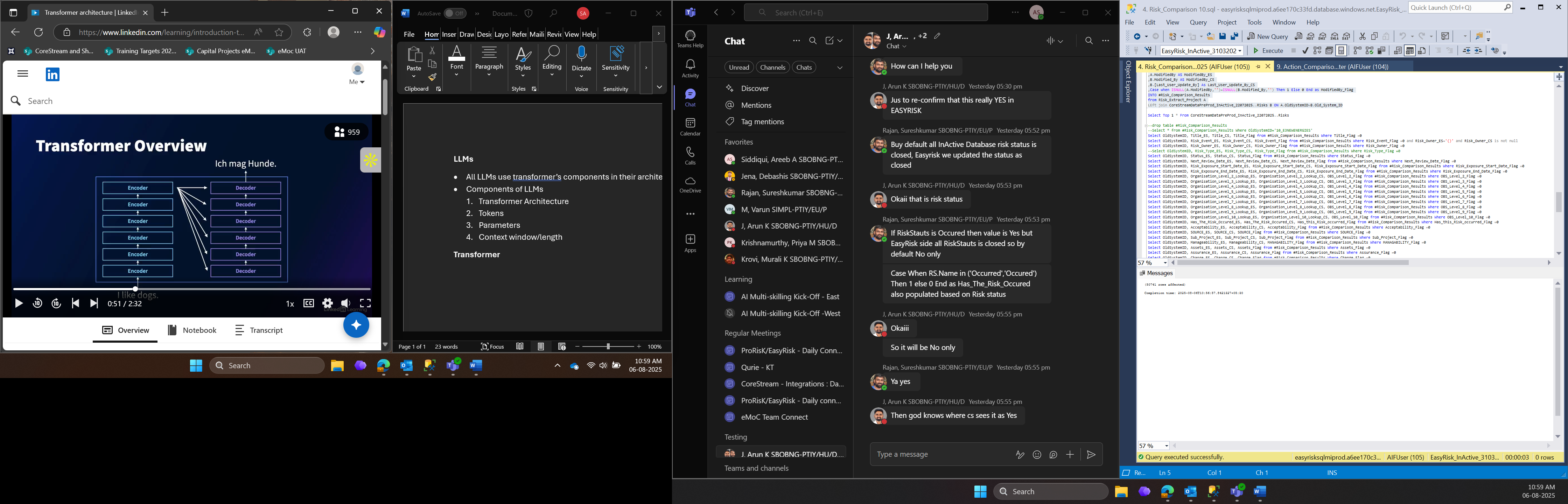
**LLMs**

* All LLMs use transformer’s components in their architecture
* Components of LLMs

1. Transformer Architecture
2. Tokens
3. Parameters
4. Context window/length
5. **Transformer**



6 encoders and 6 decoders

Encoders get the input one after the other ( So they are best for getting deep understanding of the language that is given)

So, as the process goes they can collect a lot of information

Decoder on the other hand is best for text generation

Depending upon our use case we can use Encoder only model or Decoder only model

1. **Parameters**

* Variables that the model learns during the training process
* LLMs have billions of parameters
* They are **adjusted during training** using massive datasets to help the model learn patterns in language.

1. **Tokens**

‘Tokenization is the process of splitting words into smaller chunks pr tokens’

In the above example we have go 14 tokens in total

Each word as well as spaces are considered as different tokens

For example Tokenization has two tokens -> token and ization

Is -> another token

The -> another token

Process -> another token and so on

1. Context Window

Prompt + content generated

Tokens + tokens = context length ( context length )

Longer context length = the model has more information to generate a response

**🧠 How It Works**

The context window is measured in **tokens**, and it includes:

