Introduction to Large Language Models (LLMs)

Prof. Tanmoy Chakraborty, Prof. Soumen Chakraborti

Department of Computer Science & Engineering

Indian Institute of Technology, Delhi

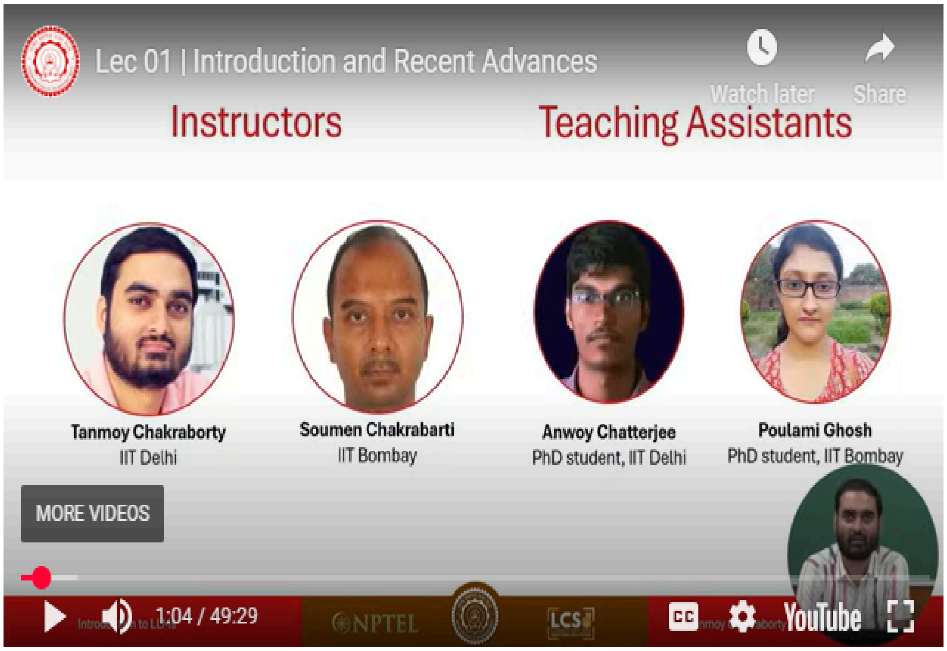
Lecture 1

Introduction and Recent Advances

Hello everyone. Welcome to the course on large language model. This is an introductory chapter for this course. So, in this specific lecture, we are basically, trying to understand why large language model is important to learn. I'm pretty sure I don't need to convince you why this is an important topic, at least in today's era. We are all aware of chat GPT and how chat GPT has disrupted our society.

So in this lecture, we will essentially try to cover the content that we are going to cover in the entire course and some of the logistics that are needed to essentially you know understand different content and so on and so forth okay.

(Refer Slide Time: 0:58)


Lec 01 | Introduction and Recent Advances

Instructors

Teaching Assistants

Tanmoy Chakraborty
IIT Delhi

Soumen Chakrabarti
IIT Bombay

Anwoy Chatterjee
PhD student, IIT Delhi

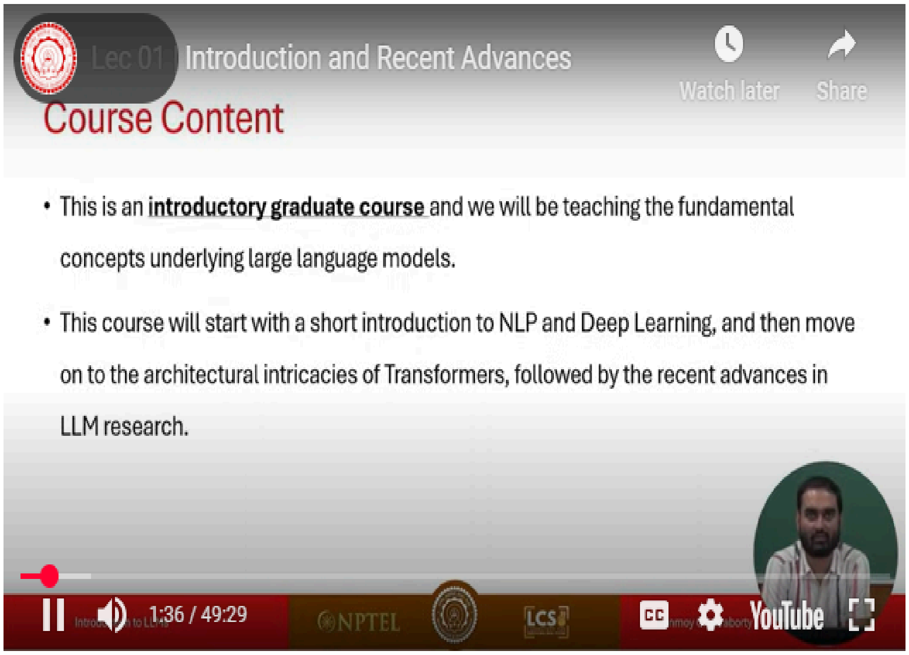
Poulami Ghosh
PhD student, IIT Bombay

So, let's get started first things first. My name is Tanmay Chakraborty. I am one of the instructors for this course.

I am an associate prof at IIT Delhi. Along with me, Professor Swamin Chakraborty from IIT Bombay. He will also teach a part of this course. And this course also has two wonderful TAs, very supportive TAs. One is my student Anuay Chatterjee from IIT Delhi and other is Pallami Ghosh.

He is a PhD student at IIT Bombay.

(Refer Slide Time: 01:32)



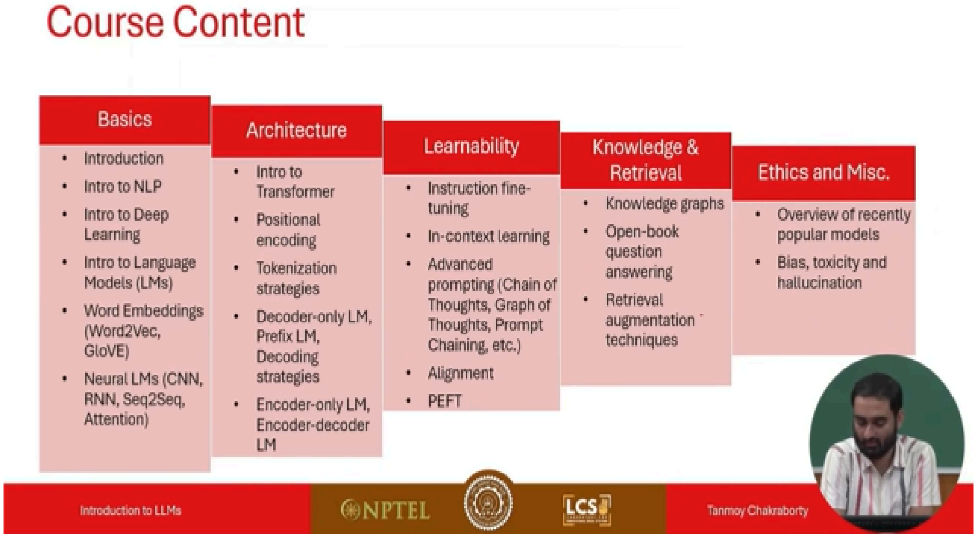
Course Content

- This is an **introductory graduate course** and we will be teaching the fundamental concepts underlying large language models.
- This course will start with a short introduction to NLP and Deep Learning, and then move on to the architectural intricacies of Transformers, followed by the recent advances in LLM research.

So this is the course content. So this is essentially an introductory graduate-level course. And the hope is that at least the audience would have some understanding of NLP. Although we will cover basics of NLP, but it's not possible to cover the details of NLP in this course.

And of course, deep learning is needed. So if you haven't learned NLP and deep learning, it's recommended that you can go through some of the online courses or you can go through some of the books. There are many wonderful books in NLP and deep learning. So these two topics are very important to learn as a prerequisite for this course.

(Refer Slide Time: 02:16)



Course Content

Basics

- Introduction
- Intro to NLP
- Intro to Deep Learning
- Intro to Language Models (LMs)
- Word Embeddings (Word2Vec, GloVE)
- Neural LMs (CNN, RNN, Seq2Seq, Attention)

Architecture

- Intro to Transformer
- Positional encoding
- Tokenization strategies
- Decoder-only LM, Prefix LM, Decoding strategies
- Encoder-only LM, Encoder-decoder LM

Learnability

- Instruction fine-tuning
- In-context learning
- Advanced prompting (Chain of Thoughts, Graph of Thoughts, Prompt Chaining, etc.)
- Alignment
- PEFT

Knowledge & Retrieval

- Knowledge graphs
- Open-book question answering
- Retrieval augmentation techniques

Ethics and Misc.

- Overview of recently popular models
- Bias, toxicity and hallucination

Introduction to LLMs

Tanmoy Chakraborty

And this is more or less the content that we are going to cover.

We will start with basics, okay. So this is, so today's lecture is the introductory lecture. Then we have a lecture on the introduction of NLP. Introduction to NLP, this is essentially you know it's going to be an hour-long lecture on NLP but you know it's not possible to cover the I mean all different topics of NLP in one hour. But what I will try, I will try to basically give you a very high-level overview of the different interesting contexts in NLP.

And then I will also briefly cover neural networks, specifically you know activation functions, back propagation, right, different types of architectures, basic architectures in neural network. And then, We will have one lecture on language model, statistical language model, where we will cover in-gram language model, different types of smoothing techniques, evaluation of language model, perplexity, extrinsic evaluation, intrinsic evaluation and so on and then we move to word embeddings, so we will essentially connect the concept of distributional semantics and recent word embeddings. We will essentially see why old school techniques like TF-IDF based methods for word representation may not be a good idea, why we need the concept of distributional semantics and why we should learn word embeddings from documents. There we will cover what we can love, these two methods and then we move to Neural language model where we will essentially teach CNN, RNN, LSTM, GRU models for language models and then we move to sequence-to-sequence models right for machine translation for text to text kind of problems and then we will introduce attention okay. So these are the basics that we will cover.

And then we move to architecture. In this bucket we will start with the introduction to transformer, the transformer model which is the foundation of existing large language models available, you know, publicly available or some of the black box models, right. And then we also talk about the concept of positional encoding. We will see that in transformer, the beauty about transformer is that, unlike RNN right, where RNN kind of models where sequence is important right and you know RNN LSTM kind of models, they process a sequence of data and that's the bottleneck because of the sequential nature of the data, the model is not able to basically access the input in parallel, right. So transformer was introduced, where they showed that how to access the input in parallel and to access the input in parallel as you understand we essentially miss out the information of position.

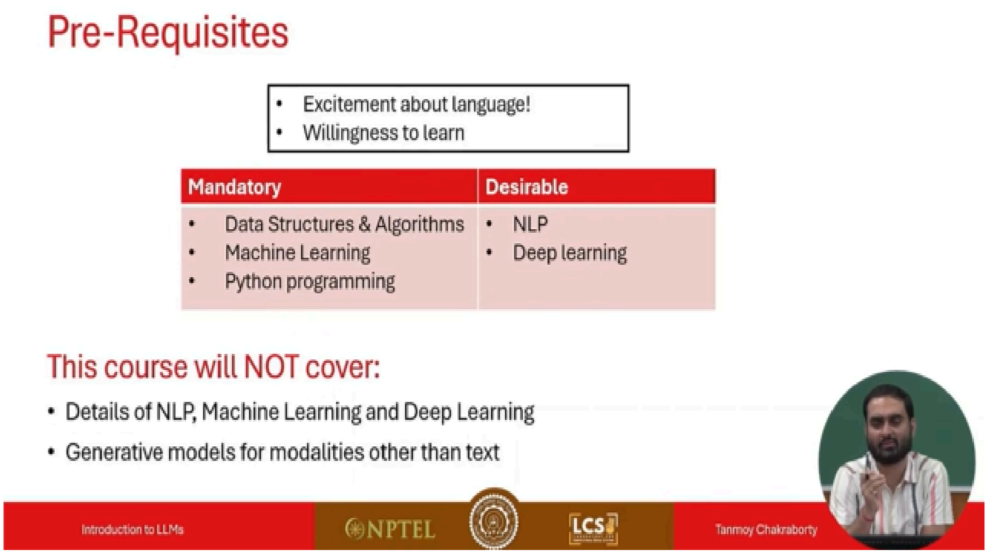
So positional encoding is essentially a way to encode the position information along with the actual information present within the token. Then we will talk about different tokenization strategies, byte pair, you know sentence piece, word piece and so on. Then we move to different pre-training strategies, decoder only models, prefix model, encoder only model, encoder decoder model and so on and so forth. So there we will talk about bot and different GPT models and also T5, okay and then we move to different aspects of learnability. These are more advanced concepts.

So we will talk about instruction fine-tuning, we will talk about in-context learning, we will also talk about advanced prompting techniques, Chain of thoughts, graph of thoughts, tree of thoughts, prompt chaining and so on. Then we will discuss very important concept of human human alignment, right, there, we will talk about RLHF, you know, if time permits will talk, also talk about DPO and so on. And then, we will see how we can essentially reduce the size of these models, right, systematically. One way of reducing the size is basically something called PEFT, parameter efficient fine tuning where we will essentially leverage the ideas of distillation, knowledge distillation where you have a teacher model, a large teacher model and a small student model and you essentially distill knowledge from a teacher model to a small student model, so that the student model performs as similar as the teacher model. We will also talk about different types of adapter based methods.

and so on and so forth. Then we move to more advanced concept knowledge and retrieval. There we will talk about knowledge graphs and open book question answering systems or models which essentially can be used for question answering in an open book setting and we also talk about retrieval augmented techniques. And at the end of this course, we will cover some of the ethical aspects of large language models, bias, toxicity, hallucination and we will also try to cover some of the advanced concepts in LLMs including state space model and You know, if possible, some other aspects as well.

Okay. So this is more or less the content, the tentative content. Hopefully we'll try to cover, we will be able to cover most of the content, but it again also depends on the time and the context.

