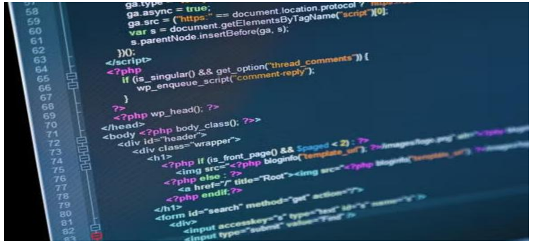
Text Generation - From Traditional LMs to LLMs

Dr. Pruthwik Mishra, DoAI, SVNIT, Surat
Dr. Pruthwik Mishra, Department of AI, SVNIT Surat
Session in STTP on Recent Trends and Practices in AI
Organized by DoAI, SVNIT Surat
10.07.2025

Text Generation and its Evolution

- Long Journey From Statistical Language Models (LMs) to Large Language Models (LLMs)
- Text Generation deals with the method of generating text by machines
    - Not natural text generation as humans
        - Synthetic Generation
    - Multiple Sources of Text

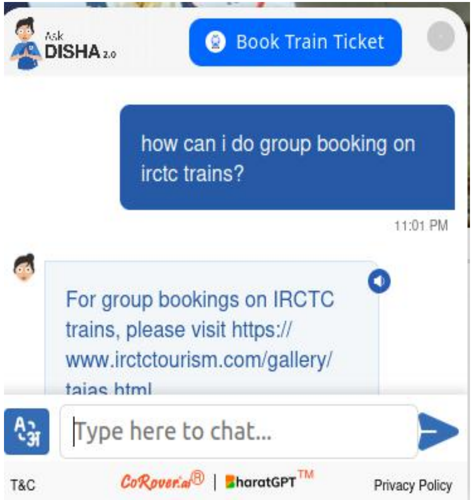


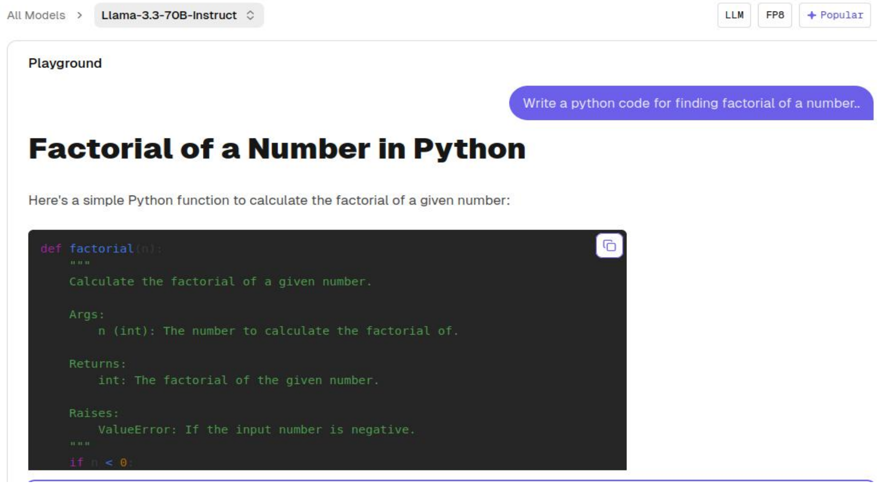
Key Milestones in the Evolution of Text Generation

- 1950s-1960s: Early Rule-Based Systems / Symbolic NLP (e.g., ELIZA)
    - Predefined rules, pattern matching, simple templates
    - Lacked understanding, couldn't generate truly novel or creative text
- 1980s-Early 2000s: Statistical Language Models (e.g., N-grams, HMMs)
    - Probabilistic models based on word frequencies and sequences from large corpora
    - Fluent and grammatically correct sentences than the rule based systems, but struggled with long-range dependencies and true semantic understanding
- Mid-2000s-Early 2010s: Neural Language Models (e.g., Feedforward Neural Networks, Word Embeddings like Word2Vec/GloVe)
    - Can "remember" past information in a sequence
    - Vanishing/exploding gradients, computational bottleneck due to sequential processing
- Late 2010s (Era of Transformers)
    - Demonstration of pre-training on huge datasets in an unsupervised manner
    - Capabilities of impressive zero-shot and few-shot learning capabilities
- Early 2020s - Present: Large Language Models (LLMs) / GPT-3, PaLM, Llama, Gemini, Claude, GPT-4/4o
    - Extreme scaling of parameters, training data, and computational power
    - Often combine pre-training with Reinforcement Learning from Human Feedback (RLHF)
    - Human-like performance in many language tasks as well as emergent abilities
        - coherent, contextually relevant, and diverse text across a vast range of tasks
            - creative writing, coding, summarization, conversation
    - Best model for text generation

What is Text Generation?

- The automatic creation of human-like text by a computer.
- Why is it important?
  - Several Applications
    - chatbots, content creation, summarization, translation, creative writing, coding assistance
  - Humans have always wanted machines to “think” and “write”.





