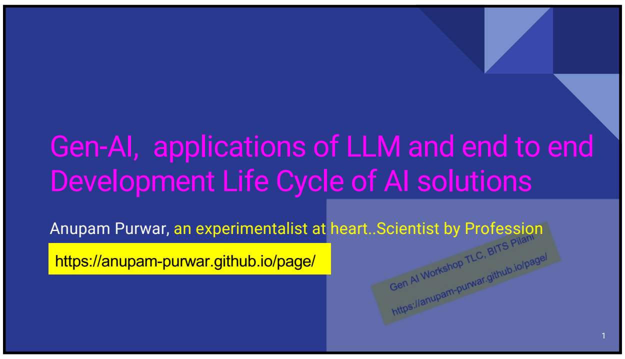
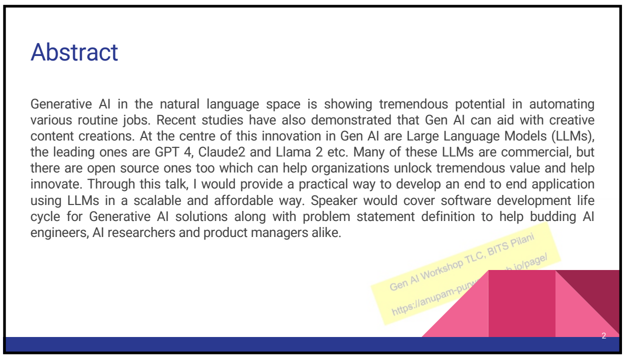


Gen-AI, applications of LLM and end to end Development Life Cycle of AI solutions

Anupam Purwar, an experimentalist at heart..Scientist by Profession

https://anupam-purwar.github.io/page/

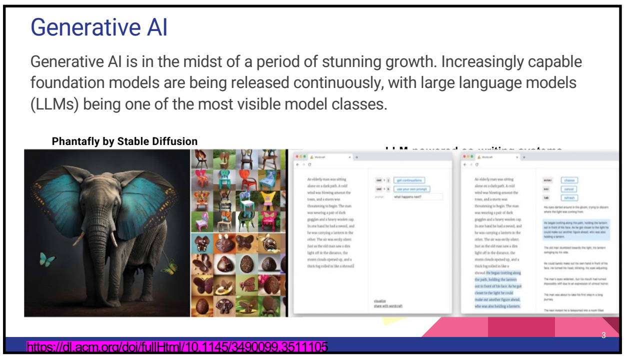
1



Abstract

Generative AI in the natural language space is showing tremendous potential in automating various routine jobs. Recent studies have also demonstrated that Gen AI can aid with creative content creations. At the centre of this innovation in Gen AI are Large Language Models (LLMs), the leading ones are GPT 4, Claude2 and Llama 2 etc. Many of these LLMs are commercial, but there are open source ones too which can help organizations unlock tremendous value and help innovate. Through this talk, I would provide a practical way to develop an end to end application using LLMs in a scalable and affordable way. Speaker would cover software development life cycle for Generative AI solutions along with problem statement definition to help budding AI engineers, AI researchers and product managers alike.

2



Generative AI

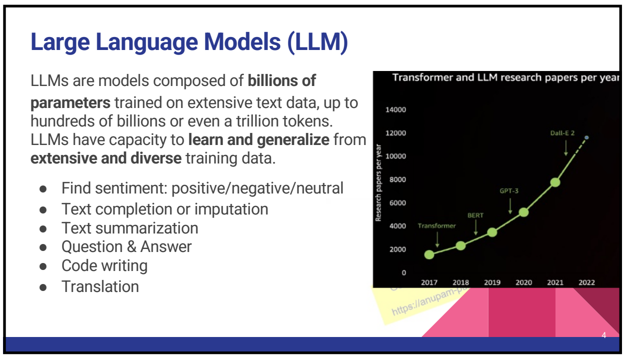
Generative AI is in the midst of a period of stunning growth. Increasingly capable foundation models are being released continuously, with large language models (LLMs) being one of the most visible model classes.