(Refer Slide Time: 08:37)



Pre-Requisites

- Excitement about language!
- Willingness to learn

Mandatory	Desirable
• Data Structures & Algorithms • Machine Learning • Python programming	• NLP • Deep learning

This course will NOT cover:

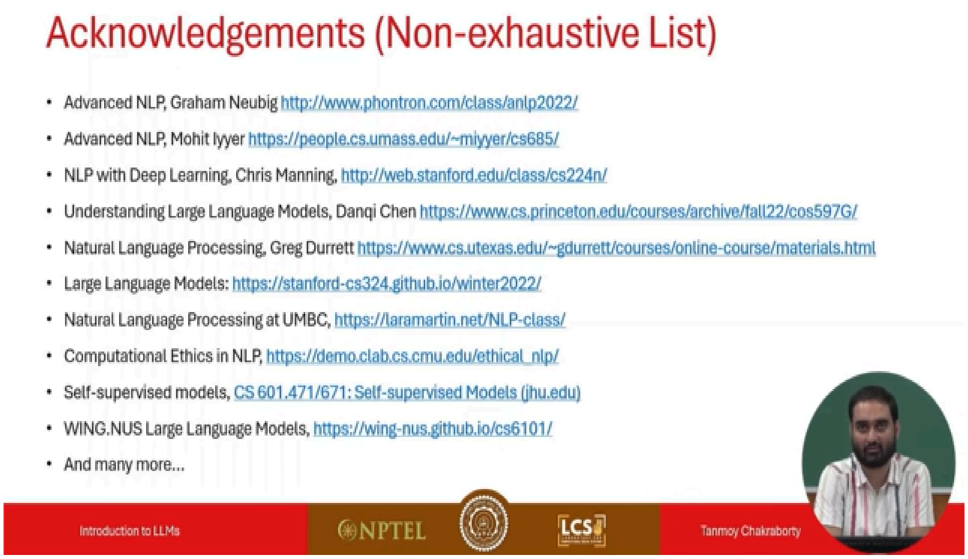
- Details of NLP, Machine Learning and Deep Learning
- Generative models for modalities other than text

So the prerequisite, you know, I always show this slide, that I believe, that for any course there is no such prerequisite right, the prerequisites are basically you should have an excitement about learning something new, in this case, learning about language, about language models and of course willingness to learn something new right. But officially, you know data structure algorithms, machine learning should be the prerequisites for the course, we will not touch upon any of these concepts in this specific lecture and then again, the hope is that you have strong understanding of Python programming. You have strong Python programming skill.

There are a few lectures on essentially Python programming for this large language model. So the two TAs will take a couple of lectures on how to basically, code to use LLMs for our day-to-day life. And as I mentioned earlier, the desirable subjects include NLP and deep learning. We will not be able to cover NLP in details and deep learning in details. So this course will not cover NLP, machine learning, deep learning, this course will not cover generative AI for modalities other than the text.

So this course is essentially generative model for text. So for example, we will not cover any stable diffusion kind of models or let us say mid journey dally kind of models right, but here we will briefly cover the some vision language models right, only one there is one chapter on vision language model that we will cover.

(Refer Slide Time: 10:33)



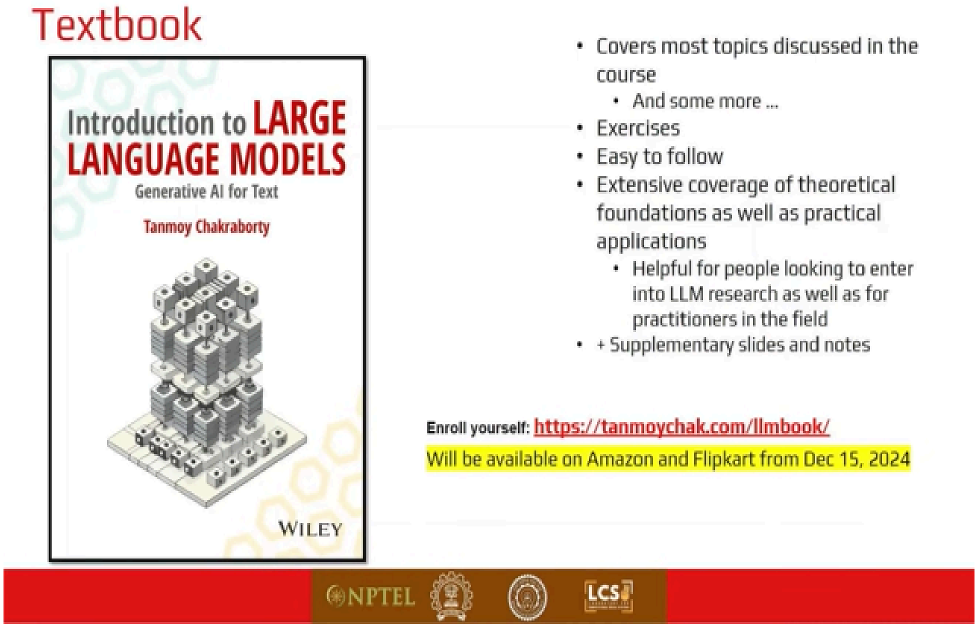
Acknowledgements (Non-exhaustive List)

- Advanced NLP, Graham Neubig http://www.phontron.com/class/anlp2022/
- Advanced NLP, Mohit Iyyer https://people.cs.umass.edu/~miyyer/cs685/
- NLP with Deep Learning, Chris Manning, http://web.stanford.edu/class/cs224n/
- Understanding Large Language Models, Danqi Chen https://www.cs.princeton.edu/courses/archive/fall22/cos597G/
- Natural Language Processing, Greg Durrett https://www.cs.utexas.edu/~gdurrett/courses/online-course/materials.html
- Large Language Models: https://stanford-cs324.github.io/winter2022/
- Natural Language Processing at UMBC, https://laramartin.net/NLP-class/
- Computational Ethics in NLP, https://demo.clab.cs.cmu.edu/ethical_nlp/
- Self-supervised models, CS 601.471/671: Self-supervised Models (jhu.edu)
- WING.NUS Large Language Models, https://wing-nus.github.io/cs6101/
- And many more...

So this is the non-exhaustive list of online content which have been used to design some of the slides for this course. So we acknowledge all the content creators. So some of the content are taken from CMU Graham Newbig's course, some content from Mohith Iyer's course, from UMass, Chris Manning's course from Stanford.

you know some material from University of Texas, Austin, UMBC and so on and so forth. This is of course not a non, this is not an exhaustive list, right. So, you know over the time, we will duly acknowledge the sources from where we have taken content.

(Refer Slide Time: 11:19)


Introduction to LARGE LANGUAGE MODELS
Generative AI for Text
Tanmoy Chakraborty
WILEY
NPTEL
LCS

Textbook

- Covers most topics discussed in the course
  - And some more ...
- Exercises
- Easy to follow
- Extensive coverage of theoretical foundations as well as practical applications
  - Helpful for people looking to enter into LLM research as well as for practitioners in the field
- + Supplementary slides and notes

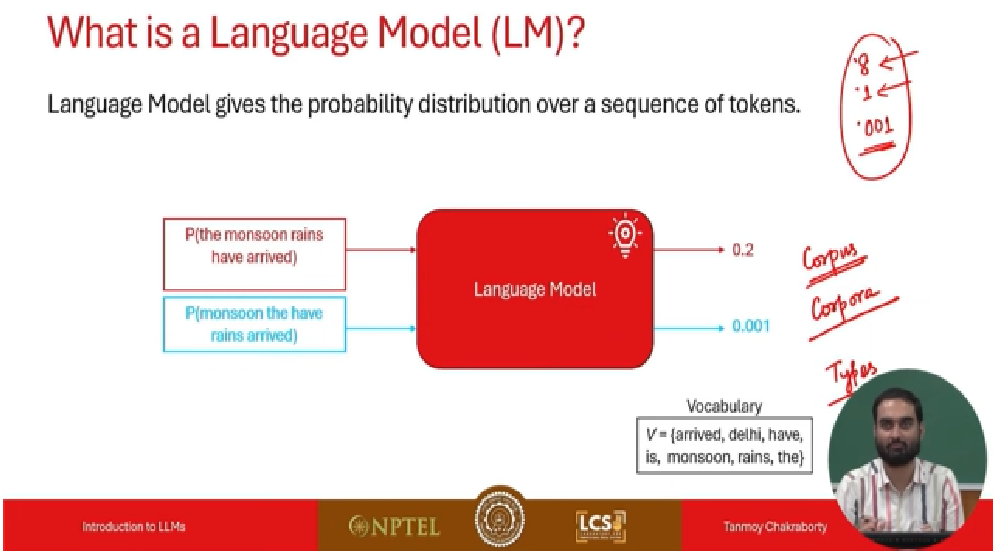
Enroll yourself: https://tanmoychak.com/llmbook/

Will be available on Amazon and Flipkart from Dec 15, 2024

For this course We will follow this textbook, this is the textbook on large language models written by me. The book covers various topics, those which will be covered in this lecture as well as other topics as well.

With every chapter, we have comprehensive exercise and slides, supplementary tools and you know interesting papers that are there related to the chapter.

(Refer Slide Time: 11:45)



What is a Language Model (LM)?

Language Model gives the probability distribution over a sequence of tokens.

P(the monsoon rains have arrived) → Language Model → 0.2

P(monsoon the have rains arrived) → Language Model → 0.001

Vocabulary

V = {arrived, delhi, have, is, monsoon, rains, the}

Introduction to LLMs | NPTEL | LCS | Tanmoy Chakraborty

So, let us get started. So this course is about large language model, so let us try to understand what is a language model, although there is a separate chapter on language model, but here you know just to start the overall content right, so I thought let me introduce language model in this particular lecture. Essentially is a probability distribution over a sequence of tokens.

Okay. Let's say you are given a token, you are given a sequence of tokens. The tokens can be words, tokens can be characters, tokens can be subwords, tokens can be a sequence of numbers, tokens can be anything. Right. So if you are given a sequence of tokens, the task would be to essentially measure the probability of the sequence of tokens. Now remember when I mention the term sentence, I use the term sentence vaguely or loosely.

A sentence, I mean when we use the term sentence, it may not indicate that it's a valid sentence. A sentence can be just an invalid sequence of tokens right, valid slash invalid sequence of tokens okay. So language model so if you let's say if there is a sequence of tokens like the monsoon rains have arrived okay, so language model should be able to say that the probability of the sequence of tokens is let us say 0.

2 or 0.3 or 0.4 right. This number is the 0.2, 0.3 is quite high by the way remember this right. But let us say if there is a sentence like this, a sequence of tokens like this monsoon on the have rains arrived right. The models, the language model should be able to say that the probability of this sequence is 0.

001 for example, right, because this is not a valid sentence, okay. This is syntactically incorrect sentence. Now, when you talk about language model, right, we also assume that the language model understands world's knowledge, right. Let's say there is a sequence of tokens like the man eats the cheese. Language model will say that the probability is 0.

8. Let us say there is another sequence of tokens, the cheese eats the man. The language model should be able to say that the probability of the sequence is let us say 0.

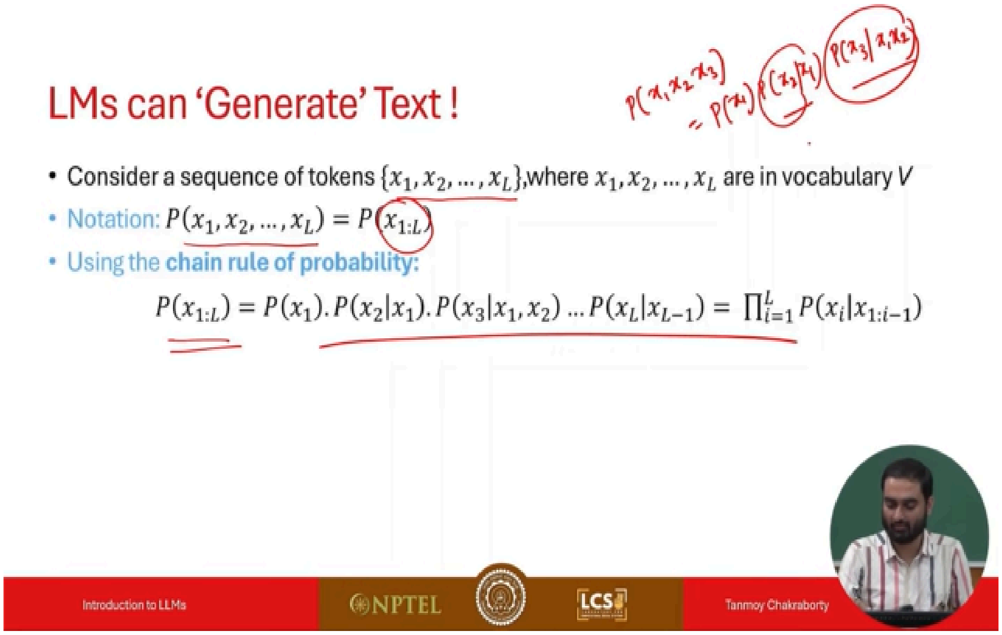
1. Earlier it was 0.8, now this is 0.1 and if let us say the sequence is something like that, the man cheese eats. So then the model will say that the probability is 0.001 right. Now remember the first one was of course that was a valid English sentence but the second was also a valid English sentence right. It was valid in the sense like it was valid in terms of the syntax, in terms of the grammar, there was subject, verb and object right.

But semantically, that was not the right sentence because a cheese can't eat a man right. On the other hand, the last sequence of tokens that I mentioned that is not at all syntactically correct sequence, semantically correct sequence, so therefore the probability is much much lower, right. So you see that when you compare across multiple probabilities generated by language models, you can also think of, you know, which one is syntactically correct, which one is semantically correct, which one is nonsensical and so on and so forth, right. So the hope is that the language model also understands world's knowledge some ways, right. Now when we design a language model, right, what we do, we create a vocabulary, right.

How do we create a vocabulary? Let's say we are given a document, a corpus, remember the term corpus, okay. Corpus is essentially a document. And the plural is corpora. So let's say you're given a corpus. A large corpus, and you start scanning the corpus from the extreme left to the uh extreme left, start to the extreme right end, right So you scan through the document one by one, uh, the tokens one by one and then you essentially, uh, create a vocabulary where we will add all the unique tokens right all the unique tokens into the vocabulary, right the unique tokens are called types, remember this, types Tokens are words in this case.

Let's assume that tokens are words. Tokens can be characters also, but in this case, tokens are words and we add unique tokens or we add words to the vocabulary. In this way, we create a list of unique tokens present in the document and that forms the vocabulary.

(Refer Slide Time: 16:58)



LMs can 'Generate' Text !