Example

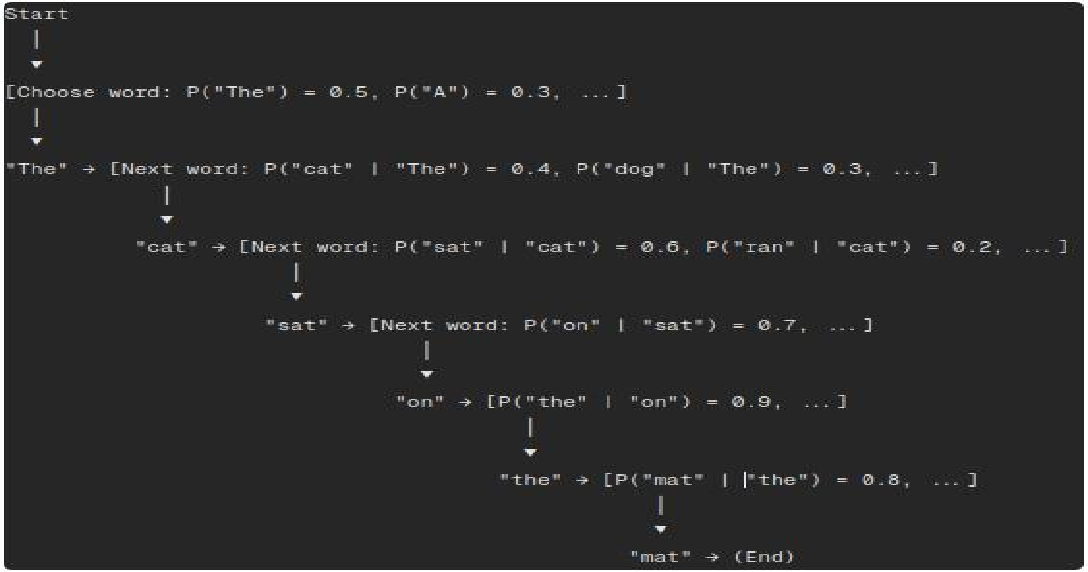
- There are 4 sentences, choose the best one.
  - he briefed to reporters on the chief contents of the statement
  - he briefed reporters on the chief contents of the statement
  - he briefed to reporters on the main contents of the statement
  - he briefed reporters on the main contents of the statement

Which is the best one and why?

- **he briefed reporters on the main contents of the statement**
  - This is the best sentence
    - *brief* is a transitive verb and requires an object
    - *main contents* is more **fluent** than *chief contents*
- “You shall know a word by the company it keeps”
  - Famous quote attributed to British linguist *J.R. Firth*
- This is the basis of language models

The Foundation: Language Models (LMs)

- What is a Language Model?
  - A model that assigns a probability to a sequence of words.
  - Using chain rule of Probability,
    - $P(w_1,w_2,...,w_n) = P(w_1) * P(w_2/w_1) * P(w_3/w_1,w_2) * ... * P(w_n/w_1,...,w_{n-1})$
  - Core idea for generation: Predict the next word given the previous words.



Early Approaches: N-gram Language Models

- The probability of the next word in a sequence of n words depends only on the previous “n-1” words
- Markov Assumption
  - First Order: the probability of the next word depends only on the previous word.
    - $P(W_n|W_{n-1})$
  - Second Order: the probability of the next word depends only on the previous word.
    - $P(W_n|W_{n-2},W_{n-1})$
- Using the Markov assumption, the probability of a sequence of n words can be further simplified
  - $P(w_1,w_2,...,w_n) = P(w_1)*P(w_2/w_1)*P(w_3/w_1,w_2)*...*P(w_n/w_1,...,w_{n-1})$ – *(No Markov Assumption)*
  - $P(w_1,w_2,...,w_n) = P(w_1)*P(w_2/w_1)*P(w_3/w_2)*...*P(w_n/w_{n-1})$ –*(With First Order Markov Assumption)*
  - $P(w_1,w_2,...,w_n) = P(w_1)*P(w_2/w_1)*P(w_3/w_1,w_2)*P(w_4/w_2,w_3)...*P(w_n/w_{n-2},w_{n-1})$ –*(With Second Order Markov Assumption)*

Types of Statistical LMs

- Unigram Model
  - Composed of unigrams or single tokens
  - $P(w_1,w_2,...,w_n) = P(w_1) * P(w_2) * P(w_3) * ... * P(w_n)$
  - All the words are independent of each other
    - Zeroth Order Markov Assumption
    - Very strong and weak assumption
- Bigram Model
  - Composed of bigrams or strings consisting of two tokens each
  - First Order Markov Assumption
  - $P(w_1,w_2,...,w_n) = P(w_1) * P(w_2/w_1) * P(w_3/w_2) * ... * P(w_n/w_{n-1})$
- Trigram Model
  - Composed of trigrams or strings consisting of three tokens each
  - Second Order Markov Assumption
  - $P(w_1,w_2,...,w_n) = P(w_1) * P(w_2/w_1) * P(w_3/w_1,w_2) * P(w_4/w_2,w_3) ... * P(w_n/w_{n-2},w_{n-1})$

How to Compute the Probabilities?

