Ann Arbor Algorithms 2024 AI Bootcamp

Large Language Models Outline

Instructor: Wei Dong PhD, wdong@aaalgo.com

Github: https://github.com/aaalgo/AI_training_2024

AAALGO provides training and consulting services to corporations and organizations. Please contact Dr. Dong for inquiries.

Module 1 Introduction

- Basic Concepts
  - Context window
  - Model size (billion of parameters)
  - Model flavors (basic vs instruct)
  - License and restrictions, open-weights models
- LLM Life Cycle
  - Pre-train
    - Case Study: Llama 3.1 training setup
    - Open large language datasets
  - Finetune
    - Upstream vs downstream
    - Instruct finetune vs RLHF
  - Case Study: Winning Solution of 2024 AI Math Olympiad
    - Deepseek Coder
    - Deepseek math
    - Two Stage finetune of Numina's winning solution
- LLM Inference
  - Causal language models
  - Tokenizers and Byte Pair Encoding (BPE)
  - Logits, softmax and temperature
  - Generation strategy
    - Greedy
    - Sampling
    - Beam search
- Lab 1
  - https://github.com/aaalgo/AI_training_2024
  - OpenAI API

        - Basic
        - Parameters
        - Tool usage
    - Transformers

Module 2 Evaluation

- Importance of benchmarking
    - Quantitative evaluation of what LLMs can do and how intelligent they are.
    - How to start your own research.
- Natural language understanding (before ChatGPT)
    - CoLA: grammatical acceptance
    - SST2: sentiment analysis
    - MRPC: semantic equivalence
    - MNLI: natural language reasoning
    - GLUE
    - HellaSwap: natural language reasoning
- General AI Benchmarks (after ChatGPT)
    - IFEval: instruction following
    - MMLU: domain expertise
    - GPQA: difficult domain expertise
    - HumanEval: coding
    - BMPP: coding
    - GSM8K: math
    - MiniF2F and ProofNet: math proving
    - ARC: reasoning
    - BFCL: tool utilization
    - Text summarization

Module 3 Prompt Engineering

- Prompt Engineering
    - Overview
    - Few shot learning and in-context learning
    - CoT: Chain-of-Thought
    - Zero-Shot-CoT
    - Self-consistency
    - Auto-CoT
    - Three-of-Though
    - Graph-of-Though
    - ReACT
    - Emotional Prompting

- Lab 2
    - Prompt-engineering in action (llama-cpp-python)
- Hallucination
- RAG: Retrieval Augmented Generation
    - The RAG prompt template
    - RAG framework
    - RAG paradigms
    - RAG techniques
        - Query rewrite
        - Reranking
        - Summarization
    - Text retrieval with vector DB
    - Chunking
    - Embedding
    - The ANN benchmark
    - Mainstream vector DBs
    - Text search engine, Lucene and ElasticSearch

Module 4 LLM Deployment

- Overview of industrial landscape
- Proprietary API tiering and pricing
    - OpenAI
    - Google/Gemini
    - Anthropic/Claude
    - Mistral
- Anthropic and Mistral AI
- API Endpoint Pricing of Open-Weight Models
- How much is 1 million tokens
- The target market of Llama 3.1 8B, 70B and 405B models
- Estimated cost of hosting Llama 3.1 on AWS and Google Vertex AI
- Bits and Bytes – from # parameters to gigabytes
    - 16-bit, 8-bit, 4-bit
- IEEE754, float16 vs bfloat16
- Basic quantization
- GPTQ
- Brief history of NVidia
- NVidia microarchitectures
- Nvidia compute capabilities
- TFLOPS and bfloat16
- Nvidia GPU tiers. Memory size, TFLOPS and Price
    - Gaming – 4090, etc.

    - Workstations – RTX A6000, RTX 6000 Ada Generation
    - Data Center – A100, H100, L4/L40
- Sample whole system configurations
- GPU power consumption

Module 5. Architecture of Llama3

- Roadmap to understanding Llama3
- Basic linear algebra: projection and inner product
- Linear layer
- Activation functions, ReLU, SiLU
- Review of text retrieval with vector DB
- Basics of attention: Query Key and Value
- Multi-head attention
- Grouped query attention
- Llama3 attention code
- LlamaMLP: denoising with projection
- Llama3 model overview
- Llama3 decoder code
- Calculation of attention cost
- Page attention
- Lab 3
    - Logit magic and key-value cache
    - Extracting attention
- Project suggestions
- Closing remarks