$P(x_1, x_2 x_3)$
$= P(x_1) P(x_2|x_1) P(x_3|x_1,x_2)$

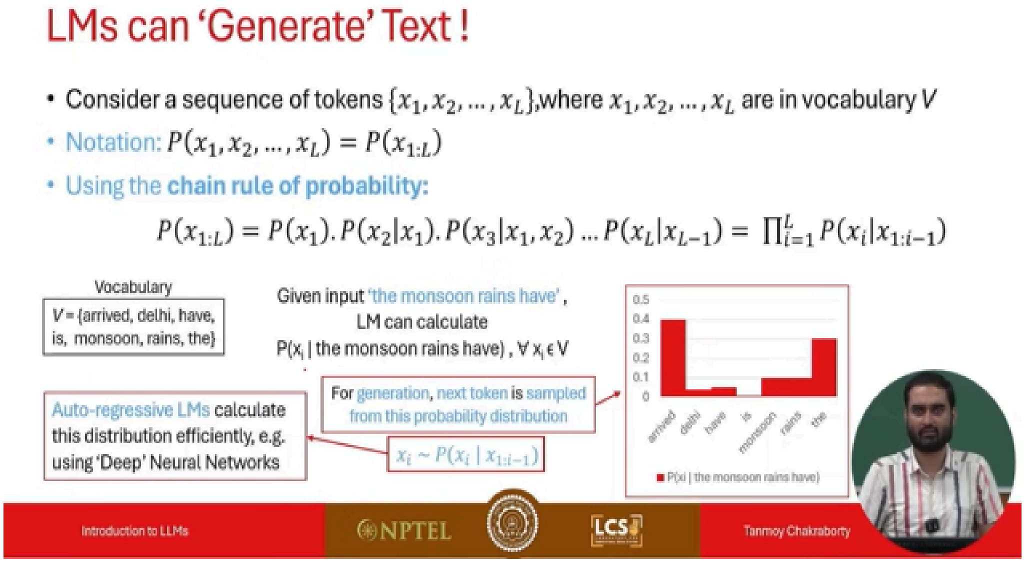
- Consider a sequence of tokens $\{x_1, x_2, \dots, x_L\}$,where $x_1, x_2, \dots, x_L$ are in vocabulary V
- Notation: $P(x_1, x_2, \dots, x_L) = P(x_{1:L})$
- Using the **chain rule of probability:**

$$P(x_{1:L}) = P(x_1).P(x_2|x_1).P(x_3|x_1,x_2) \dots P(x_L|x_{L-1}) = \prod_{i=1}^{L} P(x_i|x_{1:i-1})$$

Okay, let's say in this case the vocabulary words are arrived will, we have is monsoon rains the etc, okay. So, mathematically given a sequence of tokens x1 x2 dot dot dot xL The language model basically measures the probability of the sequence.

So it is a joint probability X1 to L. And how do we measure this joint probability? We measure the joint probability, we can unfold it to basically multiple conditional probabilities using chain rules. We all know that P of X1 x2, x3 can be computed as p of x1 times p of x2 given x1 right times p of x3 given x1, x2 and so on and so forth right as you see here. So, all these components, right these are conditional probabilities, right and we will see in the language model chapter how we will compute this conditional probabilities okay. We will also understand the problems you know in computing, this conditional probabilities and why do we need to adapt large language models for scaling it up okay.

(Refer Slide Time: 18:20)



LMs can 'Generate' Text !

- Consider a sequence of tokens $\{x_1, x_2, \dots, x_L\}$,where $x_1, x_2, \dots, x_L$ are in vocabulary V
- Notation: $P(x_1, x_2, \dots, x_L) = P(x_{1:L})$
- Using the **chain rule of probability:**

$$P(x_{1:L}) = P(x_1).P(x_2|x_1).P(x_3|x_1, x_2) \dots P(x_L|x_{L-1}) = \prod_{i=1}^{L} P(x_i|x_{1:i-1})$$

Vocabulary

V = {arrived, delhi, have, is, monsoon, rains, the}

Given input 'the monsoon rains have', LM can calculate P(x_i | the monsoon rains have) , ∀ x_i ϵ V

For generation, next token is sampled from this probability distribution

Auto-regressive LMs calculate this distribution efficiently, e.g. using 'Deep' Neural Networks

$x_i \sim P(x_i \mid x_{1:i-1})$

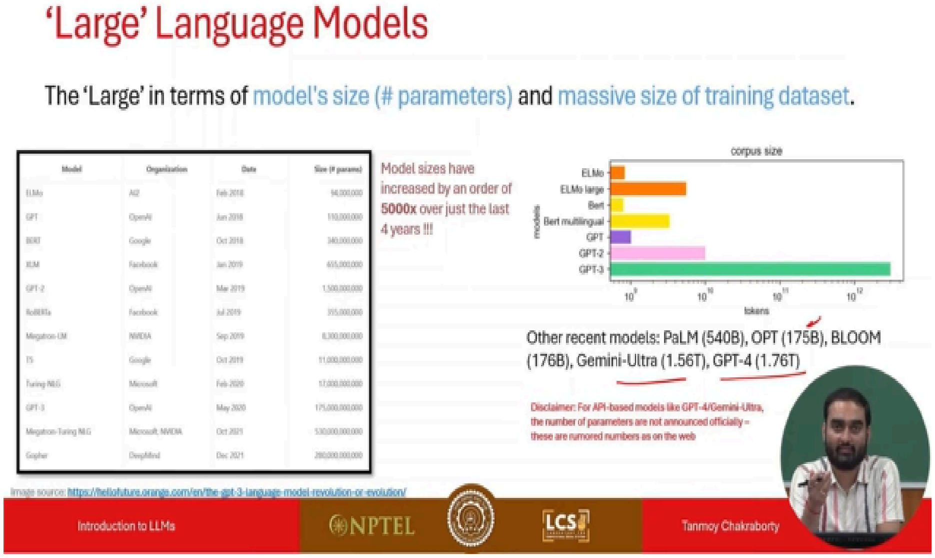
Now, as you see here, although I mentioned that a language model is a probability distribution over a sequence of tokens, when you break it into multiple conditional probabilities, it now becomes some sort of generation problem, right.

Look at this component for example, what does it say? It says that given x1 and x2, given two tokens, given two previous tokens, what is the probability that next word or next token is going to be x3, okay. Again, I will use the term word very vaguely, a word basically refers to a token, okay. So, So, in our example, if the context is the monsoon rains have, right? What is the probability that the next word is going to be, let's say arrived, okay? Or next word is going to be Delhi, next word is going to be have and so on and so forth, right? And how do we do that? We already have a probability distribution, right? How do we compute the distribution? We will discuss in the language model chapter, but you assume that you are given the distribution right, and you choose a token, right, as a placeholder for xi, right, and let's say the token is xi, okay, that you sample from this distribution, okay and you return that token as the next token given the context okay, so the next code token can be can be arrived, the next token can be begun, the next token can be started, right, the next token can also be telling right, and if the language model is good enough, the language model will say that the probability that the next token is going to be arrived or the probability that the next token is going to be started would be higher than the next token you know than let's say the word Delhi or the word let's say rain okay. The monsoon rains have rains doesn't make any sense right okay.

So we will discuss more about this. So this model is also called autoregressive model. Why autoregressive model? Because you are given a context automatically, it will generate the next token in a sequential manner. And we will see in the neural network or in the neural language model chapter that the concept of language model can be modeled using a neural network where you feed the sequence of tokens which are present in the context as an input to the neural model, and the neural model should be able to produce the output with certain distribution, with certain probability, okay. So the entire concept can be, actually modeled or can be mimicked with a neural network.

We will discuss later.

(Refer Slide Time: 21:30)



'Large' Language Models

The 'Large' in terms of model's size (# parameters) and massive size of training dataset.

Model	Organization	Date	Size (# params)
ELMo	AI2	Feb 2018	94,000,000
GPT	OpenAI	Jun 2018	110,000,000
BERT	Google	Oct 2018	340,000,000
XLM	Facebook	Jan 2019	665,000,000
GPT-2	OpenAI	Mar 2019	1,500,000,000
RoBERTa	Facebook	Jul 2019	355,000,000
Megatron-LM	NVIDIA	Sep 2019	8,300,000,000
T5	Google	Oct 2019	11,000,000,000
Turing NLG	Microsoft	Feb 2020	17,000,000,000
GPT-3	OpenAI	May 2020	175,000,000,000
Megatron-Turing NLG	Microsoft, NVIDIA	Oct 2021	530,000,000,000
Gopher	DeepMind	Dec 2021	280,000,000,000

Model sizes have increased by an order of **5000x** over just the last 4 years !!!

Other recent models: PaLM (540B), OPT (175B), BLOOM (176B), Gemini-Ultra (1.56T), GPT-4 (1.76T)

Disclaimer: For API-based models like GPT-4/Gemini-Ultra, the number of parameters are not announced officially – these are rumored numbers as on the web

Image source: https://hellofuture.orange.com/en/the-gpt-3-language-model-revolution-or-evolution/

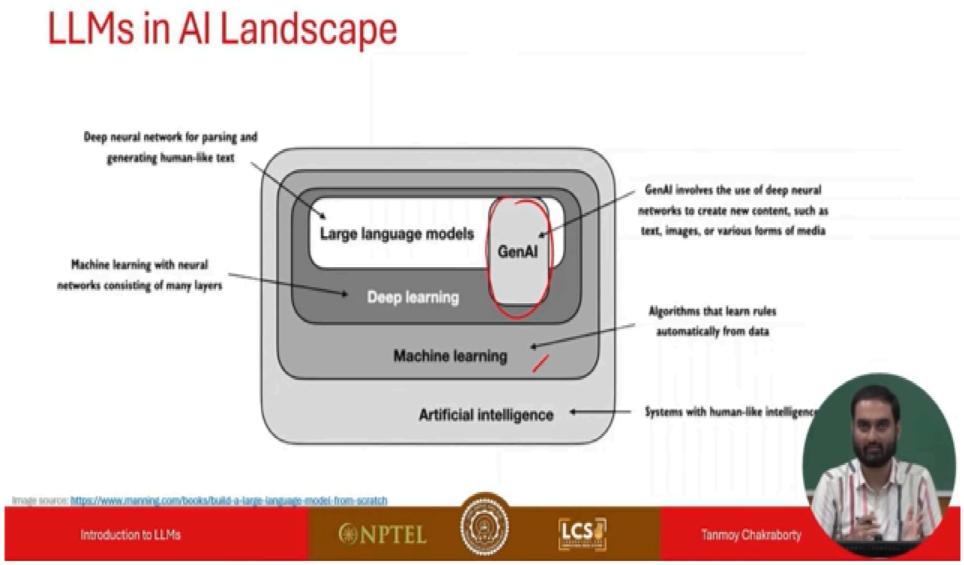
Okay so this is all about language model, we are moving to large language model right. So the term large here, right, can be referred to the parameter size of the model, right or the number of training data set or amount of training data set, amount of tokens present in the training data set that are used to train this kind of model. Okay, so you know transformer was introduced in 2017, right and then during that time another model was introduced called ELMO. We will discuss the ELMO model later, right. And the model parameter was the number of parameters were roughly 94 million in ELMO, right, and then GPT was introduced in 2018 roughly 110 million.

Then bot was introduced, it had around 340 million parameters in 2018 and people started shouting, people said this is too much, 340 million parameters training such a model, loading such a model for inference, this is impossible. And then we saw some of the, of tiny language models, small language models on top of BERT. But then in 2019 GPT-2 came and the parameter size was roughly 1.5 billion and then Megatron language model, Megatron LM, roughly 8.3 billion and then T5 11 billion right, GPT-3 was introduced in 2020 175 billion, Megatron Turing energy was introduced 570 billion and Gopher 280 billion right.

This is the history till 2021 by the way. I will talk about you know the models which are introduced after 2021 later okay. But if you see the parameter size increase right, the parameter size has increased by you know 2000 times, 5000 times roughly in 4 years okay and at the same time, if you look at the size, the corpus size, right, that were used to train the model. ELMO used roughly 1 billion tokens, right, to train the model GPT-2, GPT-3, you know, much, much higher than that. And in the recent models like Palm, OPT, Gemini, GPT-4, we don't know the exact, exact amount of tokens that were there in the training set to train this model. But, you know, from social media, from Twitter, we came to know that roughly the parameter size of GPT-4 is 1.

76 trillion, right, Gemini is 1.56 ultra, is 1.56 trillion roughly, GPT 1.76 trillion, right, OPT 175 billion. OPT is an open source model. So you see that a massive scaling up in terms of number of parameters, in terms of number of tokens used to train such models.

(Refer Slide Time: 24:53)



LLMs in AI Landscape

Deep neural network for parsing and generating human-like text

Large language models

GenAI

GenAI involves the use of deep neural networks to create new content, such as text, images, or various forms of media

Machine learning with neural networks consisting of many layers

Deep learning

Algorithms that learn rules automatically from data

Machine learning

Artificial intelligence

Systems with human-like intelligence

Image source: https://www.manning.com/books/build-a-large-language-model-from-scratch

Introduction to LLMs

NPTEL

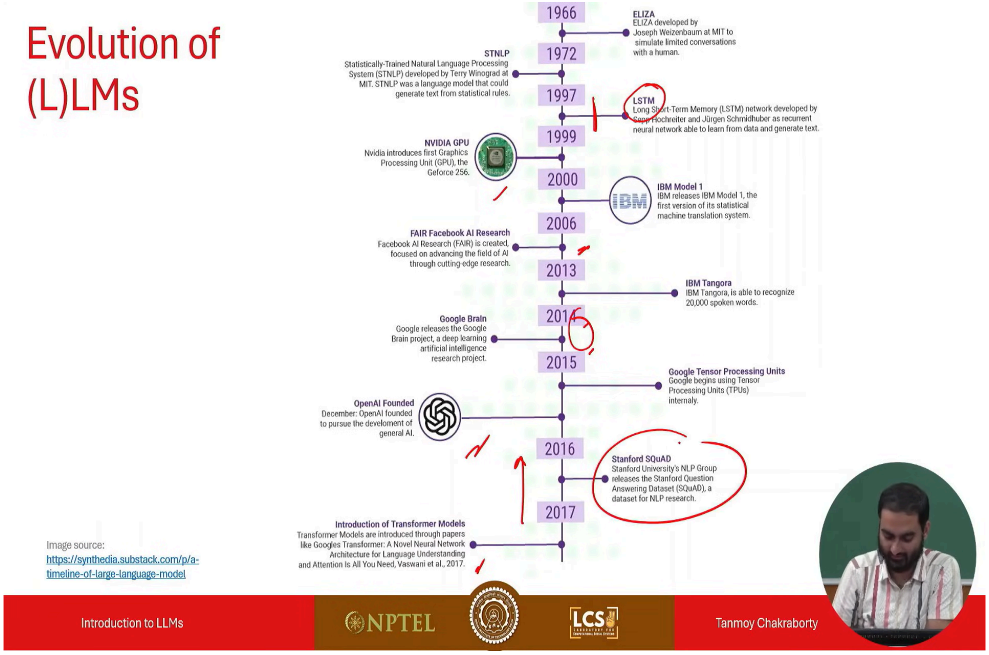
LCS

Tanmoy Chakraborty

Let's look at the overall landscape of AI in today's era. So we have AI, the classical AI, where along with the learning part, we also learn the planning part of it, planning, scheduling, and other parts of it which may not need the learning component as such. And then within AI, We have machine learning where we are interested in learning automated rules from data. Then, within ML we have deep learning, where now, we introduce the concept of neural network, deep neural network, multiple layers, deep layers. And then large language models. Large language model is essentially a deep neural network, right, which has the capability to generate things right.

The traditional deep learning models you know do not have the capacity to generate text, to generate images but today's gen AI models, generative models have the capability to generate languages right. So LLM is essentially a part of deep learning, whereas generative AI is essentially talks about many things apart from text. It also has images, various types of media, speech and so on and so forth.

(Refer Slide Time: 26:25)



Evolution of (L)LMs

1966 — ELIZA: ELIZA developed by Joseph Weizenbaum at MIT to simulate limited conversations with a human.