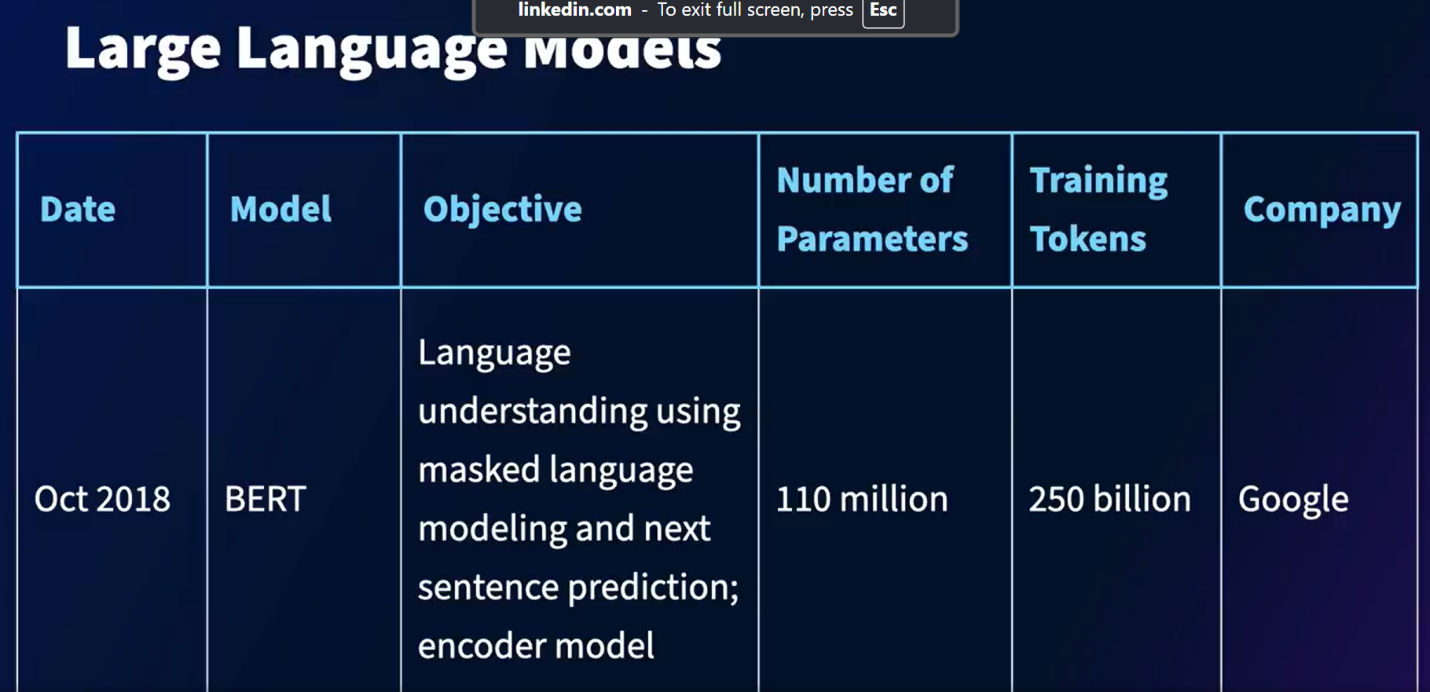
1. **Prompt tokens**: All the text you provide — instructions, questions, documents, etc.
2. **Output tokens**: The model’s generated response.

So, if a model has a context window of **8,000 tokens**, and your prompt uses **6,000 tokens**, the model can only generate up to **2,000 tokens** in its response before hitting the limit.

**📌 Why This Matters**

* If your prompt is **too long**, it limits how much the model can respond.
* If you want a **longer output**, you need to keep the prompt **shorter**.
* For tasks like summarization or code generation, balancing prompt and output size is key.

| **Model** | **Context Window Size** |
| --- | --- |
| GPT-3 | 2,048 tokens |
| GPT-3.5 | 4,096 tokens |
| GPT-4 | Up to 32,768 tokens |
| Claude 2 | 100,000+ tokens |
| Gemini 1.5 | 1 million+ tokens |

BERT was one of the LLM developed by Google ( Encoder only )