Phantafly by Stable Diffusion

LLM powered co-writing systems

https://dl.acm.org/doi/fullHtml/10.1145/3490099.3511105

3



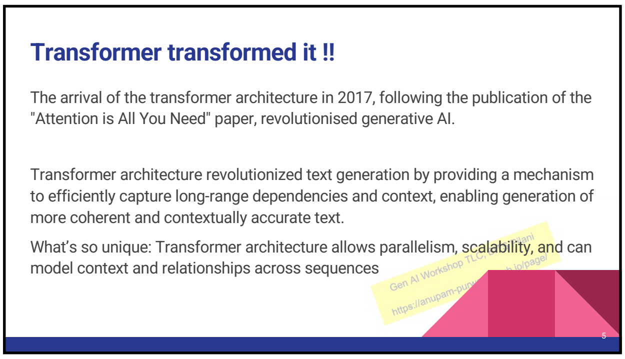
Large Language Models (LLM)

LLMs are models composed of **billions of parameters** trained on extensive text data, up to hundreds of billions or even a trillion tokens. LLMs have capacity to **learn and generalize** from **extensive and diverse** training data.

- Find sentiment: positive/negative/neutral
- Text completion or imputation
- Text summarization
- Question & Answer
- Code writing
- Translation

Transformer and LLM research papers per year

4



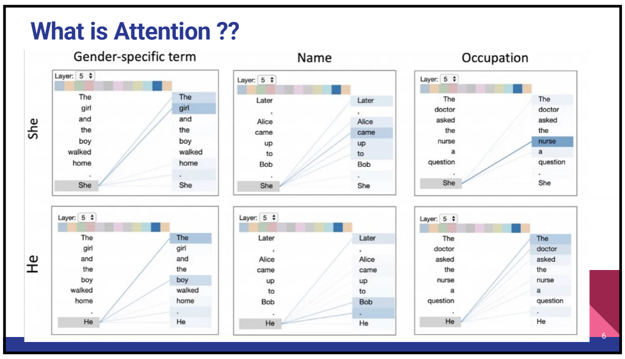
Transformer transformed it !!

The arrival of the transformer architecture in 2017, following the publication of the "Attention is All You Need" paper, revolutionised generative AI.

Transformer architecture revolutionized text generation by providing a mechanism to efficiently capture long-range dependencies and context, enabling generation of more coherent and contextually accurate text.

What's so unique: Transformer architecture allows parallelism, scalability, and can model context and relationships across sequences

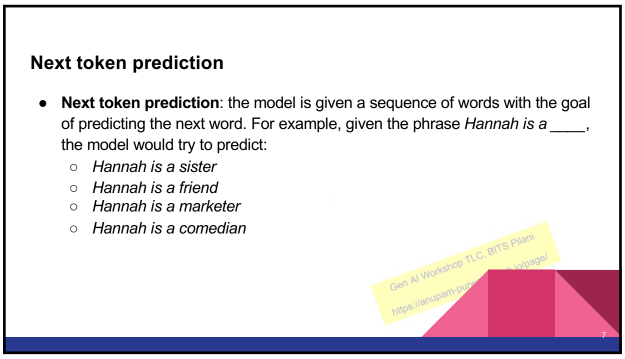
5



What is Attention ??

Gender-specific term
Name
Occupation
She
He
Layer: 5
The girl and the boy walked home . She
Later , Alice came up to Bob . She
The doctor asked the nurse a question . She
The girl and the boy walked home . He
Later , Alice came up to Bob . He
The doctor asked the nurse a question . He

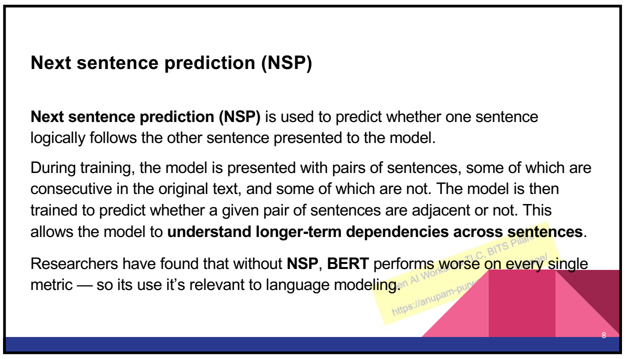
6



Next token prediction

- **Next token prediction**: the model is given a sequence of words with the goal of predicting the next word. For example, given the phrase *Hannah is a* ____, the model would try to predict:
  - *Hannah is a sister*
  - *Hannah is a friend*
  - *Hannah is a marketer*
  - *Hannah is a comedian*

