

Running Local LLMs

Gem City Machine Learning

Nick Gerakines, 4/10/2024

Disclaimer: This presentation is provided “as is”, without warranty of any kind, express or implied. All information, content, and material are for general informational purposes only.

I'm Nick
(he/him)

I'm a software engineer in
the artificial intelligence,
machine learning, and
cognitive science space.

<https://ngerakines.me/>





Microsoft



What are LLMs?

What are they good for?

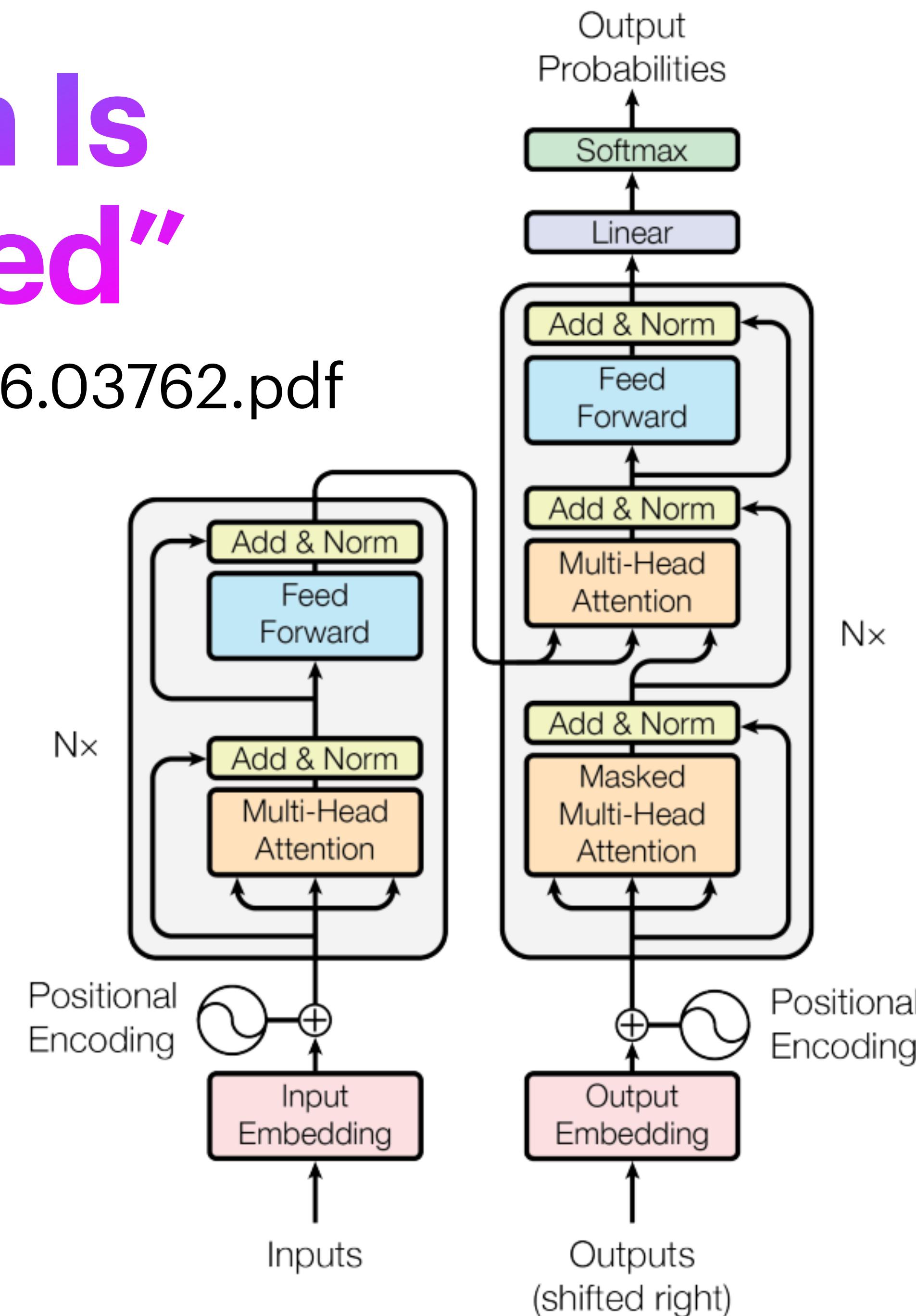
- Sentiment analysis
- Large data research
- Customer service
- Chat bots
- Online search and information retrieval
- General knowledge augmentation

But how do they work?

Transformers: Neural Networks

“Attention Is All You Need”

<https://arxiv.org/pdf/1706.03762.pdf>



**LLM internals are
complicated.**

LLM intervals are
ANOTHER DAY complicated.