- Maximum Likelihood Estimation (MLE)
- Use relative frequency
  - $P(w_i|w_{i-1})=count(w_{i-1},w_i)/count(w_{i-1})$
  - This count is computed on a large corpus of data
    - For reliable estimation of probabilities
- Computationally Inexpensive and Easy
- Suffers from Sparsity Problem
  - Zero probabilities for unseen sequences
  - Smoothing Used to handle this
- Limited context window
  - Can't capture long-range dependencies
- Lack of generalization.

	i	want	to	eat
i	6	828	1	10
want	3	1	609	2
to	3	1	5	687
eat	1	1	3	1

Smoothed bigram counts from BeRP Corpus[1]

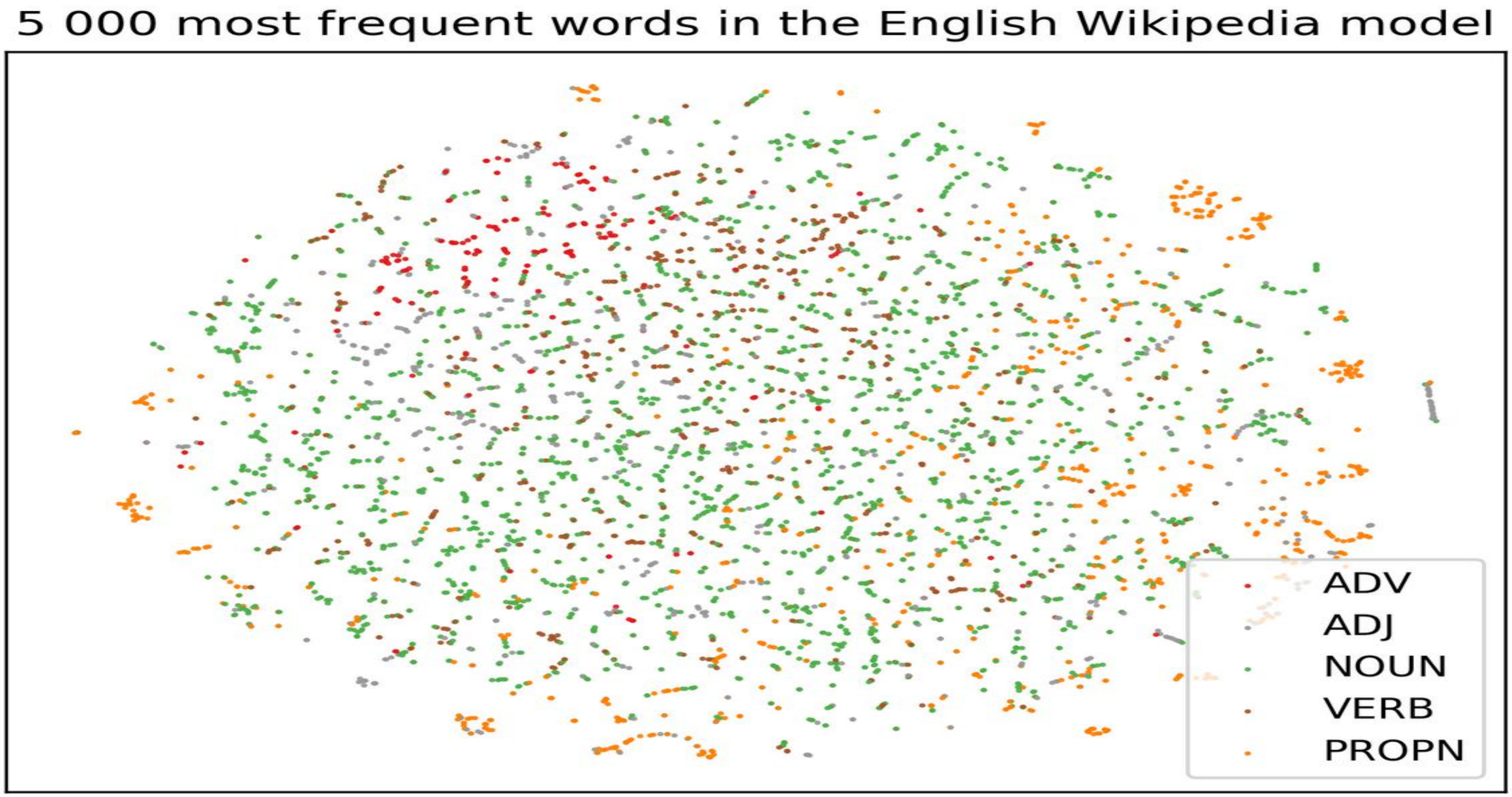
	i	want	to	eat
i	0.0015	0.21	0.00025	0.0025
want	0.0013	0.00042	0.26	0.00084
to	0.00078	0.00026	0.0013	0.18
eat	0.00046	0.00046	0.0014	0.00046

Smoothed bigram probabilities from BeRP Corpus[1]