7



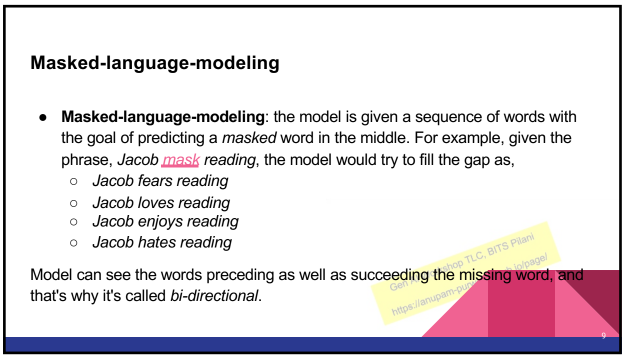
Next sentence prediction (NSP)

Next sentence prediction (NSP) is used to predict whether one sentence logically follows the other sentence presented to the model.

During training, the model is presented with pairs of sentences, some of which are consecutive in the original text, and some of which are not. The model is then trained to predict whether a given pair of sentences are adjacent or not. This allows the model to **understand longer-term dependencies across sentences**.

Researchers have found that without **NSP**, **BERT** performs worse on every single metric — so its use it's relevant to language modeling.

8

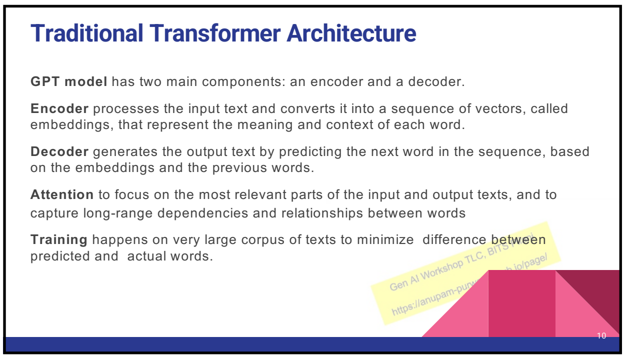


Masked-language-modeling

- **Masked-language-modeling**: the model is given a sequence of words with the goal of predicting a *masked* word in the middle. For example, given the phrase, *Jacob mask reading*, the model would try to fill the gap as,
  - *Jacob fears reading*
  - *Jacob loves reading*
  - *Jacob enjoys reading*
  - *Jacob hates reading*

Model can see the words preceding as well as succeeding the missing word, and that's why it's called *bi-directional*.

9



Traditional Transformer Architecture

GPT model has two main components: an encoder and a decoder.

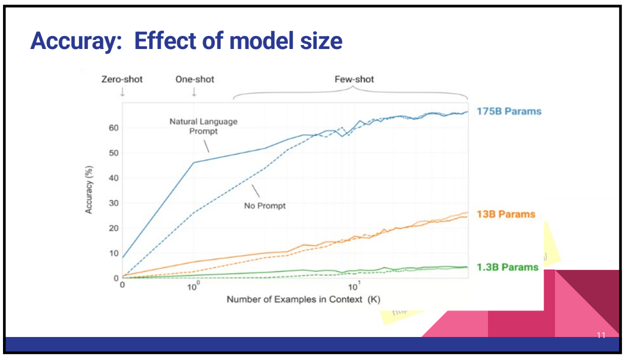
Encoder processes the input text and converts it into a sequence of vectors, called embeddings, that represent the meaning and context of each word.

Decoder generates the output text by predicting the next word in the sequence, based on the embeddings and the previous words.