1972 — STNLP: Statistically-Trained Natural Language Processing System (STNLP) developed by Terry Winograd at MIT. STNLP was a language model that could generate text from statistical rules.

1997 — LSTM: Long Short-Term Memory (LSTM) network developed by Sepp Hochreiter and Jürgen Schmidhuber as recurrent neural network able to learn from data and generate text.

1999 — NVIDIA GPU: Nvidia introduces first Graphics Processing Unit (GPU), the Geforce 256.

2000 — IBM Model 1: IBM releases IBM Model 1, the first version of its statistical machine translation system.

2006 — FAIR Facebook AI Research: Facebook AI Research (FAIR) is created, focused on advancing the field of AI through cutting-edge research.

2013 — IBM Tangora: IBM Tangora, is able to recognize 20,000 spoken words.

2014 — Google Brain: Google releases the Google Brain project, a deep learning artificial intelligence research project.

2015 — Google Tensor Processing Units: Google begins using Tensor Processing Units (TPUs) internaly.

2015 — OpenAI Founded: December: OpenAI founded to pursue the develoment of general AI.

2016 — Stanford SQuAD: Stanford University's NLP Group releases the Stanford Question Answering Dataset (SQuAD), a dataset for NLP research.

2017 — Introduction of Transformer Models: Transformer Models are introduced through papers like Googles Transformer: A Novel Neural Network Architecture for Language Understanding and Attention Is All You Need, Vaswani et al., 2017.

Image source: https://synthedia.substack.com/p/a-timeline-of-large-language-model

So now let us look at the evolution of language models, evolution of large language models. So if you think of language model, Let us go back to the year 1966 or 1967 during that time.

The famous model ELISA was introduced by MIT lab. ELISA was a simple chat bot which was able to interact with humans. It was basically operated with a set of rules. Rules were predefined. There was not a, there was no learning component as such within ELISA. In 1996-1997 LSTM was introduced, RNN was introduced much before that.

LSTM was introduced, LSTM, GRU, you all know that these are basically, some sort of gated mechanisms right on top of RNN which would address the vanishing gradient problem. We will discuss later. NVIDIA you know came into the picture in 1999-2000. And then they started producing their GPUs, graphics processing unit, which is essentially the hardware that we use these days to train such gigantic models, right. IBM has always been one of the pioneers, one of the major figures in AI.

All of you know the IBM Watson model. In 2011, IBM Watson basically, defeated the human participants in a quiz game called Geopaddy Challenge. It's a TV show. And then you know around 2006, 2007 Facebook introduced their lab called Facebook AI research lab, FAIR lab, right. In 2014, 2015 during that time Google released their Google brain project, okay. And Google also started producing their own hardware called TPUs, tensor processing unit, okay.

And then in 2017, Google Brain, Ashish Basbani and his colleagues from Google Brain, they introduced this model called Transformer. Slightly before the time when Transformer was introduced, OpenAI another company right at that time it was a very small company right that basically came into the picture in 2015, 2016 and then in 2016 Stanford released this data set called the squad data set for question answering. So this is the history till 2017. We are in 2017 now and transformer has just been introduced by Google brain.

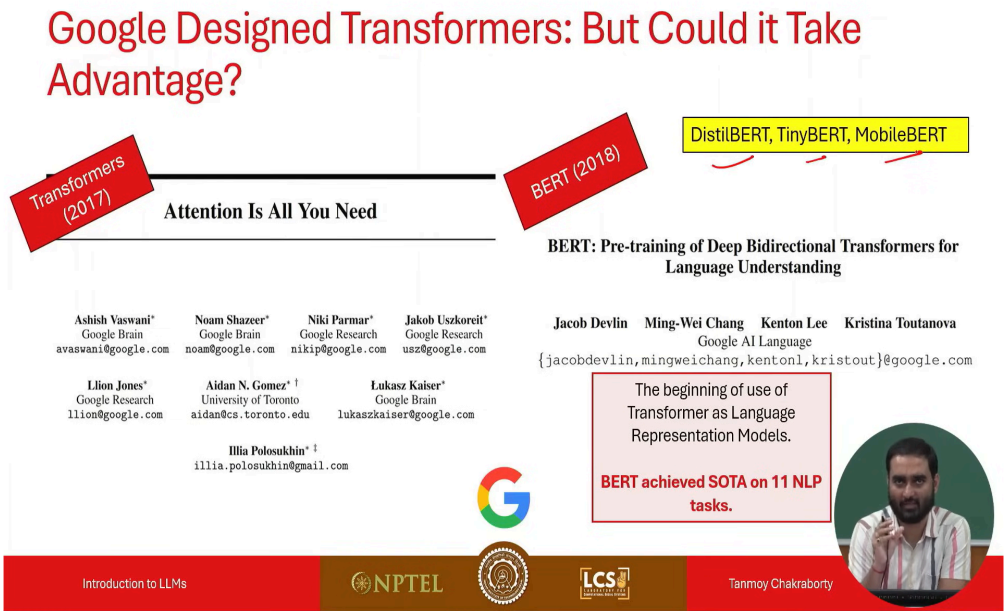
(Refer Slide Time: 29:34)

Post-Transformers Era

The LLM Race

We are moving to 2018. Let's see how post-transformer era looks like.

(Refer Slide Time: 29:40)



Google Designed Transformers: But Could it Take Advantage?

Transformers (2017)

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

BERT (2018)

DistilBERT, TinyBERT, MobileBERT

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova

Google AI Language

{jacobdevlin,mingweichang,kentonl,kristout}@google.com

The beginning of use of Transformer as Language Representation Models.

BERT achieved SOTA on 11 NLP tasks.

Introduction to LLMs NPTEL LCS Tanmoy Chakraborty

So this is the paper, the transformer paper, the famous paper. I don't know how many hits this paper has received so far. So Google continued their journey. In 2018, they introduced this famous BERT model, bi-directional encoder model.

This is essentially a new architecture, a new training protocol. They introduced the concept of mask language model. They just focus on the encoder part of transformer. We will discuss the transformer model in details. Essentially in transformer there are two blocks. One is called the encoder block, other is called the decoder block, and encoder decoder blocks are linked.

So BERT only focused on the encoder block of transformer model and with the concept of mask language model, so this model was trained. Then although the transformer model was introduced for machine translation task, specifically for machine translation task, but here showed superior performance across 11 NLP tasks and different types of task in NLP. So as I mentioned earlier, that time people thought that this is too big, a 300 million parameter model. Let us try to think of a tiny model.

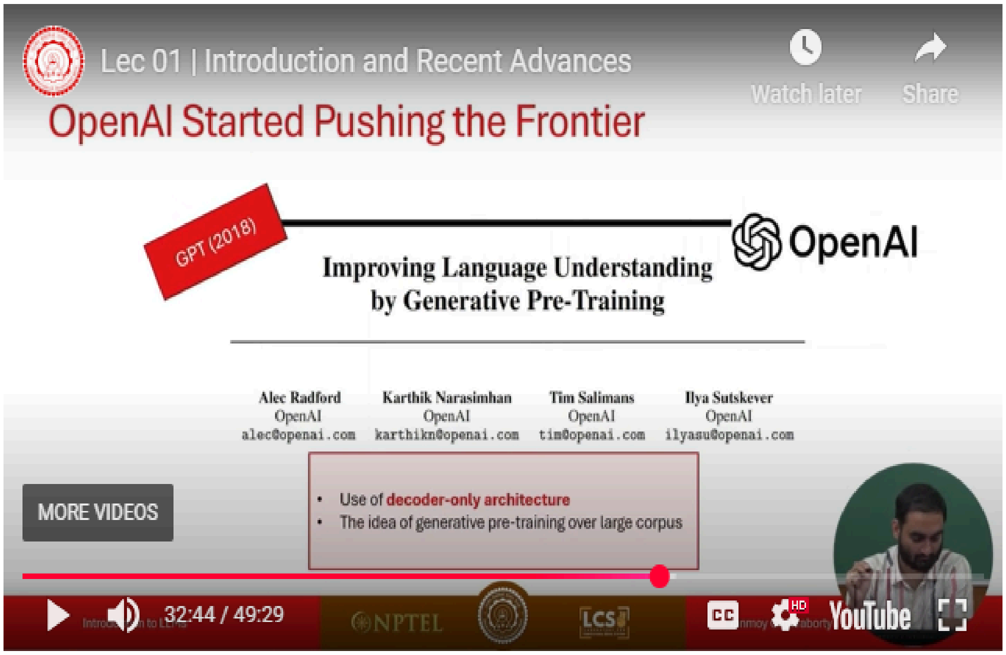
People proposed a digital bot, tiny bot, mobile bot. These are all small models based on the bot architecture.

(Refer Slide Time: 31:19)

However, someone was waiting for the right opportunity!!

In 2018, there was another company who was just waiting for the right opportunity, right? And it is none other than OpenAI.

(Refer Slide Time: 31:30)



Lec 01 | Introduction and Recent Advances

OpenAI Started Pushing the Frontier

GPT (2018)

Improving Language Understanding by Generative Pre-Training

Alec Radford, OpenAI, alec@openai.com
Karthik Narasimhan, OpenAI, karthikn@openai.com
Tim Salimans, OpenAI, tim@openai.com
Ilya Sutskever, OpenAI, ilyasu@openai.com

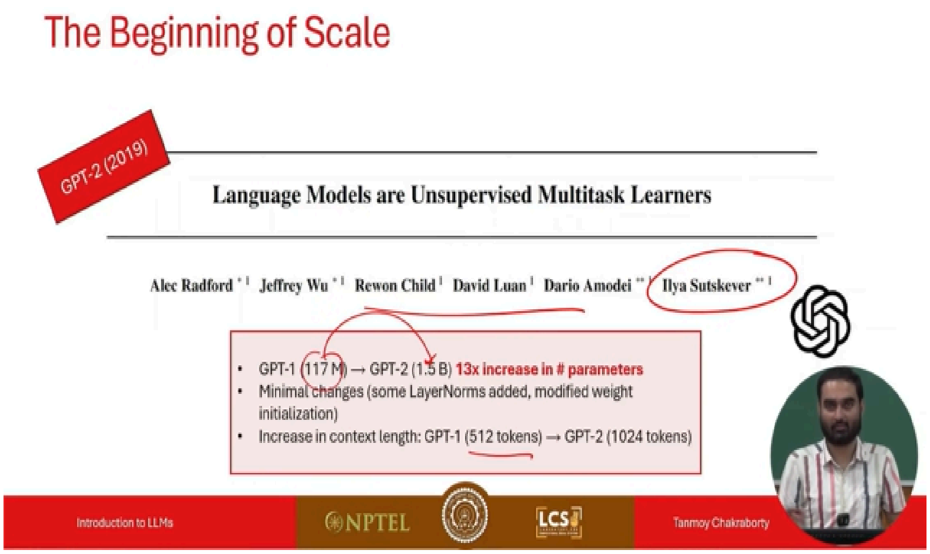
- Use of decoder-only architecture
- The idea of generative pre-training over large corpus

So OpenAI in 2018, they wrote their first GPT paper, right? And remember this gentleman, Ilya Sutskever, right? So he is one of the co-founders of OpenAI, right? He is the chief architect of OpenAI. He was the chief architect because he recently resigned. Some of you know Ilya because he was also the co-authors of this famous AlexNet paper, right? Ilya was Jeff Hilton's PhD student.

So in 2012, I guess, 2012-2013, they wrote this AlexNet paper, and then he started with Sam Altman and other members, other founders. This OpenAI company was started in 2015-2016, I guess. This is the first GPT paper and in this paper they only focused on the decoder part of it. Remember encoder decoder but was only based on the encoder model, encoder layer.

GPT only focused on the decoder part of it.

(Refer Slide Time: 32:45)



The Beginning of Scale

GPT-2 (2019)

Language Models are Unsupervised Multitask Learners

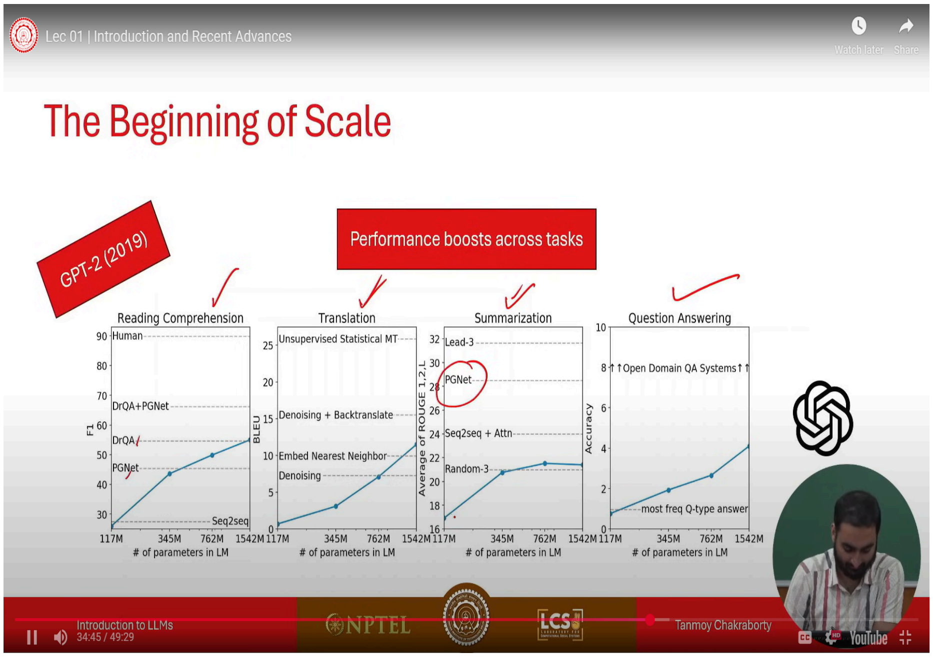
Alec Radford *[1] Jeffrey Wu *[1] Rewon Child [1] David Luan [1] Dario Amodei ** [1] Ilya Sutskever ** [1]

- GPT-1 (117 M) → GPT-2 (1.5 B) **13x increase in # parameters**
- Minimal changes (some LayerNorms added, modified weight initialization)
- Increase in context length: GPT-1 (512 tokens) → GPT-2 (1024 tokens)

Just one year after that, 2019, GPT-2 was introduced. The same guy, same set of people. The beauty about GPT-2 is that the architecture is more or less same. It's a decoder-only model.

But the parameter size has now jumped from 17 million to 1.5 billion, a 13x kind of increase. And now the context length, the length of the input that the model can process is now 1024 tokens, earlier it was 512 tokens.