Beyond N-grams: Introducing Neural Language Models

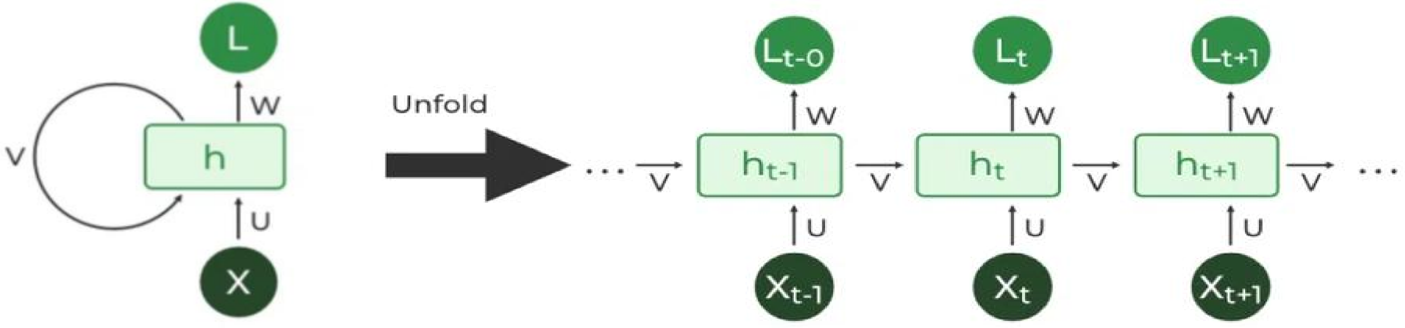
- Motivation for neural LMs
  - Overcome sparsity and capture longer dependencies
- Word Embeddings: A crucial step
  - Representing words as dense vectors in a continuous space
  - Capturing semantic relationships [2]
    - *India - Sachin + Australia = Pointing* [Word2vec model trained on GoogleNews]
    - *France: Paris:: India: New Delhi*
    - 3 most Similar words to *Computer*
      - *Microcomputer*
      - *Laptop*
      - *Mainframe*
  - Simple feed-forward Neural Networks were used for early word embedding

Word Vectors Visualization [2]



Recurrent Neural Networks (RNNs) for Text Generation

- How RNNs work?
  - Process sequences by maintaining a hidden state that carries information from previous steps

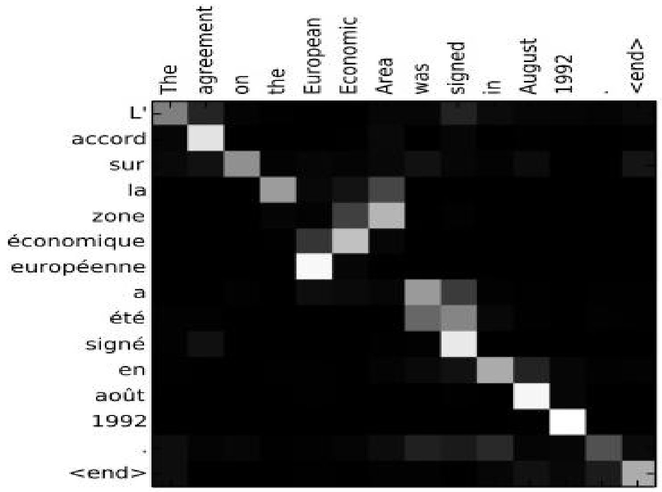


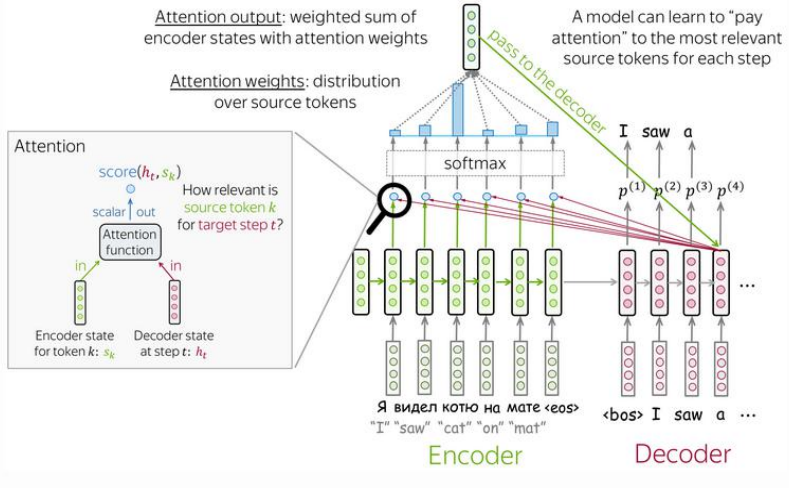
RNN Unfolding

- Vanishing/Exploding gradients
  - Difficulty learning very long-range dependencies
  - To alleviate this problems, LSTM/GRU are proposed using gating mechanism
- Slow training
- Lack of Parallelism

The Breakthrough: The Attention Mechanism

- Allows the model to "*focus*" on relevant parts of the input sequence when generating output
- Encoder-Decoder Architecture with Attention
- Resulted in improved performance in machine translation, summarization, and other sequence-to-sequence tasks