Attention to focus on the most relevant parts of the input and output texts, and to capture long-range dependencies and relationships between words

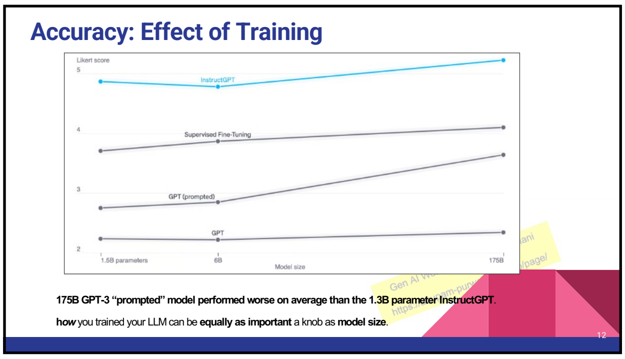
Training happens on very large corpus of texts to minimize difference between predicted and actual words.

10



Accuray: Effect of model size

11

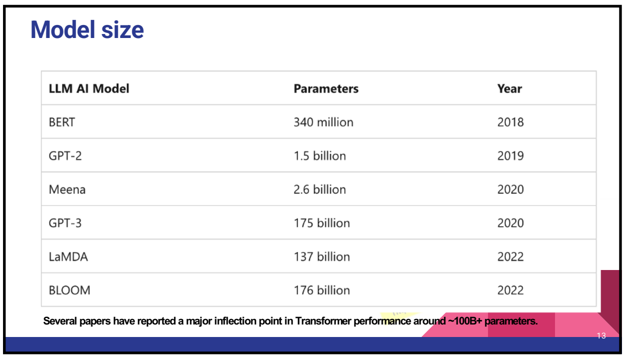


Accuracy: Effect of Training

175B GPT-3 “prompted” model performed worse on average than the 1.3B parameter InstructGPT.

how you trained your LLM can be **equally as important** a knob as **model size**.

12



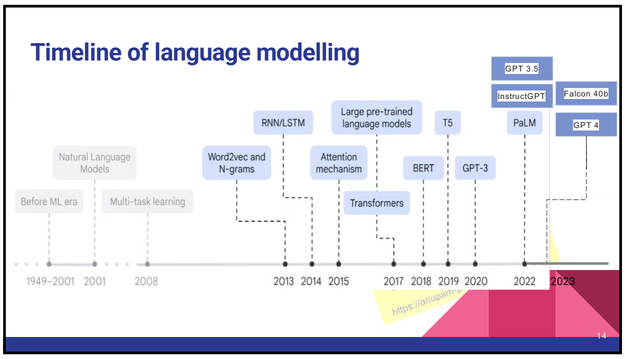
Model size

LLM AI Model	Parameters	Year
BERT	340 million	2018
GPT-2	1.5 billion	2019
Meena	2.6 billion	2020
GPT-3	175 billion	2020
LaMDA	137 billion	2022
BLOOM	176 billion	2022

Several papers have reported a major inflection point in Transformer performance around ~100B+ parameters.