(Refer Slide Time: 33:25)



The Beginning of Scale

GPT-2 (2019)

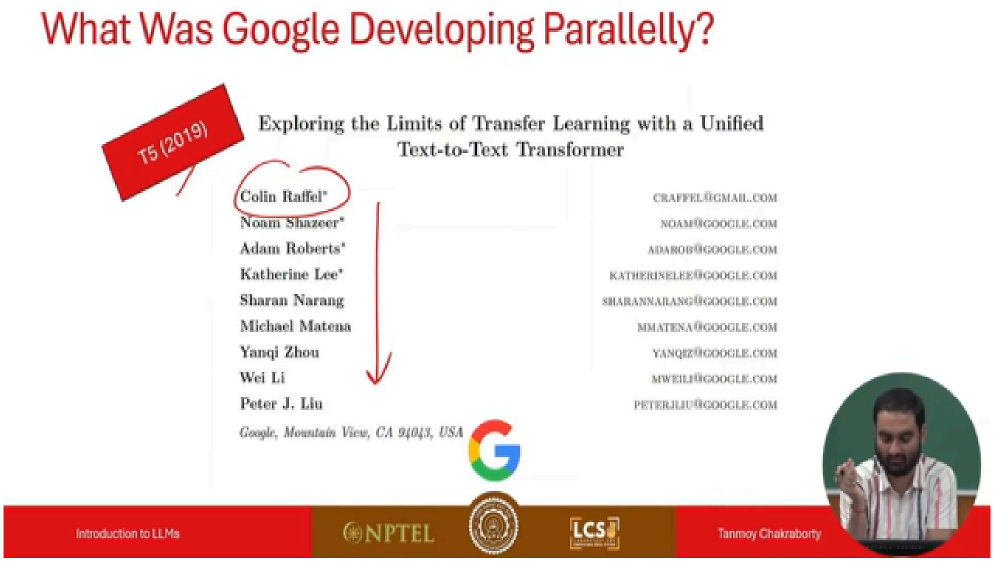
Performance boosts across tasks

In GPT-2, they observed a very interesting idea of prompting. They showed that they came up with the concept of zero shot, few shot and so on and so forth. They showed that a pre-trained model, pre-trained on gigantic corpus, on a downstream task, let us say the task is translation or summarization, on a downstream task without knowing the training data, the model performs significantly well in a zero-shot setting.

Zero-shot means without knowing about the task or about any examples related to the task and so on. If you look at the results across different tasks, reading comprehension, translation, summarization, question answering, you see this blue line which indicates the performance of GPT-2. The performance is significantly is comparable compared to the other expert models. These expert models are essentially trained for this particular task. For example, this PGNet, pointer generator network, this is trained for the summarization task.

But GPT-2 is not trained for this task. In a zero-shot setting during inference time, the model performs significantly well.

(Refer Slide Time: 34:56)



What Was Google Developing Parallelly?

T5 (2019)

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Colin Raffel*	CRAFFEL@GMAIL.COM
Noam Shazeer*	NOAM@GOOGLE.COM
Adam Roberts*	ADAROB@GOOGLE.COM
Katherine Lee*	KATHERINELEE@GOOGLE.COM
Sharan Narang	SHARANNARANG@GOOGLE.COM
Michael Matena	MMATENA@GOOGLE.COM
Yanqi Zhou	YANQIZ@GOOGLE.COM
Wei Li	MWEILI@GOOGLE.COM
Peter J. Liu	PETERJLIU@GOOGLE.COM

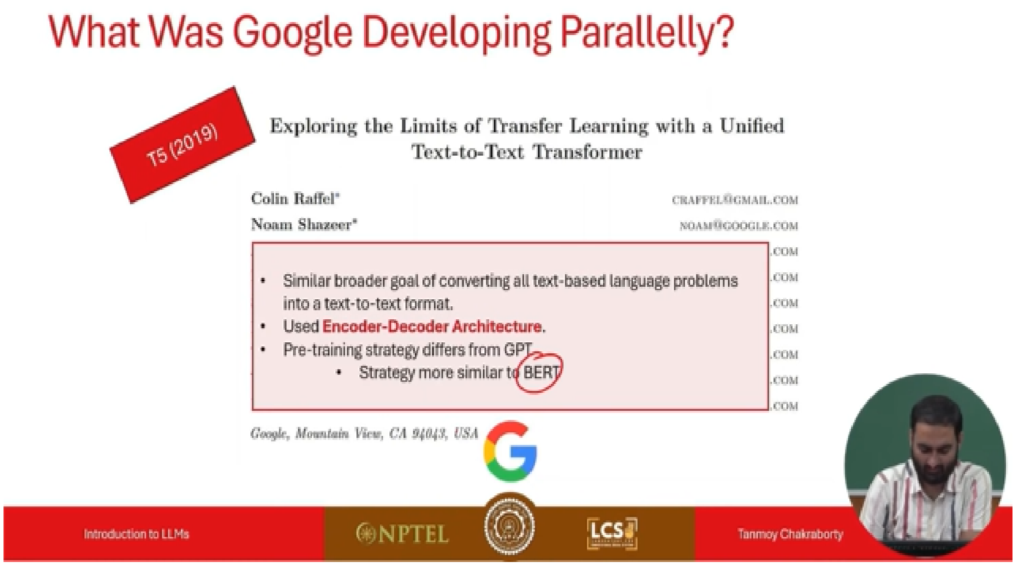
Google, Mountain View, CA 94043, USA

Then the third that the GPD guy, the open air people, they thought let us scale up further. This is 1.5 billion, let us scale it up further and let us see what happens, right.

So, now they took some more time but in between, in 2019. Google is Google, right? So Google after introducing the transformer and bot, they kept on exploring the other avenues and they proposed this model called T5. This is again a billion size model. Colin Raffel and his colleagues from Google, they proposed the idea of T5. T5 is text-to-text transform, text-to-text transfer transformer, right? We'll discuss about T5 later.

So it's an encoder-decoder model. Unlike what which is an encoder model or GPT, which is a decoder-only model, this is an encoder-decoder model, right. But in this model they unified all the tasks as a text-to-text problem where a text is an input and a text is an output. It can be a classification task, it can be a regression task, a number is also considered as a text, right.

(Refer Slide Time: 36:17)



What Was Google Developing Parallelly?

T5 (2019)

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

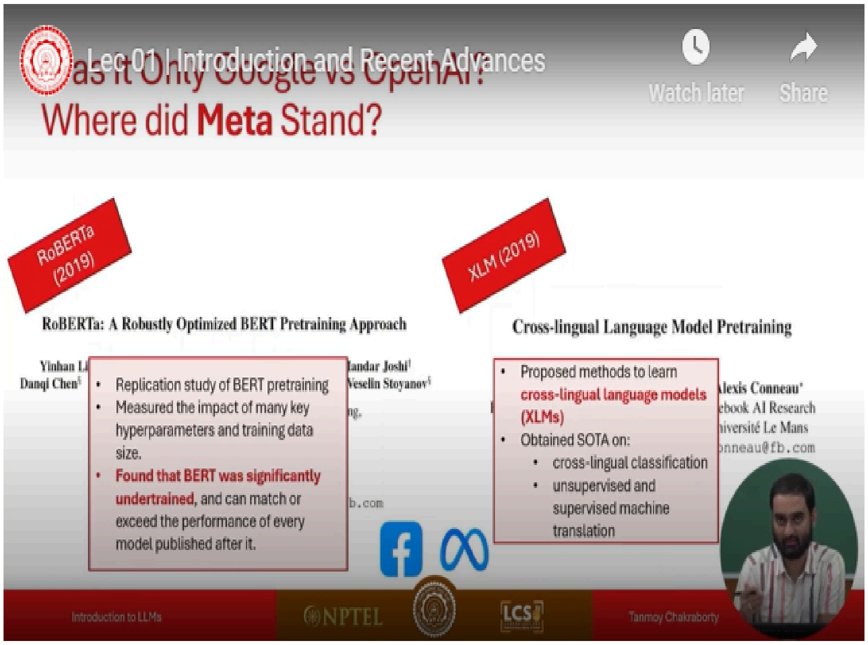
Colin Raffel* CRAFFEL@GMAIL.COM
Noam Shazeer* NOAM@GOOGLE.COM

- Similar broader goal of converting all text-based language problems into a text-to-text format.
- Used **Encoder-Decoder Architecture.**
- Pre-training strategy differs from GPT
  - Strategy more similar to BERT

Google, Mountain View, CA 94043, USA

In T5 we have same kind of same strategy, training strategy that was used in what kind of model, but here they introduce some more, you know, some advanced strategies like span corruption and so on and so forth for training, we will discuss later.

(Refer Slide Time: 36:36)



Was it Only Google vs OpenAI?
Where did **Meta** Stand?

RoBERTa (2019)

RoBERTa: A Robustly Optimized BERT Pretraining Approach

Yinhan Li
Danqi Chen§
andar Joshi†
Veselin Stoyanov§

- Replication study of BERT pretraining
- Measured the impact of many key hyperparameters and training data size.
- **Found that BERT was significantly undertrained**, and can match or exceed the performance of every model published after it.

XLM (2019)

Cross-lingual Language Model Pretraining

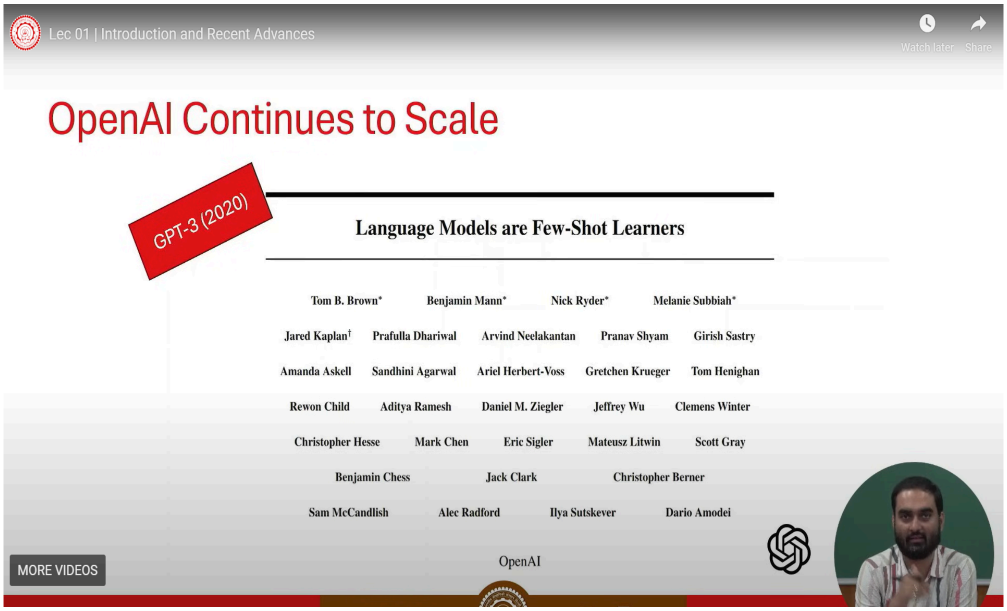
lexis Conneau*
ebook AI Research
niversité Le Mans
onneau@fb.com

- Proposed methods to learn **cross-lingual language models (XLMs)**
- Obtained SOTA on:
  - cross-lingual classification
  - unsupervised and supervised machine translation

Right, they also came into the picture into the race 2019, they proposed the idea of RoBERTa, RoBERTa of you know, it's a advanced method, it's a follow-up of bot, right, and this paper they understood they showed that the bot model is significantly undertrained, right, with more and more training, can we boost the performance of what kind of model, right.

In 2019 they also proposed the XLM model, the follow up of XLM all of you know it is a cross lingual model which has both supervised and unsupervised kind of, I mean it has applications in both supervised and unsupervised machine translation.

(Refer Slide Time: 37:28)



OpenAI Continues to Scale

GPT-3 (2020)

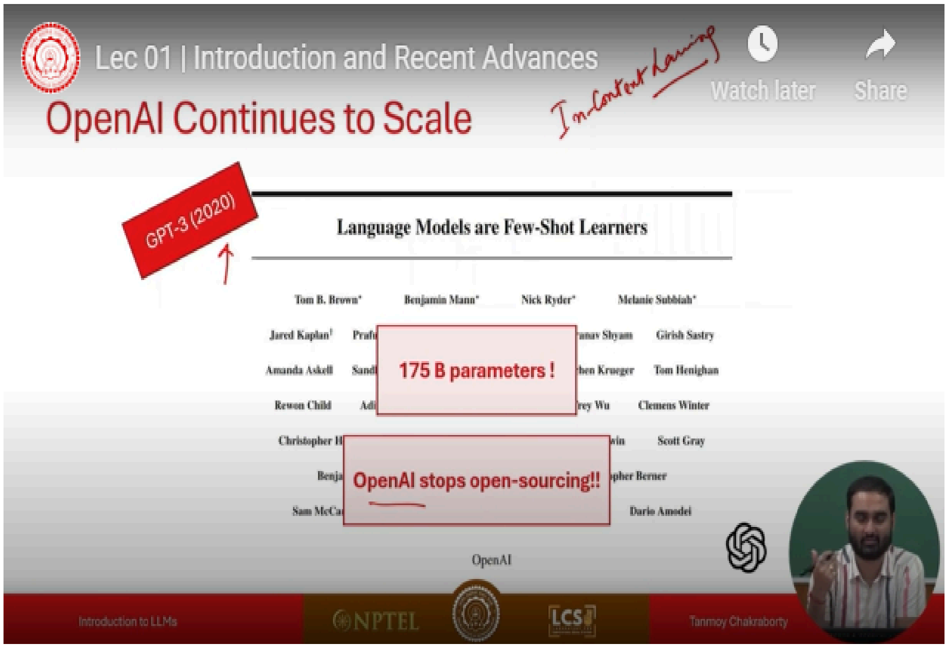
Language Models are Few-Shot Learners

Tom B. Brown* Benjamin Mann* Nick Ryder* Melanie Subbiah*
Jared Kaplan† Prafulla Dhariwal Arvind Neelakantan Pranav Shyam Girish Sastry
Amanda Askell Sandhini Agarwal Ariel Herbert-Voss Gretchen Krueger Tom Henighan
Rewon Child Aditya Ramesh Daniel M. Ziegler Jeffrey Wu Clemens Winter
Christopher Hesse Mark Chen Eric Sigler Mateusz Litwin Scott Gray
Benjamin Chess Jack Clark Christopher Berner
Sam McCandlish Alec Radford Ilya Sutskever Dario Amodei

OpenAI

Right after GPT-2, GPT-3 was introduced. And in this paper, first time, we saw a term called large language model, GPT-3.