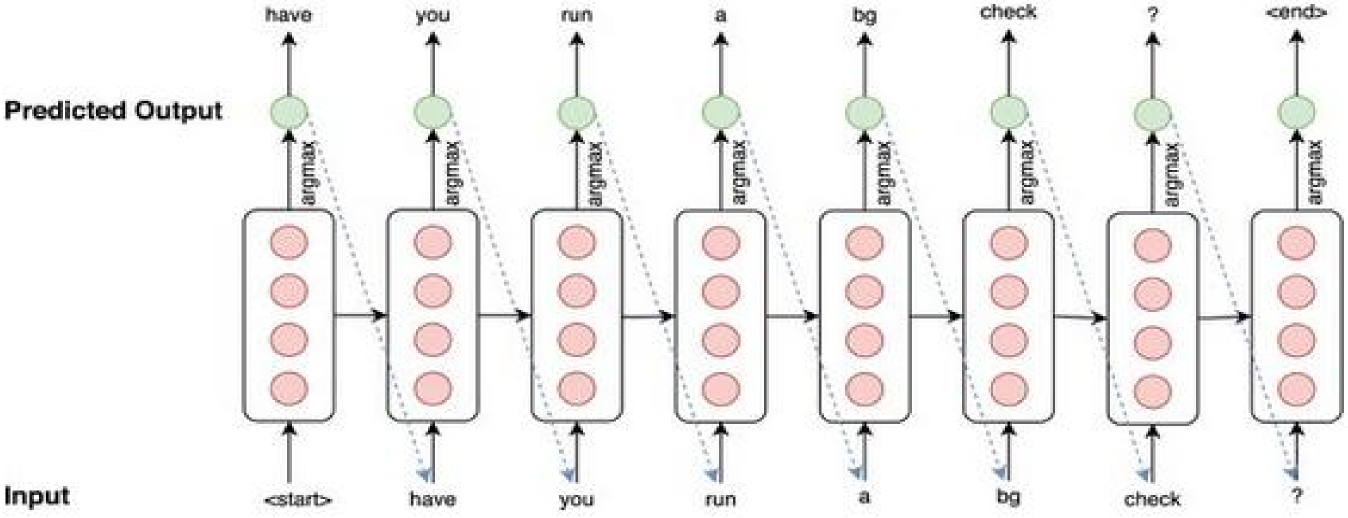




[4]

Just a Next Word Prediction Task

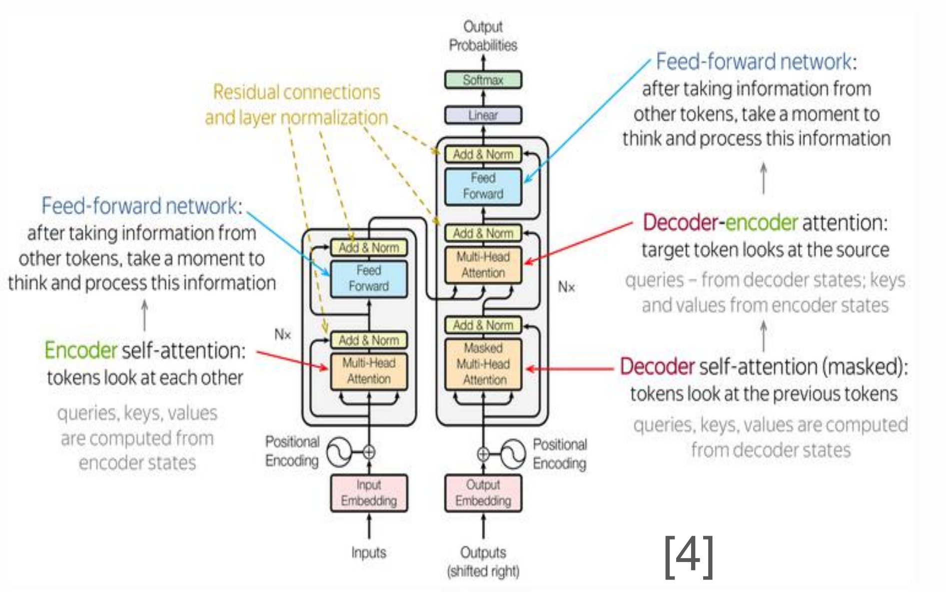
- Text Generation is a next word prediction task in a recursive manner
  - When do you stop generating?
    - After a fixed number of words
    - After the end sentence token is generated



[9]

The Transformer Architecture: The New Paradigm

- Key Innovation: Self-Attention mechanism
  - allowing each word to attend to all other words in the same sequence
- Elimination of recurrence
  - Introduction of Parallelization
- Encoder and Decoder stacks
- Positional Encoding
  - How to retain word order information
- Introduction of Key, Query, Value