13

13



Timeline of language modelling

Before ML era

Natural Language Models

Multi-task learning

Word2vec and N-grams

RNN/LSTM

Attention mechanism

Large pre-trained language models

Transformers

BERT

T5

GPT-3

PaLM

GPT 3.5

InstructGPT

Falcon 40b

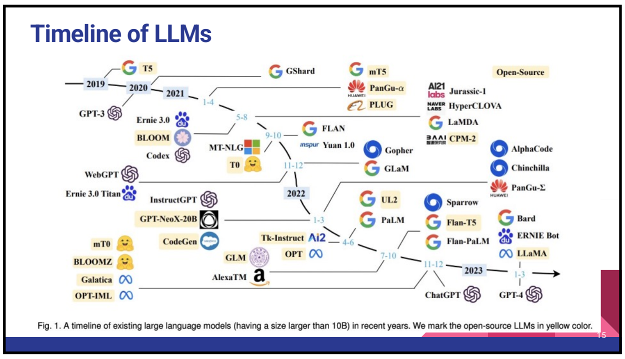
GPT 4

1949–2001 2001 2008 2013 2014 2015 2017 2018 2019 2020 2022 2023

https://anupam-p

14

14



Timeline of LLMs

2019 2020 2021 2022 2023

1-4 5-8 9-10 11-12 1-3 4-6 7-10 11-12 1-3

T5

GPT-3

GShard

mT5

PanGu-α

PLUG

Open-Source

Jurassic-1

HyperCLOVA

Ernie 3.0

BLOOM

Codex

MT-NLG

T0

FLAN

Yuan 1.0

LaMDA

CPM-2

Gopher

GLaM

AlphaCode

Chinchilla

PanGu-Σ

WebGPT

Ernie 3.0 Titan

InstructGPT

GPT-NeoX-20B

CodeGen

UL2

PaLM

Sparrow

Flan-T5

Flan-PaLM

Bard

ERNIE Bot

LLaMA

mT0

BLOOMZ

Galatica

OPT-IML

GLM

AlexaTM

Tk-Instruct

OPT

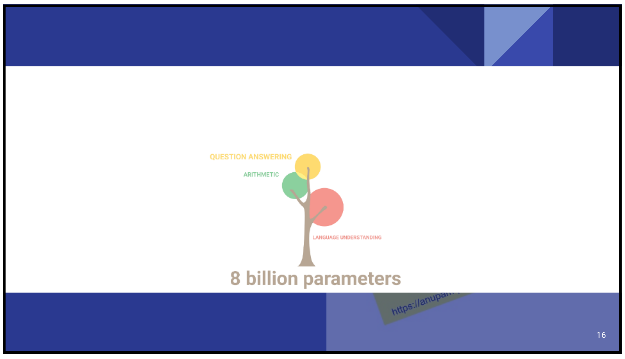
ChatGPT

GPT-4

Fig. 1. A timeline of existing large language models (having a size larger than 10B) in recent years. We mark the open-source LLMs in yellow color.

15

15



QUESTION ANSWERING

ARITHMETIC

LANGUAGE UNDERSTANDING

8 billion parameters

https://anupa

16

16

What are the Use Cases for LLMs?

While Chatbots have emerged to become the most popular applications of LLMs, there are a variety of other tasks that LLMs can be used to accomplish

Writing - From essays to emails to reports and more

Summarisation - Summarise long content into a meaningful shorter length

Language Translation - Translate text from one language to the other

Code - Translate natural language to machine code

Information Retrieval - Retrieve specific information from text like names, locations, sentiment

Augmented LLM - Power interactions with real world by providing information outside of LLM training

17

17

Standard NLP vs LLMs

	Standard NLP	LLM
Learning Approach	Rule based	Data Driven
Feature Engineering	Manual	Automated
Contextual Understanding	Limited	Excellent
Few-Shot and Zero-Shot Learning	Not possible	Possible
Usage	Task specific	Multi-tasking
Resources required	Low	High
Development time	Medium	High*/Low**

18

18

What is a Prompt?

Natural language instruction in which we interact with an LLM is called a Prompt. Prompt construction is called Prompt Engineering.

The inferencing that an LLM does and completes the instruction given in prompt is called 'in context learning

Zero Shot Learning: Ability of LLM to respond to instruction in prompt without any example

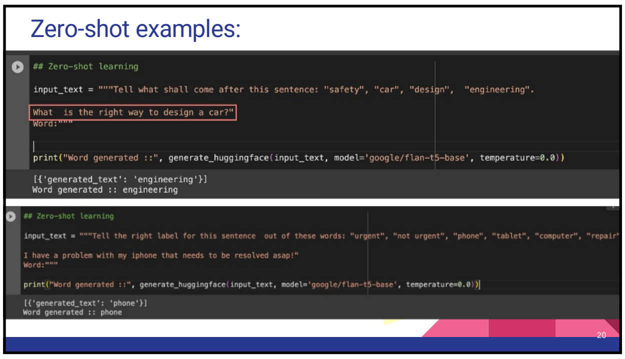
1 shot learning: When a single example is provided, it's called 'One Shot Learning'

Few shot learning: When few examples are provided, it's called 'few Shot Learning'