A Brief History Of Time

- 2015
 - OpenAI Founded
- 2016
 - Stanford SQuAD (Stanford Question Answering Dataset) released
- 2017
 - “Attention is all you need” paper published
 - Tensor Flow Framework
 - USE (Universal Sentence Encoder)
- 2018
 - OpenAI GPT paper published
 - Google releases BERT
- 2019
 - OpenAI releases GPT-2
 - Microsoft releases LXMERT + invests \$1 billion into OpenAI
 - Facebook releases RoBERTa
- 2020
 - OpenAI releases GPT-3
- 2021
 - OpenAI DALL-E
 - Google Transformer-X
 - Facebook XLM-R
 - Google LaMDA
 - Github Copilot
 - OpenAI Codex
- 2022
 - OpenAI InstructGPT
- 2023
 - Hugging Face BLOOM
 - Google Woodcraft
 - OpenAI Chat GPT
- 2024
 - Microsoft \$10 billion into OpenAI
 - Google BARD
 - Meta LLaMA
 - OpenAI GPT-4
 - Nvidia NEMO
 - Amazon Bedrock
 - Anthropic Claude 2
 - Meta LLaMA 2
- 2025
 - **Too Many To List**

Models

(The thing people are actually talking about.)

- GPT-4 (OpenAI)
- LLaMA (Meta)
- Bloom (BigScience)
- Gemma / Gemini (Google)
- Claude (Anthropic)
- Mistral / Mixtral (mistral.ai)

More: <https://github.com/Hannibal046/Awesome-LLM>

How do we actually use them?

ollama

<https://ollama.com/>

<https://github.com/ggerganov/llama.cpp>



Blog

Discord

GitHub

Models

Sign in

Download



Get up and running with large language models, locally.

Run [Llama 2](#), [Code Llama](#), and other models.

Customize and create your own.

Download ↓

Available for macOS, Linux,
and Windows (preview)



[Blog](#)[Discord](#)[GitHub](#)[Models](#)[Sign in](#)[Download](#)

Download Ollama



macOS



Linux



Windows

[Download for macOS](#)

Requires macOS 11 Big Sur or later

While Ollama downloads, sign up to get
notified of new updates.

[Get updates](#)

[Blog](#)[Discord](#)[GitHub](#)[Models](#)[Sign in](#)[Download](#)

Models

[Featured](#)

gemma

Gemma is a family of lightweight, state-of-the-art open models built by Google DeepMind.

306.6K Pulls

69 Tags

Updated 5 days ago

llama2

Some of my favorites:

ollama run llama2

ollama run gemma

ollama run llava

ollama run mixtral

ollama run codellama

ollama run phi

But offline?

(demo)

Bonus

RAG

langchain

<https://python.langchain.com/>

The screenshot shows the LangChain website's 'Introduction' page. At the top, there's a navigation bar with links for Components, Integrations, Guides, API Reference, More, and a search bar. The main content area has a breadcrumb trail: Home > Get started. The title 'Introduction' is displayed prominently. Below it, a paragraph explains that LangChain is a framework for developing applications powered by large language models (LLMs). It highlights three stages: Development, Productionization, and Deployment. A bulleted list details these stages: 'Development: Build your applications using LangChain's open-source building blocks and components. Hit the ground running using third-party integrations and Templates.', 'Productionization: Use LangSmith to inspect, monitor and evaluate your chains, so that you can continuously optimize and deploy with confidence.', and 'Deployment: Turn any chain into an API with LangServe.'.

On the right side, there's a sidebar with various links: Get started, Use cases, Expression Language, Ecosystem (LangSmith, LangGraph, LangServe), Security, Additional resources (Components, Integrations, Guides, API reference), Debugging, Playground, Evaluation, Annotation, and Monitoring.

The central part of the page features a diagram illustrating the LangChain architecture. It shows a hierarchy of components:

- LangSmith** (Observability)
- LangServe** (Deployments) - labeled as 'Chains as Rest APIs' and supports Python.
- Templates** (Reference Applications) - supports Python.
- LangChain** (Cognitive Architectures)
 - LangChain-Community** (Integrations Components)
 - Model I/O: Model, Prompt, Example Selector, Output Parser
 - Retrieval: Retriever, Document Loader, Vector Store, Text Splitter, Embedding Model
 - Agent Tooling: Tool Toolkit
 - LangChain-Core** (Protocol)
 - LCEL - LangChain Expression Language
 - Parallelization, Fallbacks, Tracing, Batching, Streaming, Async, Composition

(demo)

Questions?