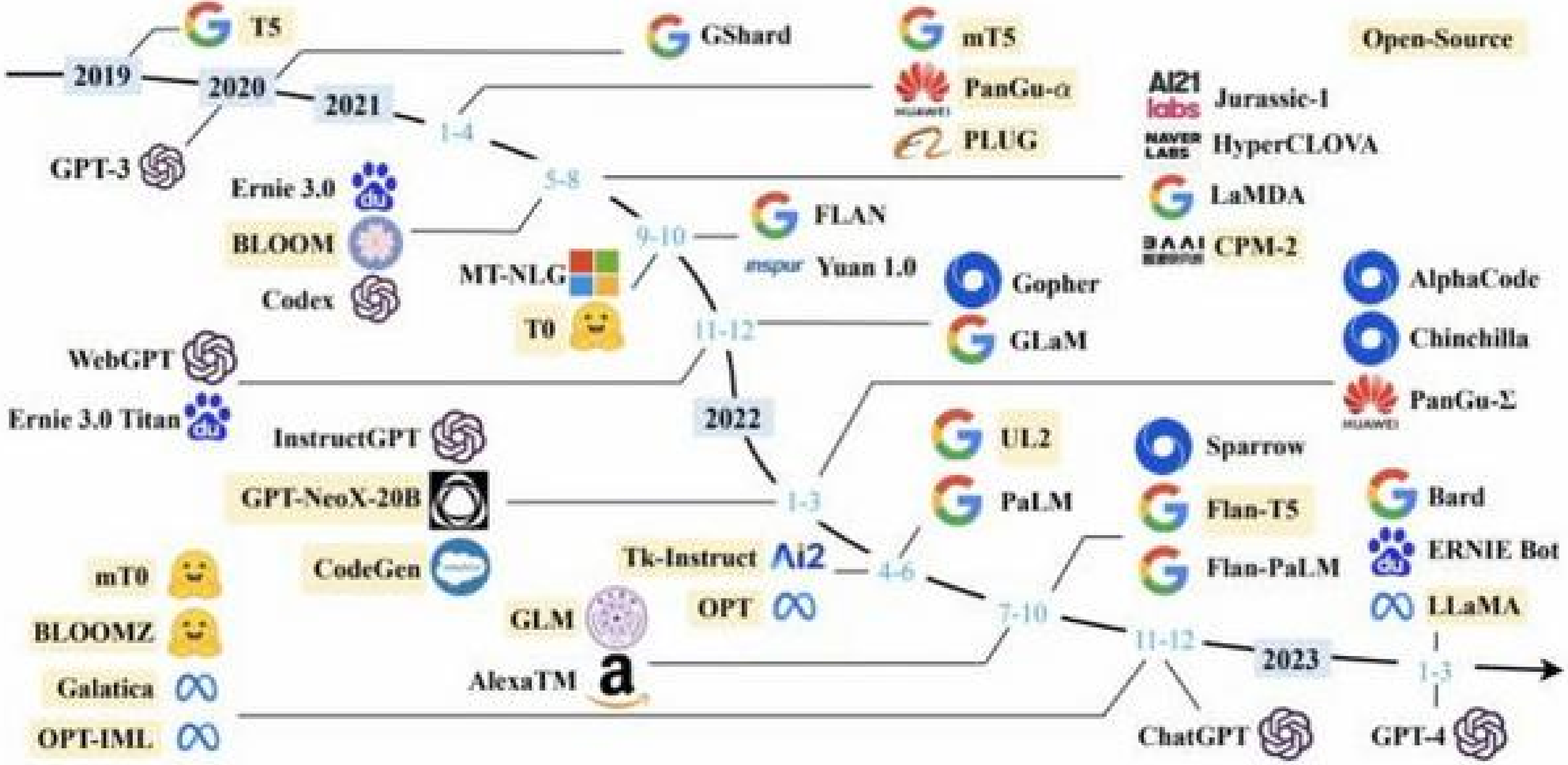


[4]

The Era of Large Language Models (LLMs)