19

19

Zero-shot examples:



```
## Zero-shot learning

input_text = """Tell what shall come after this sentence: "safety", "car", "design",  "engineering".

What  is the right way to design a car?"
Word:"""

|
print("Word generated ::", generate_huggingface(input_text, model='google/flan-t5-base', temperature=0.0))
```

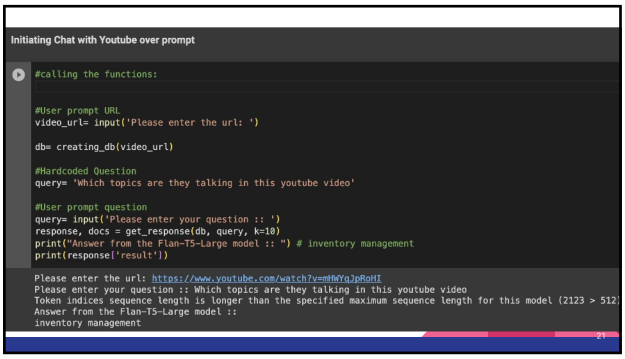
```
[{'generated_text': 'engineering'}]
Word generated :: engineering
```

```
## Zero-shot learning

input_text = """Tell the right label for this sentence  out of these words: "urgent", "not urgent", "phone", "tablet", "computer", "repair"

I have a problem with my iphone that needs to be resolved asap!"
Word:"""

print("Word generated ::", generate_huggingface(input_text, model='google/flan-t5-base', temperature=0.0))
```

```
[{'generated_text': 'phone'}]
Word generated :: phone
```

20

20



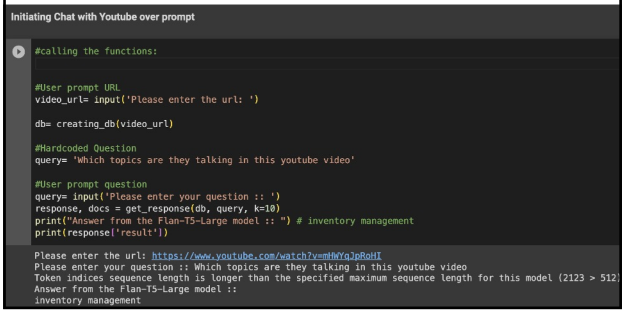
Initiating Chat with Youtube over prompt

```
#calling the functions:

#User prompt URL
video_url= input('Please enter the url: ')

db= creating_db(video_url)

#Hardcoded Question
query= 'Which topics are they talking in this youtube video'

#User prompt question
query= input('Please enter your question :: ')
response, docs = get_response(db, query, k=10)
print("Answer from the Flan-T5-Large model :: ") # inventory management
print(response['result'])
```

```
Please enter the url: https://www.youtube.com/watch?v=mHWYqJpRoHI
Please enter your question :: Which topics are they talking in this youtube video
Token indices sequence length is longer than the specified maximum sequence length for this model (2123 > 512)
Answer from the Flan-T5-Large model ::
inventory management
```

21

Compare: **Flan-T5-Large** vs GPT 3.5



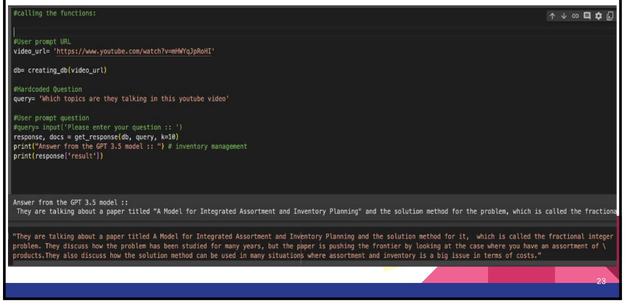
Initiating Chat with Youtube over prompt

```
#calling the functions:

#User prompt URL
video_url= input('Please enter the url: ')

db= creating_db(video_url)

#Hardcoded Question
query= 'Which topics are they talking in this youtube video'

#User prompt question
query= input('Please enter your question :: ')
response, docs = get_response(db, query, k=10)
print("Answer from the Flan-T5-Large model :: ") # inventory management
print(response['result'])
```

```
Please enter the url: https://www.youtube.com/watch?v=mHWYqJpRoHI
Please enter your question :: Which topics are they talking in this youtube video
Token indices sequence length is longer than the specified maximum sequence length for this model (2123 > 512)
Answer from the Flan-T5-Large model ::
inventory management
```