**Scaling Laws**

**More the amount training data lesser the test loss**

**More the number of parameters lesser the test loss**

**For best performance**

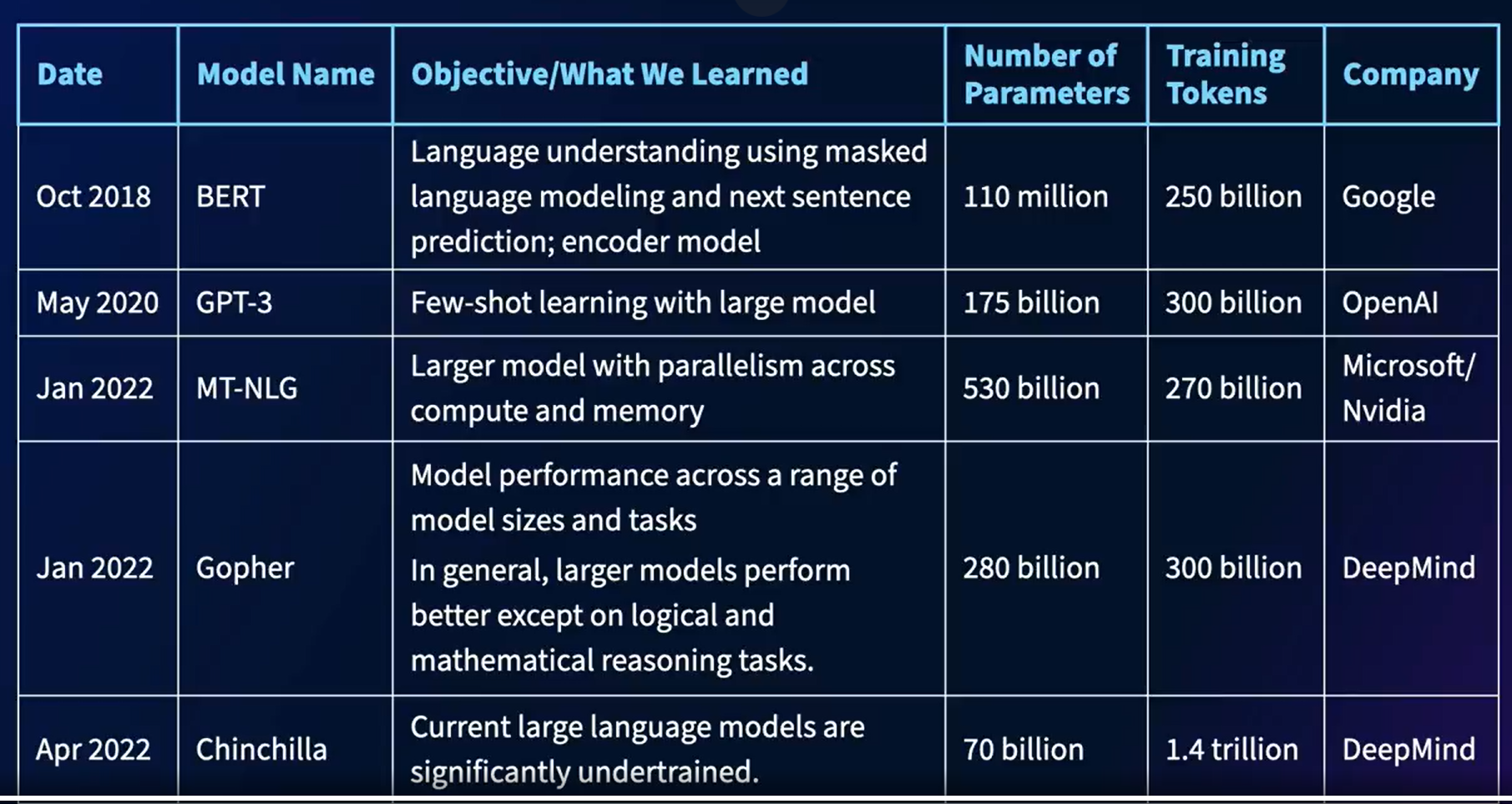
* **Increase the model size ( this is done mostly by big companies )**
* **Increase the data for training**
* **Increase the amount of compute**

**GPT-3 (175 billion parameters )**

**G**enerative : predicts a future token, given past tokens

**P**re-trained : trained on a large corpus of data

**T**ransformer : decoder portion of the architecture



Parameters vs Training Tokens

**Imagine you're training a chef (the model):**

**🍲 Training Tokens = Recipes**

* These are the **instructions and examples** the chef reads to learn how to cook.
* The more recipes the chef reads, the better they understand ingredients, techniques, and flavors.
* In LLMs, training tokens are the **text data** the model learns from — books, articles, websites, etc.

**🧠 Parameters = The Chef’s Skills**

* These are the **internal abilities** the chef develops after reading all those recipes.
* They represent the chef’s **knowledge, intuition, and decision-making**.
* In LLMs, parameters are the **learned weights** that determine how the model responds to new inputs.

**Chinchila**

* Training data is more but the parameters are less but still it outperforms many of the LLMs
* ****

One of the Largest LLM

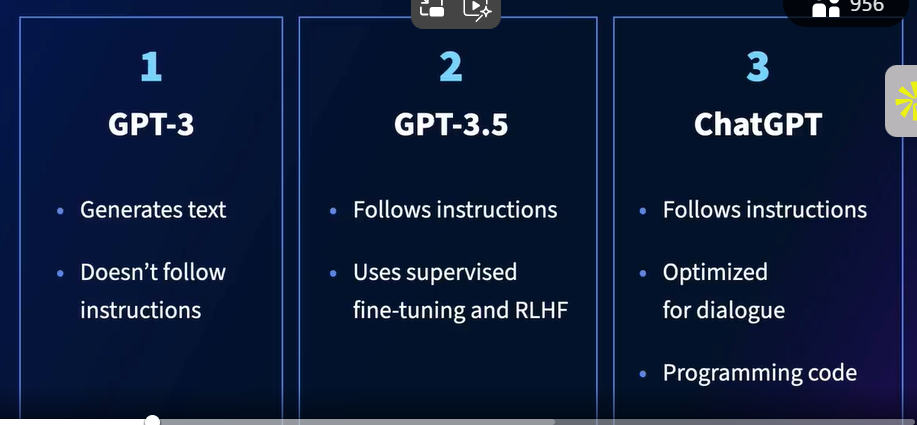
**PaLM ( Pathways Language Model )**

* 540 billion parameters and 780 billion tokens
* Pathways system – trains model faster and more efficiently