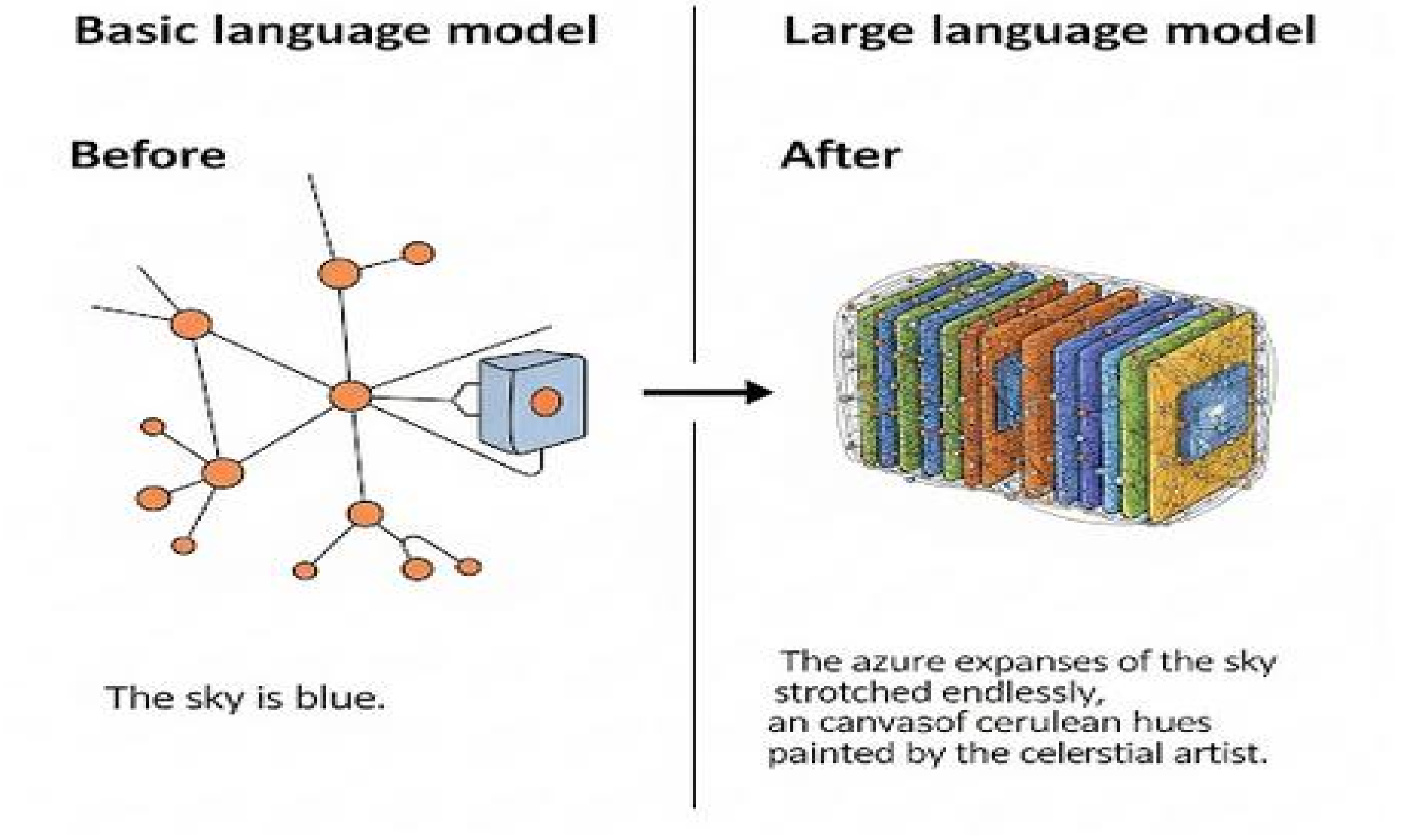
- Scaling Massively Up
  - Massive datasets (Common Crawl, web text, books)
  - Billions and Trillions of parameters
  - Enormous computational resources
- Pre-training and Fine-tuning Paradigm
  - Pre-training: Unsupervised learning on vast text data (next-word prediction, masked language modeling).
  - Fine-tuning: Supervised learning for specific downstream tasks (e.g., question answering, sentiment analysis).
  - Emergent Abilities: Capabilities that arise from scale (e.g., reasoning, code generation, summarization).

Timeline of LLMs [5]



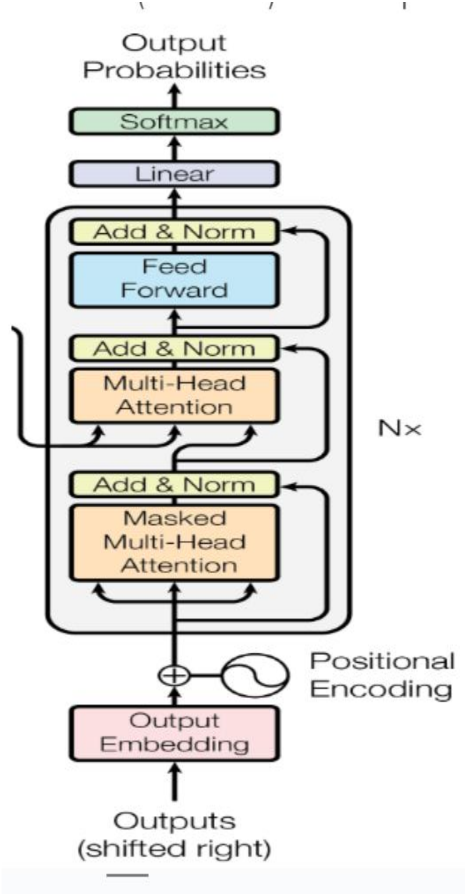
Very Very Competitive Field

Text Generation: Before and After [6]



How LLMs Generate Text: Decoding Strategies

- LLMs are Decoder-Only Models
- Autoregressive generation: One token at a time
- Decoding Strategies:
  - Greedy Search: Always pick the most probable next word
  - Beam Search: Keep track of multiple high-probability sequences
  - Sampling: Introduce randomness
    - Top-k
    - Nucleus sampling (p-sampling))
  - Temperature parameter for controlling diversity and creativity