(Refer Slide Time: 37:40)



Lec 01 | Introduction and Recent Advances

OpenAI Continues to Scale

In-Context Learning

GPT-3 (2020)

Language Models are Few-Shot Learners

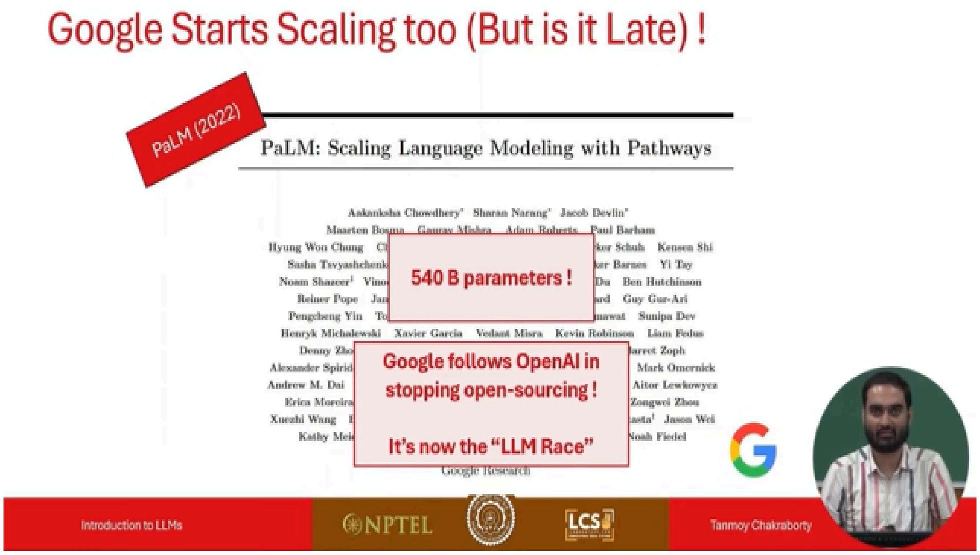
175 B parameters !

OpenAI stops open-sourcing!!

So from 1.7 billion to 175 billion. Huge, huge jump in terms of parameters. And in this paper, they also observed a very interesting phenomena called in context learning. So in context learning the idea is that during the inference time you give some examples for a task, for a downstream task and without any gradient update within the neural network the model should be able to understand or the model should be able to cater to the examples and perform significantly well in the downstream task. This in context learning is essentially an emerging property of the scaling law. Scaling law says that you just increase the size of, you keep on increasing the size of the models and you see the model should be able to perform better and better. The fun part is that from 2020 OpenAI stopped sharing their codes, their tools, their checkpoints and so on.

So OpenAI is no longer an open model, right. OpenAI stopped sharing models, stopped sharing details of the parameters, sharing protocols and so on and so forth.

(Refer Slide Time: 39:03)



Google Starts Scaling too (But is it Late) !

PaLM (2022)

PaLM: Scaling Language Modeling with Pathways

Aakanksha Chowdhery* Sharan Narang* Jacob Devlin*
Maarten Bosma Gaurav Mishra Adam Roberts Paul Barham
Hyung Won Chung Cl ker Schuh Kensen Shi
Sasha Tsvyashchenk ker Barnes Yi Tay
Noam Shazeer[1] Vino Du Ben Hutchinson
Reiner Pope Jan ard Guy Gur-Ari
Pengcheng Yin To mawat Sunipa Dev
Henryk Michalewski Xavier Garcia Vedant Misra Kevin Robinson Liam Fedus
Denny Zho Barret Zoph
Alexander Spirid Mark Omernick
Andrew M. Dai Aitor Lewkowycz
Erica Moreira Zongwei Zhou
Xuezhi Wang E asta[1] Jason Wei
Kathy Mei Noah Fiedel

540 B parameters !

Google follows OpenAI in stopping open-sourcing !

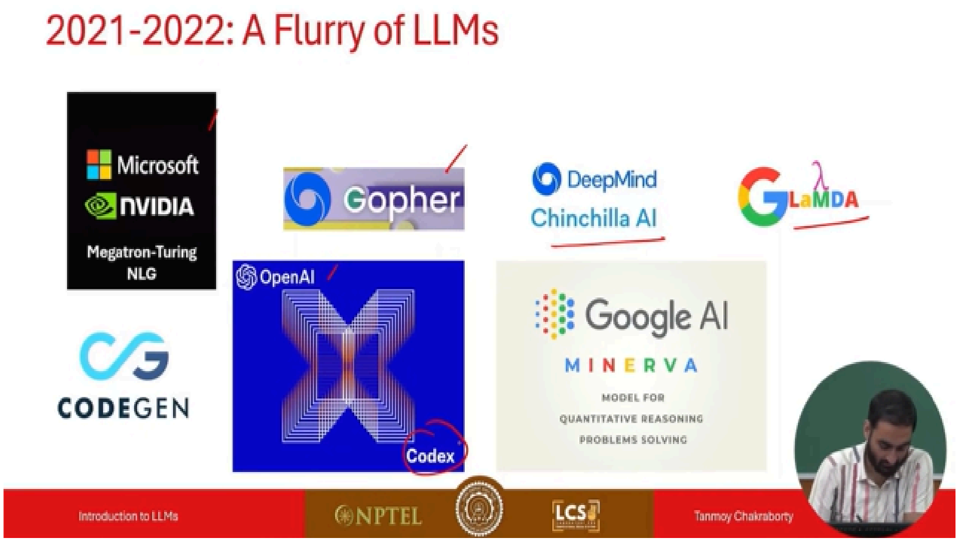
It's now the "LLM Race"

Google Research

In 2022, Google again came up with the idea of Palm. It's a very interesting idea. It was introduced mostly for conversational setting, 540 billion parameters. Google also stopped sharing their codes, their source codes, their checkpoints, and model details, and so on and so forth.

So the race has already begun.

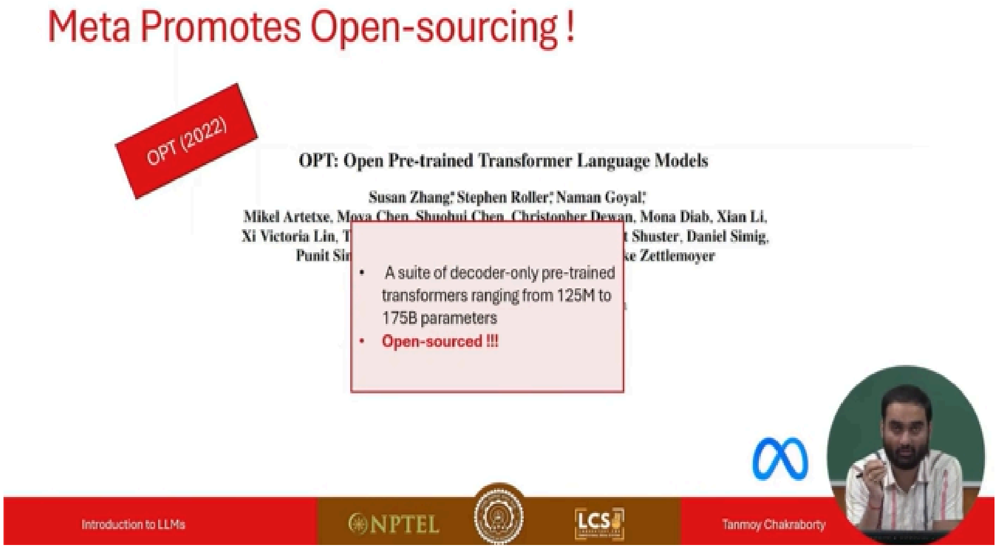
(Refer Slide Time: 39:32)



2021-2022: A Flurry of LLMs

2021, 2022, we saw a bunch of models, models like Megatron Turing by Microsoft and Nvidia collaboration. We also saw models from DeepMind like Chinchilla AI. Google also introduced the Lambda model.

DeepMind also proposed a model called Groffer. OpenAI introduced their first language model for codes. Remember, handling code has its own problem because the major problem is long context. There are other problems also. OpenAI introduced the Codex model. CodeGen was also introduced, and many other models were introduced in between.



Meta Promotes Open-sourcing !

OPT (2022)

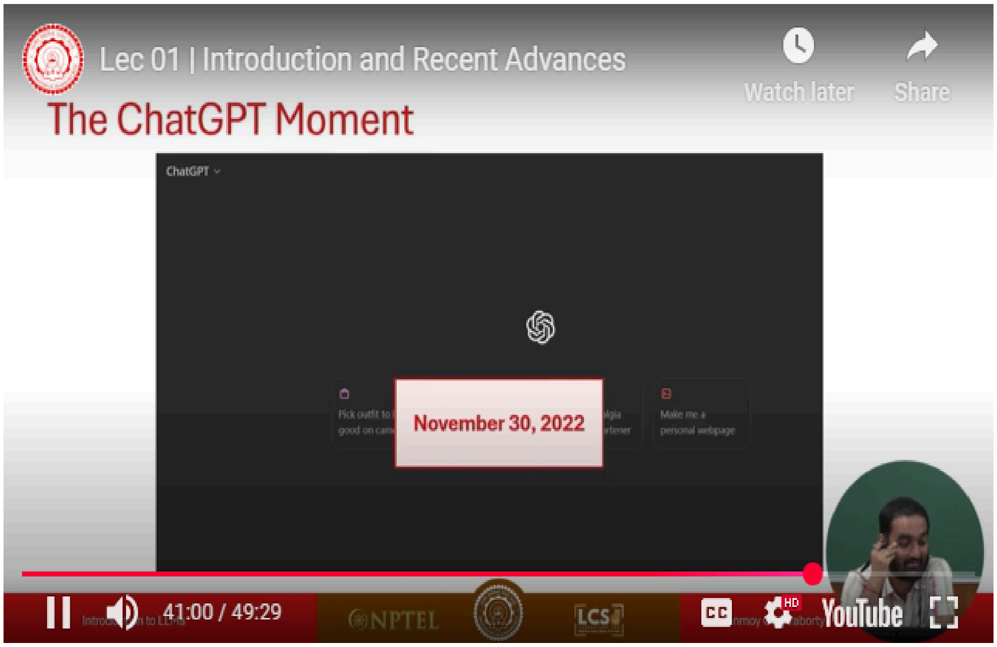
OPT: Open Pre-trained Transformer Language Models

Susan Zhang* Stephen Roller* Naman Goyal*
Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li,
Xi Victoria Lin, T t Shuster, Daniel Simig,
Punit Si ke Zettlemoyer

- A suite of decoder-only pre-trained transformers ranging from 125M to 175B parameters
- Open-sourced !!!

Meta thought that since OpenAI and Google stopped promoting open science, open sourcing, let us only focus on open sourcing.

So they proposed the idea of OPT. This is basically a suite of decoder only model from 125 million to 175 billion parameters. And these are open sourced.

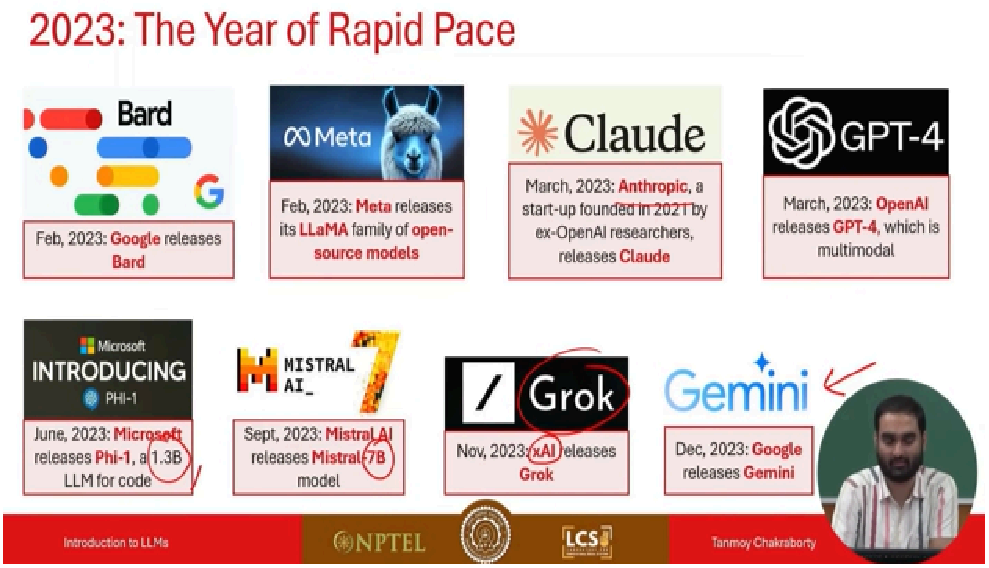


The ChatGPT Moment

November 30, 2022

And then in 2022, November, this happened.

ChatGPT came to our life and the rest is history. So we started playing with ChatGPT. We knowingly, unknowingly fed lots of our data to ChatGPT examples. queries, passages, text, and those data actually started moving to ChatGPT's account freely.



2023: The Year of Rapid Pace

Feb, 2023: Google releases Bard

Feb, 2023: Meta releases its LLaMA family of open-source models

March, 2023: Anthropic, a start-up founded ~~in 2021~~ by ex-OpenAI researchers, releases Claude

March, 2023: OpenAI releases GPT-4, which is multimodal

June, 2023: Microsoft releases Phi-1, a 1.3B LLM for code

Sept, 2023: Mistral AI releases Mistral-7B model

Nov, 2023: xAI releases Grok

Dec, 2023: Google releases Gemini

Introduction to LLMs

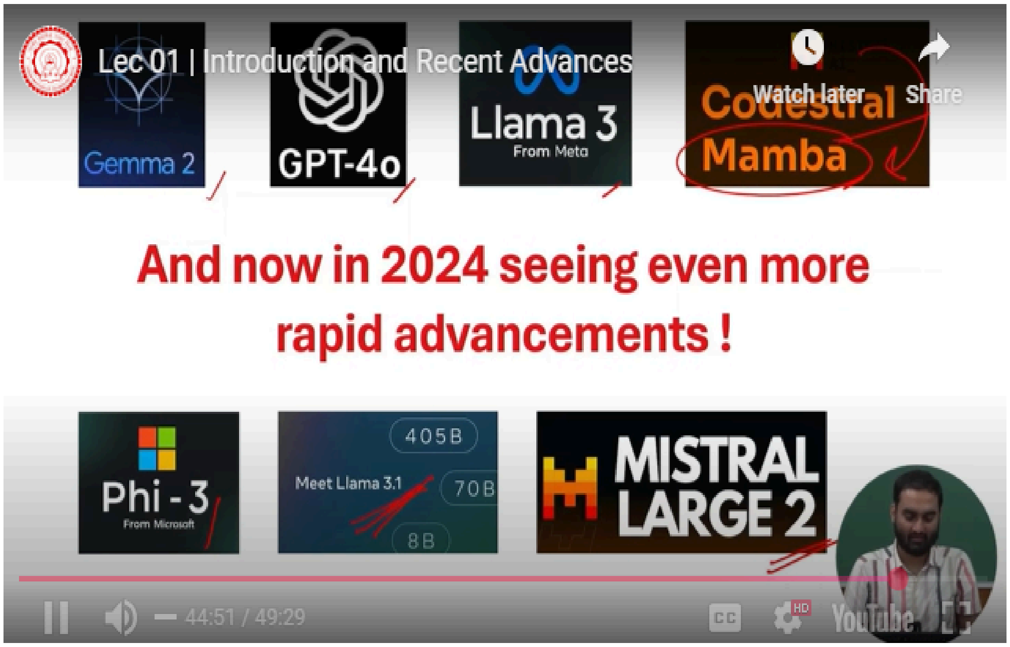
Tanmoy Chakraborty

2023, the race has continued.

