# **Evolution of Large Language Models (LLMs)**

## **1. RNNs and LSTMs: The Foundations**

### **Recurrent Neural Networks (RNNs)**

**Overview**:

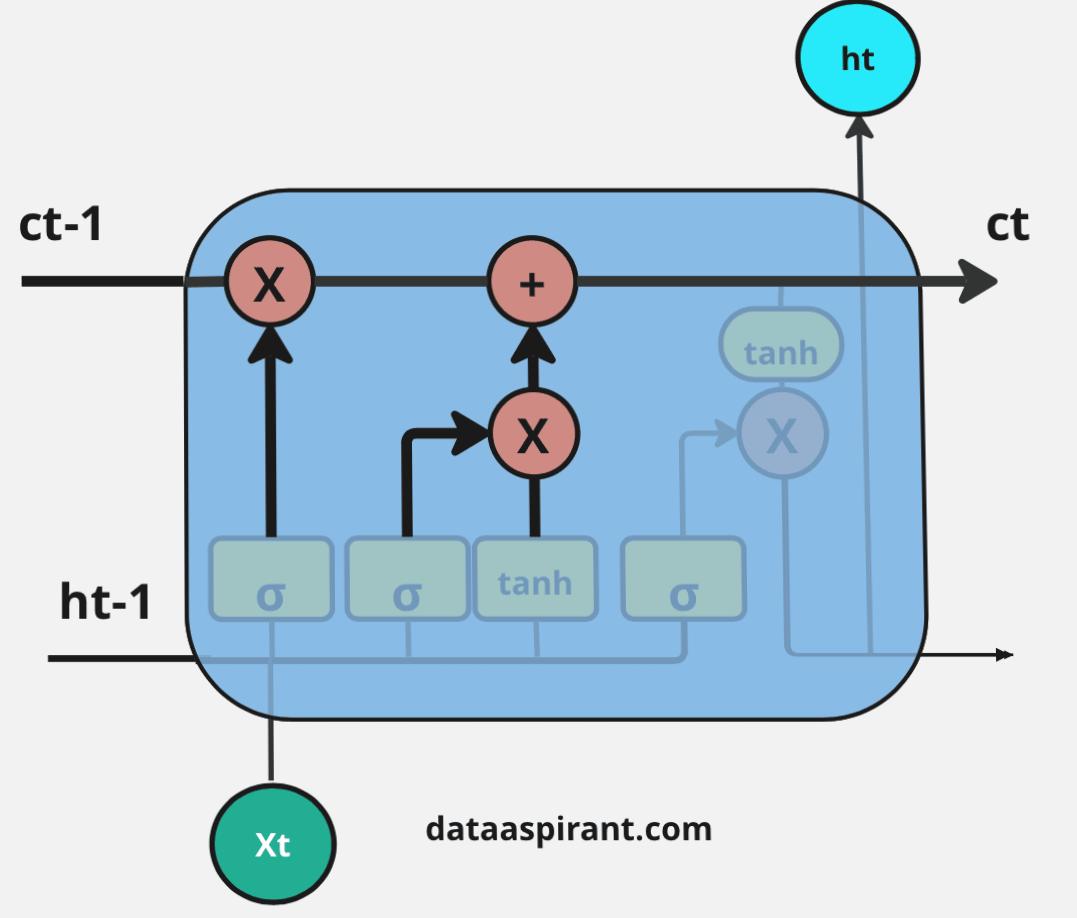
* RNNs are designed to handle **sequential data** like sentences, audio, or time series.
* They **reuse the same weights** across time steps and maintain a **hidden state** that carries contextual information.

#### **How RNNs Work**:

At each time step t, the hidden state ht is computed as:

ht = tanh(Wxh \* xt + Whh \* ht-1 + b)

* xt = current input
* ht-1 = previous hidden state
* Wxh, Whh = weight matrices
* b = bias term

**Diagram**: 

xt → [ RNN Cell ] → ht → [ RNN Cell ] → ht+1 → ...

↑ ↑

ht-1 ht

**Strengths**:

* Suitable for tasks like text generation, language modeling, and speech recognition.
* Captures short-term dependencies in sequences.

**Weaknesses**:

* **Vanishing/Exploding Gradients**: Difficult to learn long-range dependencies.
* **Sequential Computation**: Can’t be parallelized during training.
* **Context Limitation**: Cannot effectively remember information over long text spans.

### **Long Short-Term Memory Networks (LSTMs)**

LSTMs are a special kind of RNN, capable of learning **long-term dependencies** using a **gating mechanism**.