22

Compare: Flan-T5-XL vs **GPT 3.5**



```
#calling the functions:

#User prompt URL
video_url= 'https://www.youtube.com/watch?v=mHWYqJpRoHI'

db= creating_db(video_url)

#Hardcoded Question
query= 'Which topics are they talking in this youtube video'

#User prompt question
#query= input('Please enter your question :: ')
response, docs = get_response(db, query, k=10)
print("Answer from the GPT 3.5 model :: ") # inventory management
print(response['result'])
```

```
Answer from the GPT 3.5 model ::
 They are talking about a paper titled "A Model for Integrated Assortment and Inventory Planning" and the solution method for the problem, which is called the fractiona
```

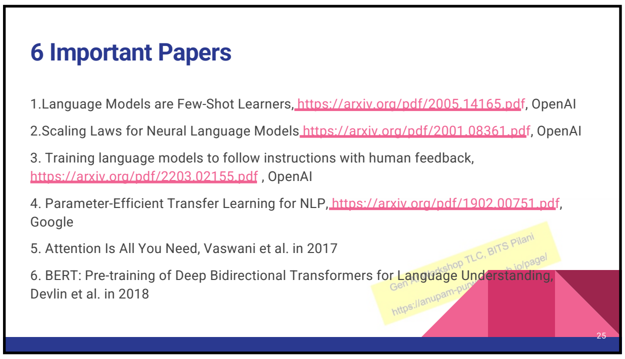
```
"They are talking about a paper titled A Model for Integrated Assortment and Inventory Planning and the solution method for it,  which is called the fractional integer
problem. They discuss how the problem has been studied for many years, but the paper is pushing the frontier by looking at the case where you have an assortment of \
products.They also discuss how the solution method can be used in many situations where assortment and inventory is a big issue in terms of costs."
```

23

How does ChatGPT work?

ChatGPT doesn't use the internet to locate answers, unlike other AI assistants like Siri or Alexa. Instead, it constructs a sentence word by word, selecting the most likely "token" that should come next based on its training. In other words, ChatGPT arrives at an answer by making a series of guesses, which is part of why it can argue wrong answers as if they were completely true.

24



6 Important Papers

1.Language Models are Few-Shot Learners, https://arxiv.org/pdf/2005.14165.pdf, OpenAI

2.Scaling Laws for Neural Language Models https://arxiv.org/pdf/2001.08361.pdf, OpenAI

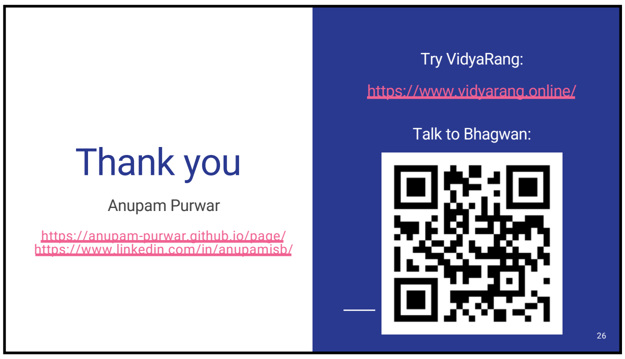
3. Training language models to follow instructions with human feedback, https://arxiv.org/pdf/2203.02155.pdf , OpenAI

4. Parameter-Efficient Transfer Learning for NLP, https://arxiv.org/pdf/1902.00751.pdf, Google

5. Attention Is All You Need, Vaswani et al. in 2017

6. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, Devlin et al. in 2018

25



Thank you

Anupam Purwar

https://anupam-purwar.github.io/page/
https://www.linkedin.com/in/anupamisb/

Try VidyaRang:

https://www.vidyarang.online/

Talk to Bhagwan:

26