1. **Massive Scale**:
   * PaLM has **540 billion parameters**, making it one of the largest models ever trained at the time of its release.
   * Trained using **6144 TPU v4 chips** across multiple TPU Pods.
2. **Pathways System**:
   * PaLM was trained using **Pathways**, a new ML system by Google that enables efficient training across multiple tasks and devices.
   * It allows a single model to generalize across many tasks without needing separate models.
3. **Few-Shot and Multilingual Capabilities**:
   * PaLM excels at **few-shot learning**, meaning it can perform tasks with very few examples.
   * It shows strong performance in **multilingual tasks** and **source code generation**.

| **Model Name** | **Developer** | **Release Date** | **Parameters** | **Training Tokens** |
| --- | --- | --- | --- | --- |
| **GPT-1** | **OpenAI** | **Jun 2018** | **117M** | **~1B** |
| **BERT** | **Google** | **Oct 2018** | **340M** | **~3.3B words** |
| **GPT-2** | **OpenAI** | **Feb 2019** | **1.5B** | **~10B tokens** |
| **XLNet** | **Google** | **Jun 2019** | **340M** | **~33B words** |
| **T5** | **Google** | **Oct 2019** | **11B** | **~34B tokens** |
| **GPT-3** | **OpenAI** | **May 2020** | **175B** | **~300B tokens** |
| **GPT-Neo** | **EleutherAI** | **Mar 2021** | **2.7B** | **~825GB text** |
| **GPT-J** | **EleutherAI** | **Jun 2021** | **6B** | **~825GB text** |
| **Megatron-Turing NLG** | **Microsoft/NVIDIA** | **Oct 2021** | **530B** | **~338B tokens** |
| **GLaM** | **Google** | **Dec 2021** | **1.2T (sparse)** | **~1.6T tokens** |
| **Gopher** | **DeepMind** | **Dec 2021** | **280B** | **~300B tokens** |
| **Ernie 3.0 Titan** | **Baidu** | **Dec 2021** | **260B** | **~4TB text** |
| **Claude 1** | **Anthropic** | **Dec 2021** | **52B** | **~400B tokens** |
| **LaMDA** | **Google** | **Jan 2022** | **Not disclosed** | **Not disclosed** |
| **PaLM** | **Google** | **Apr 2022** | **540B** | **~780B tokens** |
| **Chinchilla** | **DeepMind** | **Mar 2022** | **70B** | **~1.4T tokens** |
| **GPT-4** | **OpenAI** | **Mar 2023** | **Not disclosed** | **Estimated ~1T+ tokens** |
| **Claude 2** | **Anthropic** | **Jul 2023** | **Not disclosed** | **Estimated ~1T+ tokens** |
| **Gemini 1** | **Google DeepMind** | **Dec 2023** | **Not disclosed** | **Estimated ~1T+ tokens** |
| **Claude 3** | **Anthropic** | **Mar 2024** | **Not disclosed** | **Estimated ~1T+ tokens** |
| **Gemini 1.5** | **Google DeepMind** | **Feb 2025** | **Not disclosed** | **Estimated ~10T tokens** |
| **GPT-4 Turbo** | **OpenAI** | **Nov 2023** | **Not disclosed** | **Estimated ~1T+ tokens** |

**GPT 3 vs ChatGPT vs ChatGPT 3.5**

****

\*Optimised for dialogue mean -> good for chatbots like application

GPT-4

* Follows instructions
* Optimized for dialogue
* Programming code
* Complex reasoning
* Human-level performance on exams

It can take text as well as images as input

Model and data used? Not disclosed

**Weakness**

* Can’t be fine-tuned
* Doesn’t know knowledge in real time
* Makes up facts sometimes

**Why do we need the models to be fine tuned?**

* Particular niche ki knowledge ni hoti manlo koi aisi niche hai jahan ka data was not used that much for training, so answer poochenge to sahi to ni milne wala
* Isliye fine tuning is Important

**OpenSource LLMs**

Meta’s OPT ( Open Pre Train )

Bloom ( Hugging Face )

LLaMA ( For research and noncommercial purposes only )