#### **Gates in LSTMs**:

1. **Forget Gate (ft)**: What info to discard
2. **Input Gate (it)**: What new info to add
3. **Output Gate (ot)**: What to output from the memory

#### **LSTM Equations**:

ft = σ(Wf · [ht-1, xt] + bf)

it = σ(Wi · [ht-1, xt] + bi)

ot = σ(Wo · [ht-1, xt] + bo)

Ct = ft \* Ct-1 + it \* tanh(Wc · [ht-1, xt] + bc)

ht = ot \* tanh(Ct)

**Architecture Diagram**:

ft it ot

/ / /

xt → [Input]→(Forget)→(Input)→(Output)→ ht

↓ ↓

Ct-1 tanh(Ct) → ht

**Strengths**:

* Retains long-term dependencies.
* Less prone to vanishing gradients.

**Weaknesses**:

* Complex architecture.
* Still **sequential**, hence slow to train on large datasets.

## **2. Why Transformers Replaced Them**

### **Limitations of RNNs/LSTMs**:

* Struggle with very long sequences.
* Hard to **parallelize**.
* Contextual memory is limited.
* Training is **slow and inefficient**.

### **Transformers: A Paradigm Shift**

**Published**: 2017 by Vaswani et al.

**Title**: "Attention Is All You Need"

**Key Innovation**: **Self-Attention** — a mechanism where each word in a sequence **attends** to every other word, capturing relationships regardless of their distance.

#### **Self-Attention Explained**:

For a sentence like:

“The animal didn’t cross the road because it was too tired.”

The word “it” refers to “animal”. Transformers **learn this relationship** through attention scores between “it” and “animal”.

### **Transformer Architecture**

#### **Core Components**:

1. **Input Embeddings + Positional Encoding**
2. **Multi-Head Self-Attention**
3. **Feed-Forward Networks**
4. **Layer Normalization + Residual Connections**

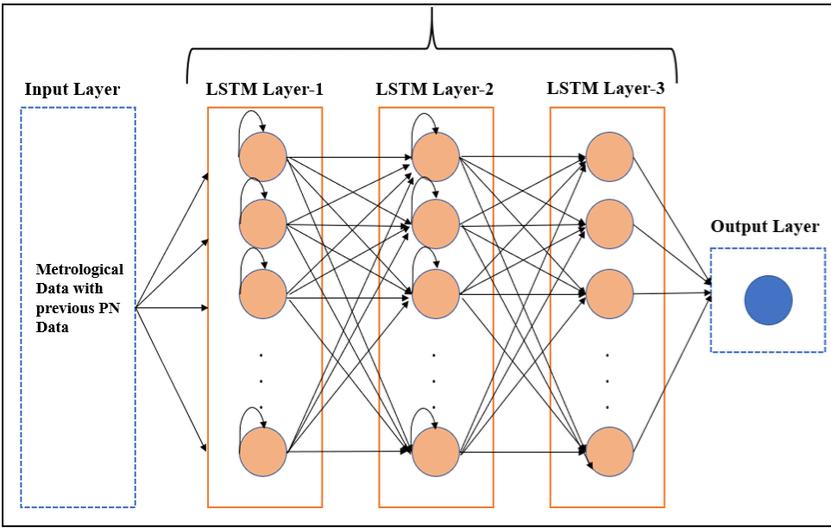
#### **Diagram**:

Input → [Embedding + Positional Encoding]

→ [Multi-Head Attention]

→ [Feed-Forward Network]

→ [Output]