February 2023, Google released their bot model. Meta kept on basically producing their open source models. The Lama series started, Lama 1, Lama 2, Lama 3, recently Lama 3.1. Another company came into the picture, Anthropic.

Anthropic started building their claude model. In March 2023, GPT-4 was introduced. It's the first multimodal model. The input may be a text, input may be an image, and so on.

Microsoft built a tiny model called PHY-1, a PHY series model. PHY-1 which is a 1.7 billion parameter model. Mistral, another very interesting company from France, they also came into the picture and they started focusing on small models right at the beginning. So, Mistral 7 billion model is very good in terms of reasoning right. Elon Musk company XAI they also build a model called Grok model and then late 2023 Google unified all their bot models and they introduced this new framework, this unified framework called Gemini. Gemini is again a closed source model or black box model like OpenAI model CPT.

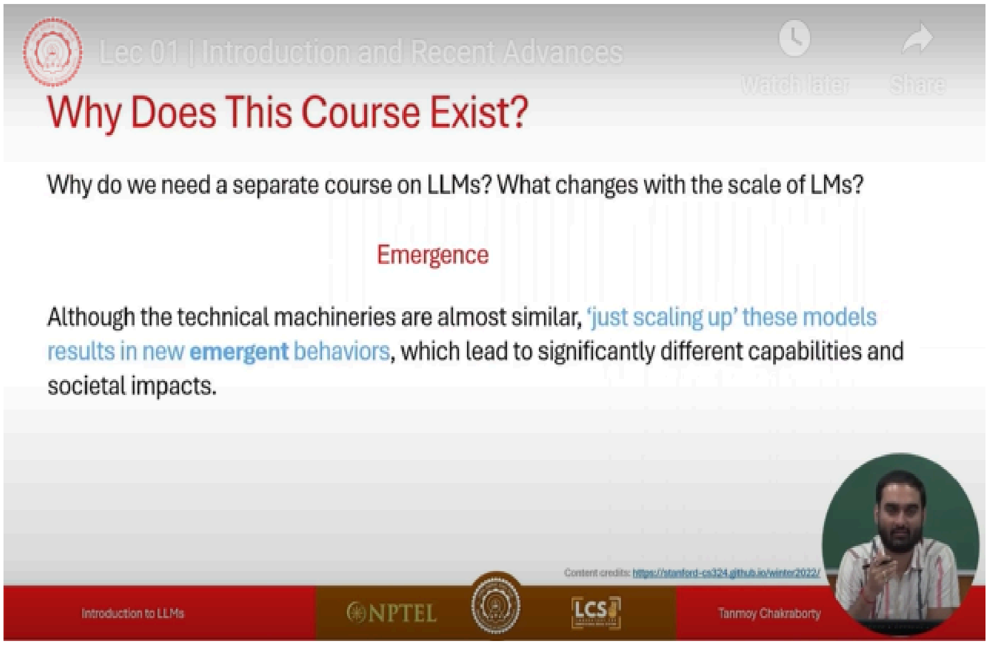

And now in 2024 seeing even more rapid advancements !

This is 2024 and I am recording this lecture today. Today is 26th of July. Last 6-7 months we have already seen a massive improvement in terms of models. The follow-up of Gemini, Gemma 2 and then GPT-4 also released their GPT, OpenAI released their advanced GPT model which is GPT-4-O. Recently they also released GPT-4-O Mini, right.

LAMA-3 was introduced recently. Mistral started collaborating with Mamba team. So Mamba is again a team which believes in non-transformer based model. We will talk about Mamba, maybe at the end of the class, at the end of the course, right. It's a state space based model, right. Mistral collaborated with Mamba and then they introduced this model called Corstral.

Phi 3 was introduced by Microsoft. Recently LAMA 3.1 was introduced. Mistral introduced their new model which is a Mistral to a large 2 model, right. It basically handles 80 plus languages, 80 plus coding languages and 8 different languages, French, German, etc., right. So this mixture large 2 has already been compared with LAMA 3.

1, and it showed reasonable performance, reasonable comparison with LAMA 3.1. LAMA 3.1 if you look at the largest version, it has more than 400 billion parameters. It's the largest open source model Facebook has produced so far.



Why Does This Course Exist?

Why do we need a separate course on LLMs? What changes with the scale of LMs?

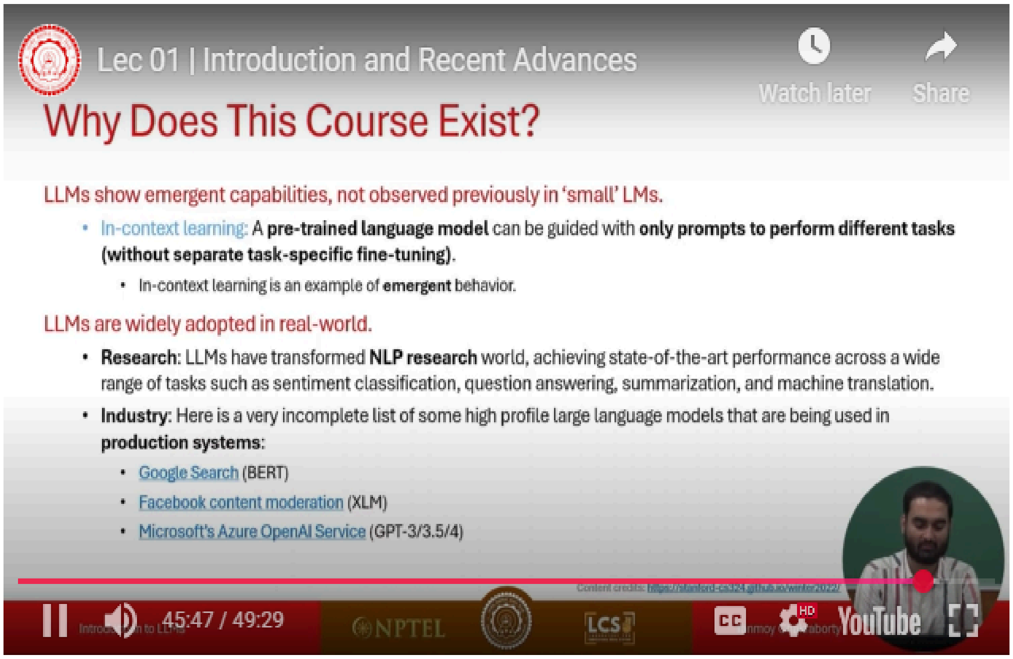
Emergence

Although the technical machineries are almost similar, 'just scaling up' these models results in new **emergent** behaviors, which lead to significantly different capabilities and societal impacts.

Content credits: https://stanford-cs324.github.io/winter2022/

Introduction to LLMs NPTEL Tanmoy Chakraborty

This course exists because the entire journey is not only about scaling law, this is also about emergence, different emerging properties. We will talk about prompting, in context learning, different types of prompting techniques, how with small gradient update we can perform significantly well across different tasks and so on and so forth.



Why Does This Course Exist?

LLMs show emergent capabilities, not observed previously in 'small' LMs.

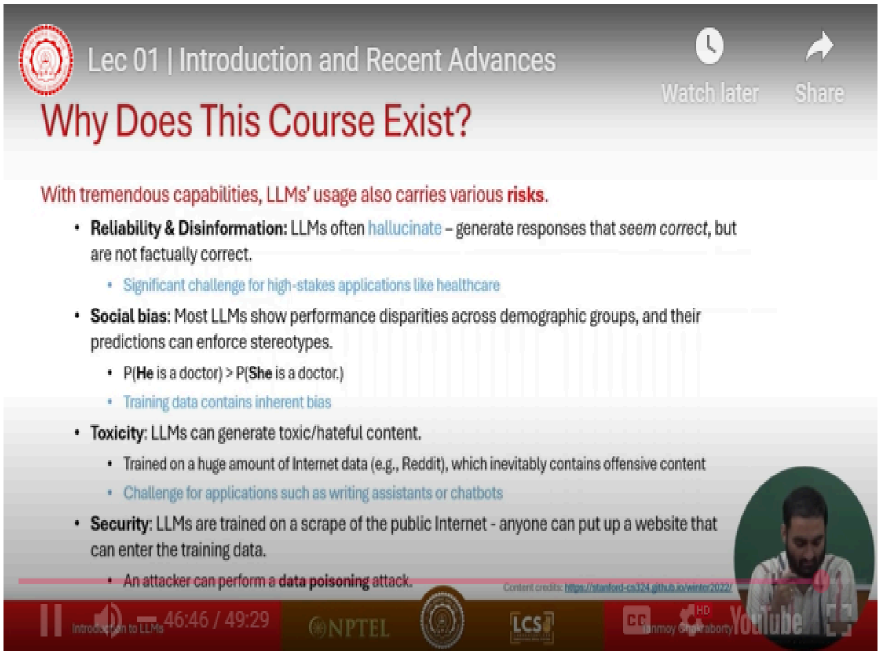
- In-context learning: A **pre-trained language model** can be guided with **only prompts to perform different tasks (without separate task-specific fine-tuning)**.
  - In-context learning is an example of **emergent** behavior.

LLMs are widely adopted in real-world.

- **Research**: LLMs have transformed **NLP research** world, achieving state-of-the-art performance across a wide range of tasks such as sentiment classification, question answering, summarization, and machine translation.
- **Industry**: Here is a very incomplete list of some high profile large language models that are being used in **production systems**:
  - Google Search (BERT)
  - Facebook content moderation (XLM)
  - Microsoft's Azure OpenAI Service (GPT-3/3.5/4)

Content credits: https://stanford-cs324.github.io/winter2022/

And LLM models like LLM, GPT, this kind of models have already been incorporated by different industries. So if you are also interested in understanding how these models have been integrated with let's say Google search, Microsoft Azure service, Microsoft Bing service, so this course is important for that.



Why Does This Course Exist?

With tremendous capabilities, LLMs' usage also carries various **risks**.

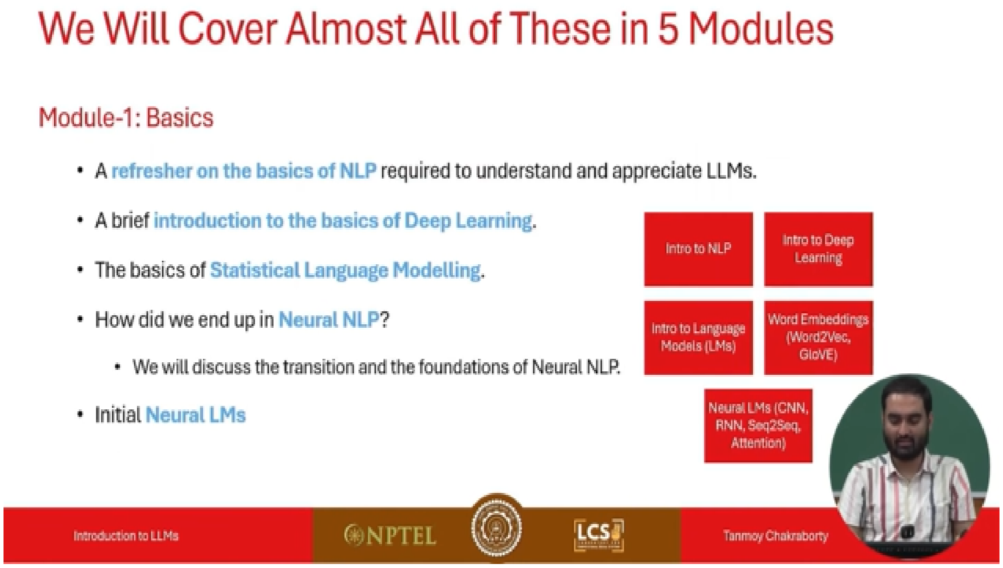
- **Reliability & Disinformation:** LLMs often hallucinate – generate responses that *seem correct*, but are not factually correct.
  - Significant challenge for high-stakes applications like healthcare
- **Social bias**: Most LLMs show performance disparities across demographic groups, and their predictions can enforce stereotypes.
  - P(**He** is a doctor) > P(**She** is a doctor.)
  - Training data contains inherent bias
- **Toxicity**: LLMs can generate toxic/hateful content.
  - Trained on a huge amount of Internet data (e.g., Reddit), which inevitably contains offensive content
  - Challenge for applications such as writing assistants or chatbots
- **Security**: LLMs are trained on a scrape of the public Internet - anyone can put up a website that can enter the training data.
  - An attacker can perform a **data poisoning** attack.

Content credits: https://stanford-cs324.github.io/winter2022/

While performance is an important factor, the other factors are the risks, right. Already LAMA, GPT's, right, all these models have been accused for different violations of policies, right.

They are not reliable in many cases. They produce disinformation. They are biased towards certain parts of our society, right. They produce toxic content, malicious content and so on and so forth, right. We will talk about the ethical aspects of it as well and of course the security is one of the aspects. If you do not know the architecture of the model, you would not be able to understand the vulnerability of these models.

Therefore the security aspects, the security loopholes are also interesting to understand, right.

