Large Language Models (LLMs): Complete Notes & Step-by-Step Study Plan

> **Goal:** Understand LLMs from fundamentals to advanced concepts, implementation, fine-tuning, evaluation, and real-world applications.

---

1. Introduction to Large Language Models

What is a Language Model?

A **language model** is a probabilistic model that predicts the next word/token given previous words.

Example:

> Input: "I am learning Data" Output: "Science"

Formally:

$$P(w_1, w_2, ..., w_n) = \prod P(w_n | w_1, ..., w_{n-1})$$

What Makes an LLM "Large"?

- Billions or trillions of parameters
- Trained on massive datasets (internet-scale text)
- Uses deep neural networks (Transformers)

Examples: - GPT-3 / GPT-4 / GPT-5 - LLaMA, Mistral - BERT, T5, PaLM

---

2. Prerequisites You Must Know

Mathematics

- Linear Algebra: vectors, matrices, dot product
- Probability: conditional probability, entropy
- Calculus: gradients, backpropagation (basic)

Programming

- Python (mandatory)
- NumPy, PyTorch or TensorFlow

NLP Basics

- Tokenization
- Stop words
- Bag of Words
- TF-IDF
- Word Embeddings (Word2Vec, GloVe)

---

3. Evolution of Language Models

Rule-Based Systems

- Hand-written grammar rules
- Not scalable

Statistical Models

- N-gram models
- Markov assumption
- Problem: data sparsity

Neural Language Models

- RNNs, LSTM, GRU
- Better context handling
- Problem: long-term dependencies

Transformer-Based Models (Breakthrough)

- Introduced in **2017**: *“Attention Is All You Need”*
- Solved long-context problem

---

4. Tokenization (Very Important)

Why Tokenization?

LLMs do not understand words, they understand **tokens**.

Types of Tokenization

1. Word Tokenization

- Simple
- Vocabulary explodes

2. Character Tokenization

- Small vocabulary
- Very long sequences

3. Subword Tokenization (Used in LLMs)

- BPE (Byte Pair Encoding)
- WordPiece
- SentencePiece

Example: "unbelievable" → un + believe + able

---

5. Embeddings

What are Embeddings?

Dense vector representations of tokens.

- Semantic meaning captured
- Similar words → similar vectors

Example: - king − man + woman ≈ queen

Types

- Token embeddings
- Positional embeddings

---

6. Transformer Architecture (CORE OF LLMs)

Why Transformer?

- Parallel processing
- Handles long-range dependencies

Main Components

1. Input Embedding Layer

- Token embedding + positional embedding

2. Self-Attention Mechanism

Key Idea: Each word attends to every other word.

Attention formula:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Where: - Q = Query - K = Key - V = Value

3. Multi-Head Attention

- Multiple attention heads
- Capture different relationships

4. Feed Forward Neural Network

- Fully connected layers

5. Residual Connections & Layer Normalization

- Prevent vanishing gradients

---

7. Types of Transformer Models

Encoder-Only

- BERT
- Used for classification, sentiment analysis

Decoder-Only

- GPT series
- Used for text generation

Encoder–Decoder

- T5
- Translation, summarization

---

8. Training an LLM

Pretraining Objective

Causal Language Modeling (GPT)

- Predict next token

Masked Language Modeling (BERT)

- Predict masked words

Dataset

- Web pages
- Books
- Code
- Wikipedia

Training Process

1. Tokenize text
2. Convert tokens to embeddings
3. Forward pass through transformer
4. Calculate loss (Cross-Entropy)
5. Backpropagation
6. Update parameters

---

9. Fine-Tuning

Why Fine-Tuning?

- Adapt model to specific task/domain

Types

Supervised Fine-Tuning (SFT)

- Input → Output pairs

Instruction Tuning

- Makes model follow instructions

Parameter Efficient Fine-Tuning (PEFT)

- LoRA
- Adapters
- Prefix tuning

---

10. Reinforcement Learning from Human Feedback (RLHF)

Steps

1. Pretrained model
2. Supervised fine-tuning
3. Train reward model
4. Reinforcement learning (PPO)

Used in ChatGPT-style models.

---

11. Inference & Decoding Strategies

Greedy Search

- Picks highest probability token

Beam Search

- Explores multiple paths

Sampling

- Temperature
- Top-k
- Top-p (nucleus sampling)

---

12. Evaluation of LLMs

Metrics

- Perplexity
- BLEU, ROUGE
- Human evaluation

Benchmarks

- GLUE
- MMLU
- HELM

---

13. Hallucinations & Limitations

Problems

- Hallucination
- Bias
- High compute cost
- Context length limits

Mitigation

- RAG (Retrieval Augmented Generation)
- Better prompts
- Fine-tuning

---

14. Retrieval Augmented Generation (RAG)

What is RAG?

LLM + External Knowledge Base

Steps

1. User query
2. Retrieve relevant documents
3. Inject into prompt
4. Generate answer

Used in chatbots, enterprise AI.

---

15. Prompt Engineering

Techniques

- Zero-shot prompting
- Few-shot prompting
- Chain of Thought
- Role prompting

---

16. Multimodal LLMs

- Text + Image + Audio
- Examples: GPT-4o, Gemini

17. LLM Deployment

Tools

- Hugging Face
- LangChain
- FastAPI
- Docker

Optimization

- Quantization
- Pruning
- Distillation

18. Ethics & Safety

- Bias
- Data privacy
- Alignment
- Responsible AI

90-Day Step-by-Step Study Plan

Month 1: Foundations

- NLP basics
- Tokenization
- Word embeddings
- Transformer theory

Month 2: Deep LLM Concepts

- Self-attention math
- Training & fine-tuning
- RLHF
- Prompt engineering

Month 3: Practical & Projects

- Fine-tune LLaMA
- Build RAG chatbot

- Deploy with FastAPI

Final Projects (Must Do)

1. Build your own mini-GPT
2. Domain-specific chatbot (Career / Medical / Legal)
3. LLM-based resume analyzer

If you want:

- Colab notebooks
- Interview Q&A
- Daily challenge-based plan
- LLM project ideas for resume

Just tell me 👍