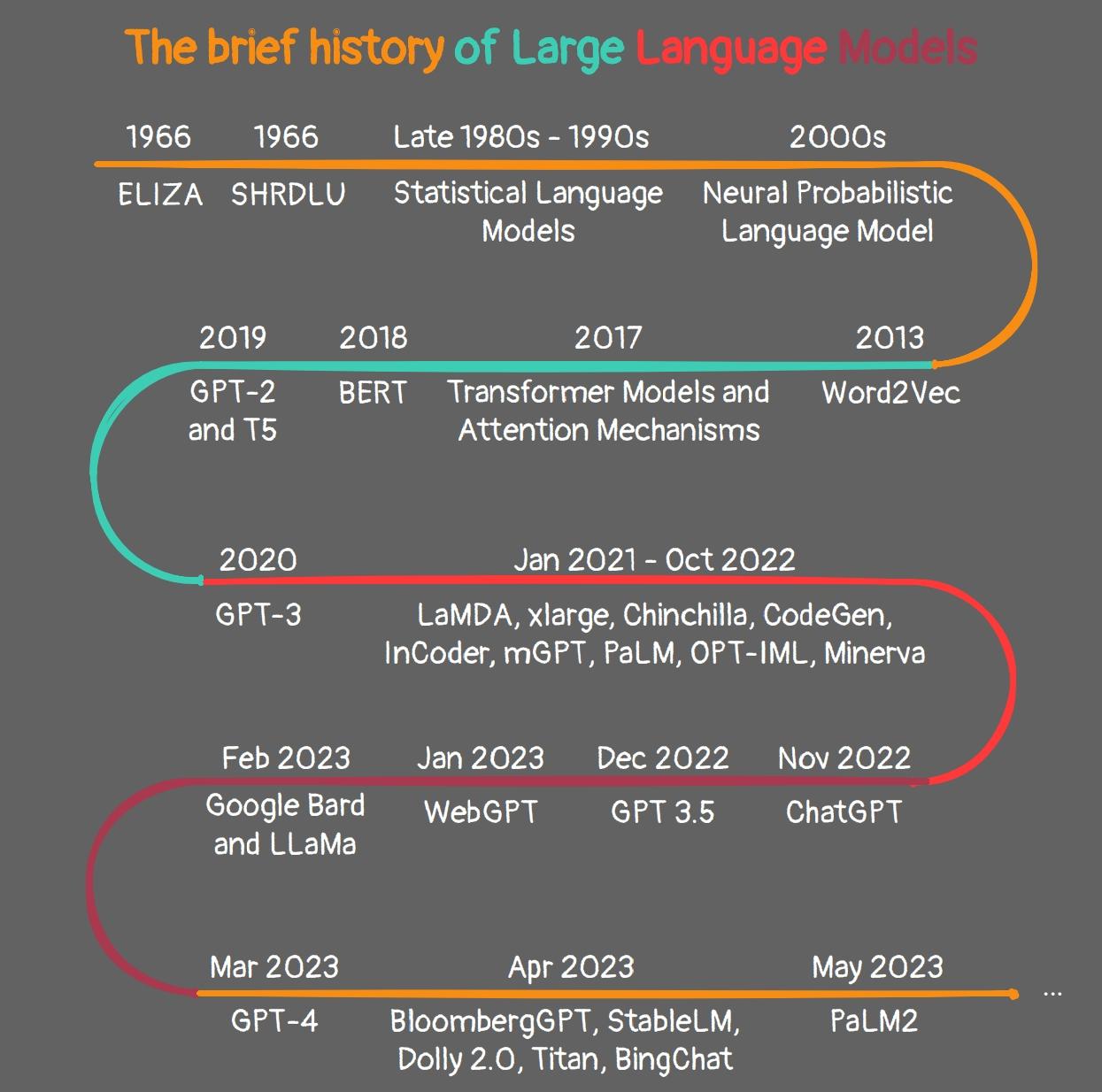


**Benefits**:

* **Parallelizable**: Unlike RNNs, all tokens are processed simultaneously.
* **Long-Range Dependencies**: Can directly attend to all positions in the input.
* **Scalability**: Easier to scale with more data and compute.

### **Comparison Table**

## **3. Key Milestones in LLM Evolution**

The timeline of LLM evolution marks rapid progress:

### **2017 - Transformer**

* Introduced self-attention mechanism.
* Replaced recurrence in NLP tasks.

### **2018 - BERT (Bidirectional Encoder Representations from Transformers)**

* Contextual embeddings in **both directions**.
* **Masked Language Modeling (MLM)** used for pretraining.
* Achieved state-of-the-art on many NLP benchmarks.

### **2018 - GPT (Generative Pre-trained Transformer)**

* **Unidirectional decoder-only** model.
* Trained to predict the next word.
* Foundation for autoregressive text generation.

### **2019 - GPT-2**

* 1.5B parameters.
* Capable of generating surprisingly coherent paragraphs.
* Showcased potential for **few-shot** and **zero-shot** learning.

### **2020 - GPT-3**

* 175B parameters.
* Learned tasks with just examples in the prompt.
* Powered apps like ChatGPT.

### **2021 - Switch Transformer**

* Used **Mixture of Experts (MoE)** to activate only parts of the model per input.
* Efficiently trained models with over 1T parameters.

### **2022 - PaLM, GLaM**

* **PaLM**: 540B parameter model by Google.
* **GLaM**: Sparse MoE model with 1.2T parameters.

### **2023 - GPT-4**

* Multimodal input (text + image).
* Improved reasoning, safety, and factual accuracy.

### **2024 - Claude, Gemini, Mistral**

* Advanced reasoning.
* High safety standards.
* Optimized for efficiency and multilingual capabilities.

### **Diagram: Evolution of LLMs**

RNN → LSTM → Transformer (2017)

↓

BERT (2018)

↓

GPT → GPT-2 (2019) → GPT-3 (2020) → GPT-4 (2023)

↓

PaLM, Claude, Gemini, Mistral (2024+)

## **Conclusion: Strengths & Weaknesses Recap**