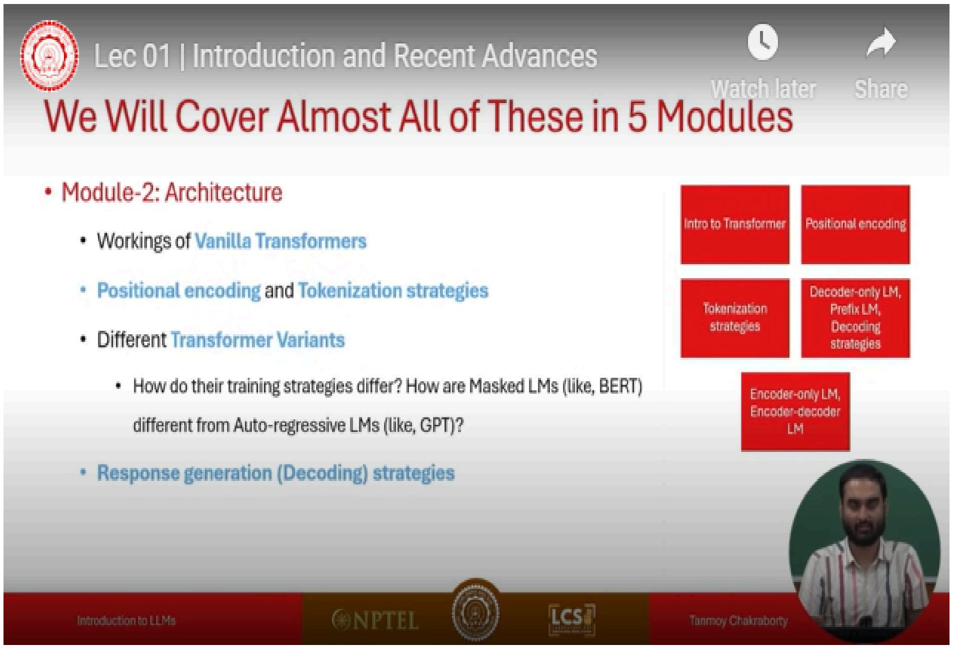


We Will Cover Almost All of These in 5 Modules

Module-1: Basics

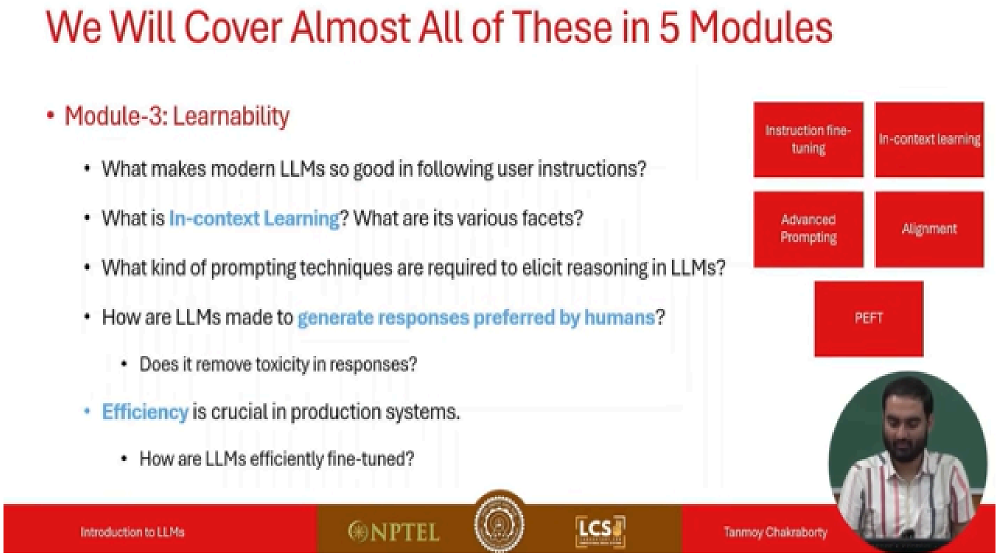
- A **refresher on the basics of NLP** required to understand and appreciate LLMs.
- A brief **introduction to the basics of Deep Learning**.
- The basics of **Statistical Language Modelling**.
- How did we end up in **Neural NLP**?
  - We will discuss the transition and the foundations of Neural NLP.
- Initial **Neural LMs**

Intro to NLP
Intro to Deep Learning
Intro to Language Models (LMs)
Word Embeddings (Word2Vec, GloVE)
Neural LMs (CNN, RNN, Seq2Seq, Attention)
Introduction to LLMs
NPTEL
LCS
Tanmoy Chakraborty



We Will Cover Almost All of These in 5 Modules

- Module-2: Architecture
  - Workings of **Vanilla Transformers**
  - **Positional encoding** and **Tokenization strategies**
  - Different **Transformer Variants**
    - How do their training strategies differ? How are Masked LMs (like, BERT) different from Auto-regressive LMs (like, GPT)?
  - **Response generation (Decoding) strategies**

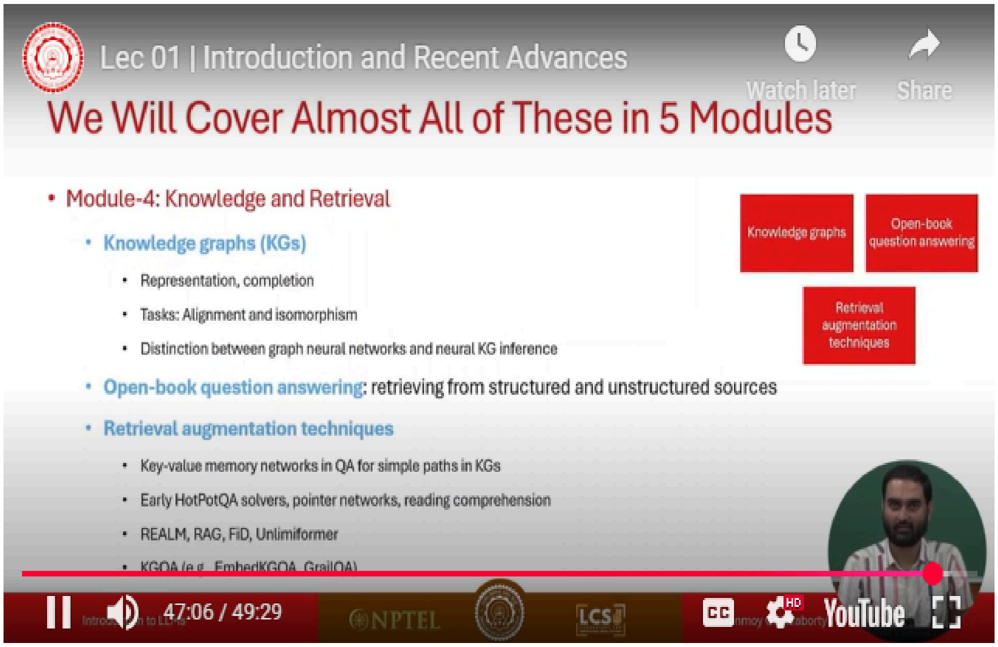


We Will Cover Almost All of These in 5 Modules

- Module-3: Learnability
  - What makes modern LLMs so good in following user instructions?
  - What is In-context Learning? What are its various facets?
  - What kind of prompting techniques are required to elicit reasoning in LLMs?
  - How are LLMs made to generate responses preferred by humans?
    - Does it remove toxicity in responses?
  - Efficiency is crucial in production systems.
    - How are LLMs efficiently fine-tuned?

Introduction to LLMs

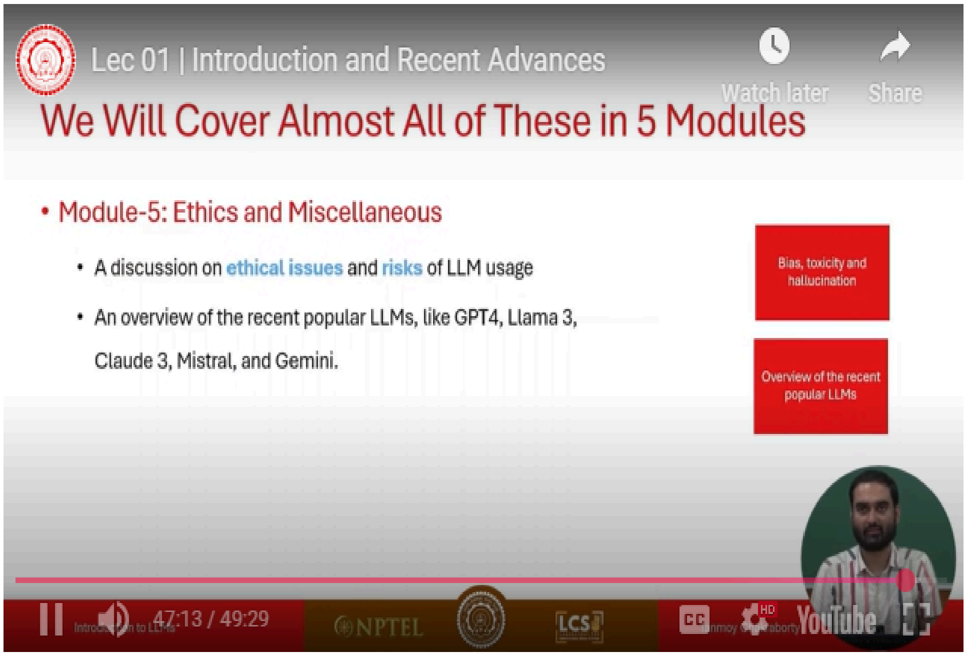
Tanmoy Chakraborty



Lec 01 | Introduction and Recent Advances

We Will Cover Almost All of These in 5 Modules

- Module-4: Knowledge and Retrieval
  - Knowledge graphs (KGs)
    - Representation, completion
    - Tasks: Alignment and isomorphism
    - Distinction between graph neural networks and neural KG inference
  - Open-book question answering: retrieving from structured and unstructured sources
  - Retrieval augmentation techniques
    - Key-value memory networks in QA for simple paths in KGs
    - Early HotPotQA solvers, pointer networks, reading comprehension
    - REALM, RAG, FiD, Unlimiformer
    - KGQA (e.g., EmbedKGQA, GrailQA)



We Will Cover Almost All of These in 5 Modules

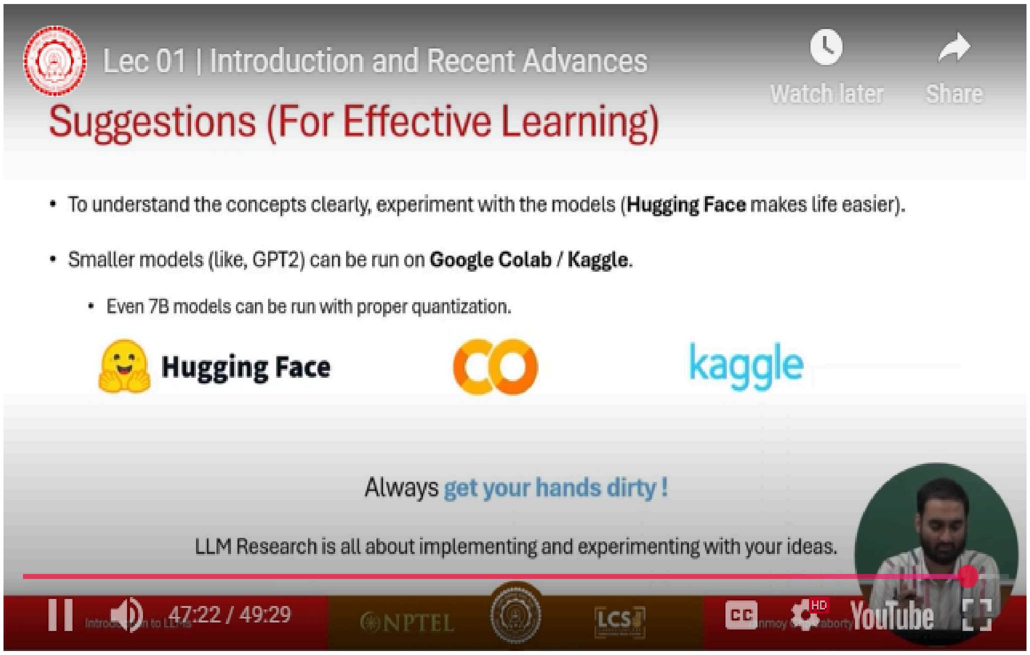
- Module-5: Ethics and Miscellaneous
  - A discussion on ethical issues and risks of LLM usage
  - An overview of the recent popular LLMs, like GPT4, Llama 3, Claude 3, Mistral, and Gemini.

Bias, toxicity and hallucination

Overview of the recent popular LLMs

So we will start with the basics. Then we move to architecture, I already mentioned. We move to learnability, right? Then move to advanced concepts, knowledge graph, integrating knowledge graph to the model rag and so on.

Then we look at different ethical concepts and other miscellaneous topics, advanced topics in large language models. okay.



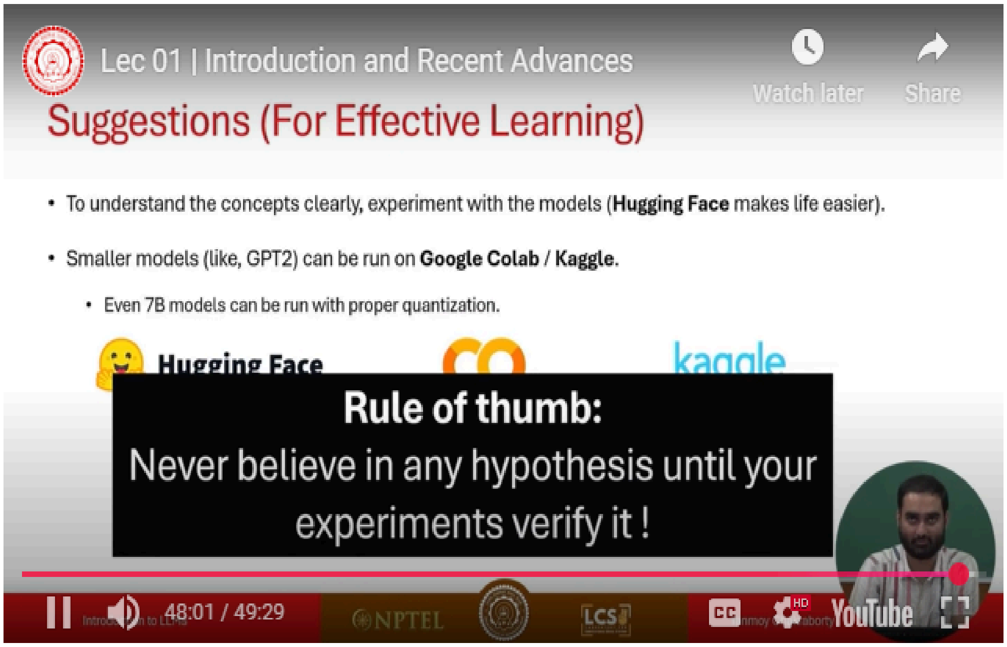
Suggestions (For Effective Learning)

- To understand the concepts clearly, experiment with the models (**Hugging Face** makes life easier).
- Smaller models (like, GPT2) can be run on **Google Colab / Kaggle**.
  - Even 7B models can be run with proper quantization.

Always get your hands dirty !

LLM Research is all about implementing and experimenting with your ideas.

If you are interested and if you are excited about this course, I strongly suggest you to make your hands dirty with Hugging Face libraries, with Kaggle. So Kaggle essentially releases many datasets, many researchers launch their competitions on Kaggle. So Many open source data sets are available in Kaggle and of course Google Colab should be one of the tools, one of the platforms, frameworks which you should know.



Lec 01 | Introduction and Recent Advances

Suggestions (For Effective Learning)

- To understand the concepts clearly, experiment with the models (**Hugging Face** makes life easier).
- Smaller models (like, GPT2) can be run on **Google Colab / Kaggle**.
  - Even 7B models can be run with proper quantization.

Rule of thumb:
Never believe in any hypothesis until your experiments verify it !

So I end today's lecture by saying this thumb rule. Never believe in any hypothesis until you experience it. Okay, so the entire space of large language model has been very shaky in the sense like, you know, People are coming up with different hypothesis, different conclusion, most cases codes are not available, most cases things are not reproducible, most cases there are multiple gurus who will teach you this to be done, this not to be done and so on and so forth. But you should not believe in all these things I mean you can of course you can capture those ideas right you can quote them right but you should not believe in this all these lectures right even you should not believe in my lectures until and unless you experience it you experimentally verify whether all this hypothesis, all these conclusions are valid conclusions or not, okay. With this I stop today's lecture. So, the next lecture we will talk about you know we will talk briefly about NLP. Thank you.