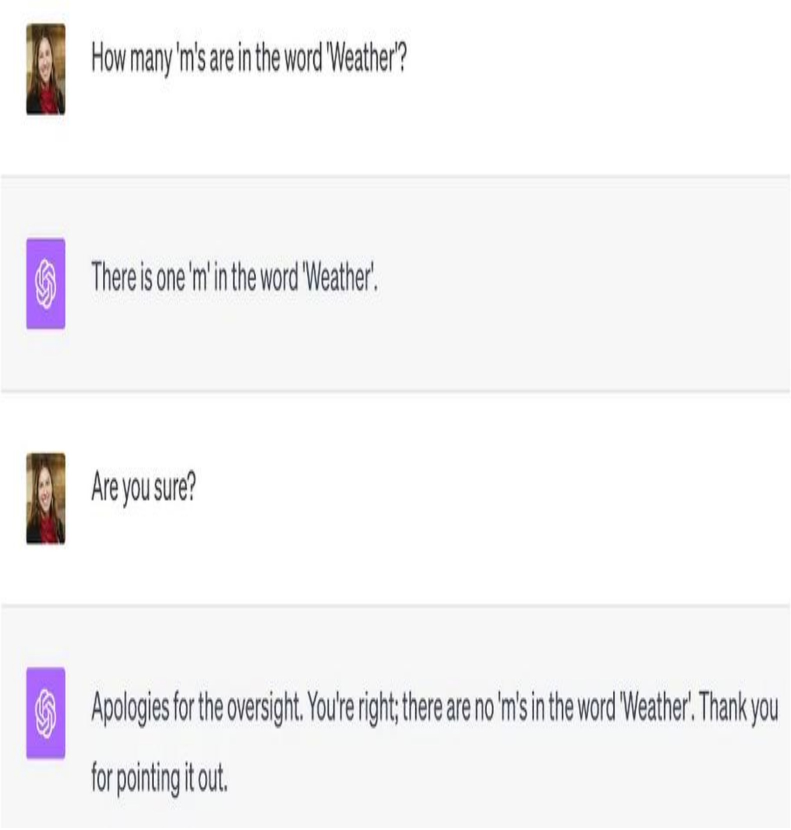
Key Characteristics and Capabilities of Modern LLMs

- Generalization
  - Perform well on a wide range of tasks
- Contextual Understanding
  - Long context windows (far better than Markov models)
- Few-shot/Zero-shot Learning
  - Ability to perform tasks with minimal or no examples
- Instruction Following
  - Responding to natural language prompts
- Reasoning and Problem Solving
  - Chain-of-thought prompting, Program-of-thought prompting
- Multimodality
  - Some LLMs are moving towards understanding images/audio

Challenges and Ethical Considerations of LLMs

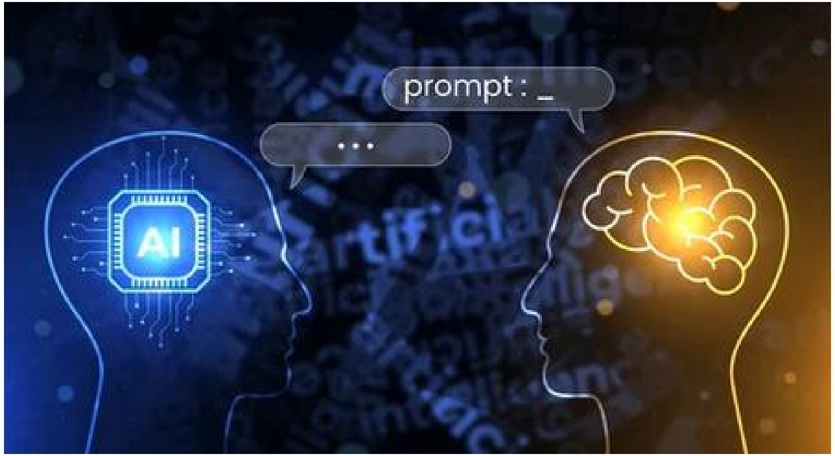
- Bias
  - Reflecting biases in training data
- Hallucination
  - Generating factually incorrect or nonsensical information
- Toxicity/Harmful Content Generation
- Computational Cost
  - Training and inference are very expensive
- Environmental Impact
- Data Privacy and Security
- The need for responsible AI development

Hallucination Example [7]



The Future of Text Generation

- Continued scaling of models and data
- Improved control with better explainability
- Addressing ethical concerns through research and regulation
- Personalized generation
- Multimodal LLMs becoming more prevalent
- Integration into more applications and workflows
- The role of human-in-the-loop



Conclusion and Key Takeaways

- Text generation has evolved from simple statistical methods to highly complex neural networks
- The Transformer architecture and the concept of scaling have revolutionized the field
- LLMs offer unprecedented capabilities but come with significant challenges.
- Responsible development and deployment are crucial for the future
- You can explore LLMs at https://app.hyperbolic.xyz/models/

Creative Future Lies Ahead…[8]

References

1. Speech and Language Processing, 2nd Edition in PDF format (complete and parts) by Daniel Jurafsky, James H. Martin
2. https://vectors.nlpl.eu/explore/embeddings/en/
3. https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/
4. https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html
5. https://www.nextbigfuture.com/2023/04/timeline-of-open-and-proprietary-large-language-models.html
6. https://gemini.google.com/
7. https://www.iguazio.com/glossary/llm-hallucination/
8. https://medium.com/@tehreemyounas/ai-as-a-catalyst-for-creativity-igniting-innovation-within-the-modern-world-36959bef2199
9. https://www.researchgate.net/publication/342609939_Developing_a_Twitter_bot_that_can_join_a_discussion_using_state-of-the-art_architectures