in der Lage, ein **eigenes kleines LLM zu trainieren, es zu evaluieren und in Anwendungen zu integrieren**.

Schulungsbeschreibung
Diese intensive **fünf**-tägige Schulung gibt einen **Deep Dive in die Architektur und das Training großer Sprachmodelle**. Der erste Tag konzentriert sich auf die **interne Struktur von LLMs**, darunter Transformer, Self-Attention, Embeddings und typische Modellarchitekturen (z. B. GPT, T5). Zudem werden **Datenauswahl, Copyright-Herausforderungen und Tokenisierung** behandelt.

Im weiteren Verlauf wird gezeigt, **wie ein eigenes Modell mit PyTorch oder TensorFlow implementiert und trainiert** wird. Dabei werden wichtige Aspekte wie **Batchgrößen, Optimizer (AdamW), GPU-Skalierung und Mixed Precision** (FP16/BF16) thematisiert. Die Teilnehmenden setzen ein eigenes **Trainingsskript für ein kleines Modell auf und experimentieren mit verschiedenen Hyperparametern**.

An Tag 3 liegt der Fokus auf **Feintuning und Evaluierung**, inklusive Methoden zur **Bewertung der Modellqualität (Perplexity, BLEU, ROUGE)** und **Retrieval-Augmented Generation (RAG)**. Es wird demonstriert, wie man ein **LLM mit Vektordatenbanken kombiniert**, um **eigene Daten als Wissensbasis** einzubinden.

Am vierten Tag werden **fortgeschrittene Optimierungsstrategien** behandelt, darunter **LoRA-Fine-Tuning, Parameterreduktion und Modellkompression**. Zudem gibt es eine Einführung in **multimodale KI-Modelle**, z. B. LLMs mit Bildverarbeitung (CLIP, Vision Transformer).

Der letzte Tag widmet sich der **Bereitstellung und Skalierung** von LLMs. Es wird gezeigt, **wie ein LLM produktionsreif gemacht wird**, mit Frameworks wie **FastAPI, Triton Inference Server oder ONNX**. Im **Abschlussprojekt** setzen die Teilnehmenden eine **eigene LLM-basierte Anwendung um**, etwa eine kleine GPT-Variante, ein RAG-System oder eine multimodale Demo.

Wer teilnehmen sollte

Dieser Kurs richtet sich an **erfahrene Entwickler:innen und Data Scientists**, die **selbst LLMs trainieren oder anpassen möchten**. Voraussetzungen sind **fundierte Kenntnisse in Python, Machine Learning (PyTorch/TensorFlow) sowie ein grundlegendes Verständnis von Deep Learning-Architekturen**. Ideal für **Forschende, ML-Ingenieur:innen oder KI-Enthusiast:innen**, die eigene Modelle entwickeln oder tiefgehendes Wissen über Sprachmodelle erlangen möchten.

Tag 1: Deep Dive – Architektur & Daten

1. **Eigene LLM-Architektur**
   - Transformer intern (Self-Attention, Multi-Head, Position Embeddings)
   - GPT-Style Decoder vs. Encoder-Decoder (T5, BART)
2. **Datenauswahl & Aufbereitung**
   - Öffentliche Datensätze (The Pile, Books3, OpenWebText2)
   - Copyright & Cleanup (Duplikate, Filters)
   - geschlossene/unternehmensinterne Daten (Datenschutz)
3. **Tokenisierung**
   - Byte Pair Encoding (BPE), SentencePiece, tiktoken
   - Vocabulary-Größe, Trade-offs
4. **Hands-on**:
   - Erstes Skript: Tokenizer-Training auf einem Mini-Korpus
   - Großer Korpus? Wie man ihn vorbereitet (Chunking, Sharding, etc.)

Tag 2: Implementierung & Training-Pipeline

1. **Implementierung eines Mini-Transformers**
   - PyTorch oder TensorFlow: Layer (Attention, Feedforward), GPTBlock / DecoderLayer
   - Konfigurationsparameter (Hidden Dim, Heads, Depth)
2. **Training-Setup**
   - GPU/TPU-Einrichtung (lokal vs. Cloud)
   - Batch Size, Lernrate, Optimizer (AdamW)
   - Mixed Precision (FP16/BF16)
3. **MLOps-Tools**
   - MLflow, ClearML oder Weights & Biases zum Tracken von Experimenten
4. **Hands-on**:
   - Starte ein **Trainingsskript** für einen kleinen Datensatz (z. B. 50–100 MB Text)
   - Protokolliere Training Loss, evaluiere gelegentliche Samples

Tag 3: Feintuning, Evaluierung & RAG

1. **Weiterführendes Training**
    - Feintuning vs. Training-from-scratch
    - RLHF (Reinforcement Learning from Human Feedback) – theoretischer Überblick
2. **Evaluierung**
    - Perplexity, Token-Accuracy
    - Generische Metriken: BLEU, ROUGE, etc.
3. **Retrieval-Augmented Generation**
    - Eigene RAG-Pipeline auf selbst trainiertem Modell?
    - Einbinden einer Vektordatenbank (Chroma, FAISS)
4. **Hands-on**:
    - Feinjustiere dein Modell auf einen **speziellen Datensatz** (z. B. interne Doku)
    - Baue eine Mini-RAG-Demo (prompt + fetch + generation)

Tag 4: Fortgeschrittene Optimierung & Multimodale Erweiterungen

1. **Fortgeschrittene Optimierung**
    - Gradient Accumulation, ZeRO, LoRA (Low-Rank Adaptation)
    - Parameter-Effiziente Methoden (QLoRA, 4-Bit/8-Bit Training)
    - Modellkompression, Distillation
2. **Multimodale Szenarien**
    - Text/Bild (CLIP-Ansätze, Vision Transformer-Encoder + GPT-Decoder)
    - Kurzer Abstecher: Diffusion Models integrieren (für Image Generation)
3. **Hands-on**:
    - Probier LoRA-Fine-Tuning oder 4-Bit-Quantisierung, um GPU-Speicher zu sparen
    - Experimentiere mit kleiner Bild/Text-Integration (z. B. Bild-Captions)

Tag 5: Production-Ready & Abschlussprojekt

1. **Deployment & Skalierung**
    - Serving-Frameworks (FastAPI, Triton Inference Server)
    - Monitoring & Logging
    - Parallel-LLM-Setups (Fallback, Ensembles)
2. **Sicherheit & Governance**
    - Prompt Injection-Verhinderung, Content Filtering
    - Datenschutz und On-Prem-Lösungen
3. **Abschlussprojekt**
    - Wähle:
        1. **Eigener GPT-Klon** (kleines Modell)
        2. **RAG-System** mit selbst trainiertem Modell
        3. **Multimodaler Mini-Prototyp** (Text + Bild)
    - Stelle Ergebnisse vor, diskutiere Performance und Limitierungen
4. **Ausblick**
    - Größere Modelle (Llama, Bloom)
    - HPC-Anforderungen, verteiltes Training
    - Kommerzielle vs. Open-Source-Varianten