11868
Large Language Model Systems

Lei Li

Carnegie Mellon University
Language Technologies Institute

Large Language Model Products

Google

Bard

PaLM

Gemini

OpenAI

ChatGPT

GPT-4

ANTHROP\C

Meta

LLaMA

Outline

- Capabilities of LLMs
- Mathematical Foundations
- Challenges in LLM Systems
- Logistics

LE create a viral tweet about my upcoming talk at Carnegie Mellon University (CMU, in Pittsburgh) with the title "Empowering Responsible Use of Large Language Models". Topics include watermark, detecting AI generated text, protecting copyright of models. Make it funny and witty.

🎙️ Hey yinzers at @CarnegieMellon! 🖐️ Get ready to turn "yinzpiration" into action with my talk on "Empowering Responsible Use of Large Language Models"! 🤖💡
We're gonna tackle AI like it's the Steelers vs. the Ravens 🏈, with tricks like watermarking (no, not on Primanti's sandwiches 🥪), AI text detective skills sharper than a Pittsburgh pierogi 🕵️, and copyright protection tighter than your grandma's knitting stitches! 🧶💼
Don't be a jagoff - join me in Pittsburgh for a talk that's more fun than a Kennywood coaster 🎢 and juicier than a Primanti's sandwich! 🤣
#CMUTalk #LanguageModelLingo #AIinSteelCity 🤖📚🌆

LLM can

- Translate
- Summarize
- Proof-read and correct
- Explain words
- Create article/email
- Make restaurant suggestions
- Chat with users
- Do Math Calculations
- Answer questions on many subjects
- Suggest names
- Write code

Translation

Google MT

他上班天天摸鱼

Tā shàngbān tiāntiān mō yú

7 / 5,000

He fishes every day at work

ChatGPT/GPT4

LE **You**

Please translate "他上班天天摸鱼"

ChatGPT

The phrase "他上班天天摸鱼" translates to "He slacks off at work every day" in English. Here, "摸鱼" (mō yú) is a Chinese slang term that literally translates to "touching fish," but it's commonly used to mean slacking off or being unproductive, especially at work.

ChatBot to Mimic a User

- https://github.com/Ads-cmu/WhatsApp-Llama
- Train on your WhatsApp chat data (about 10 chat histories are adequate), about a couple hours
- Generate responses to mimic user style

Rewrite Text with Style

You

Please write the following sentence in Yoda's style. Here are a few examples.
You have much to learn. -> Much to learn, you still have.
I would say you have found someone, hmmm? -> Found someone, you have, I would say, hmmm?

You should not drive that fast. ->

ChatGPT

Drive that fast, you should not.

Commonsense Reasoning

LE **You**

Fill in the blank: Steven was disgusted that Robert chose to cook pork for dinner, because _ was a vegetarian.
"option1": "Steven", "option2": "Robert"

ChatGPT

The correct answer is "Steven"

Math Reasoning

You

Roger has 4 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

ChatGPT

Roger initially has 4 tennis balls. When he buys 2 more cans of tennis balls, and each can has 3 tennis balls, he adds 2 cans * 3 tennis balls per can = 6 tennis balls to his collection.
So, Roger now has a total of 4 initial tennis balls + 6 newly purchased tennis balls = 10 tennis balls.

NBC NEWS

ChatGPT passes MBA exam given by a Wharton professor

The bot's performance on the test has "important implications for business school education," wrote Christian Terwiesch, a professor at the University of Pennsylvania's Wharton School.

Jan. 23, 2023, 4:46 PM EST / Updated Jan. 24, 2023, 1:19 PM EST

By Kalhan Rosenblatt

ChatGPT scored between a B- and B on the MBA exam.

LE **You**

The Circored process produces direct reduced iron in Latin America. The production process starts with a pile of iron ore fines. The ore is moved through seven process steps before it emerges as finished direct-reduced-iron briquettes. The steps and their capacities are as follows:

- Pre-Heater: 60 tons per hour
- Lock Hoppers: 55 tons per hour
- First reactor: this reactor holds up to 14 tons of iron and iron needs to spend 15 minutes in the reactor
- Second reactor: this reactor holds up to 200 tons of iron and iron needs to spend 4 hours in the reactor
- Flash heater: 62.5 tons per hour
- Discharge: 59 tons per hour
- Briquetting: three machines of 27.5 tons per hour each

Where in the process is the bottleneck and what is the overall process capacity? Justify your answer.

Christian Terwiesch. Would Chat GPT3 Get a Wharton MBA? 2023.

ChatGPT

To identify the bottleneck in the Circored process and the overall process capacity, we need to look at the capacities of each step. The bottleneck in any production process is the step with the lowest throughput rate, as it limits the rate at which the entire system can operate.

Here are the capacities given:

1. **Pre-Heater**: 60 tons per hour
2. **Lock Hoppers**: 55 tons per hour
3. **First Reactor**: Holds 14 tons total; each batch takes 15 minutes, so its rate is 14 tons0.25 hours=56 tons per hour0.25 hours14 tons=56 tons per hour
4. **Second Reactor**: Holds 200 tons total; each batch takes 4 hours, so its rate is 200 tons4 hours=50 tons per hour4 hours200 tons=50 tons per hour
5. **Flash Heater**: 62.5 tons per hour
6. **Discharge**: 59 tons per hour
7. **Briquetting**: Three machines at 27.5 tons per hour each, so 3×27.5 tons per hour=82.5 tons per hour3×27.5 tons per hour=82.5 tons per hour

Now, let's calculate and compare the capacities of each step to identify the bottleneck.

The bottleneck in the Circored process is the **Second Reactor**, with a capacity of 50 tons per hour. …

ChatGPT even generates code to verify

```python
# Capacities of each step in tons per hour
pre_heater = 60
lock_hoppers = 55
first_reactor = 14 / 0.25
second_reactor = 200 / 4
flash_heater = 62.5
discharge = 59
briquetting = 3 * 27.5
# Identifying the bottleneck (the smallest capacity) and overall process capacity
capacities = {
  "Pre-Heater": pre_heater,
  "Lock Hoppers": lock_hoppers,
  "First Reactor": first_reactor,
  "Second Reactor": second_reactor,
  "Flash Heater": flash_heater,
  "Discharge": discharge,
  "Briquetting": briquetting }
bottleneck_step = min(capacities, key=capacities.get)
bottleneck_capacity = capacities[bottleneck_step]

(bottleneck_step, bottleneck_capacity)
```

It connects to a python engine, and executes the program to obtain result

Result

```
('Second Reactor', 50.0)
```

LE **You**

You are given a 0-indexed array mountain. Your task is to find all the peaks in the mountain array. please write a python program.

ChatGPT

```
def find_peaks(mountain):
    peaks = []
    for i in range(1, len(mountain) - 1):
        if mountain[i] > mountain[i - 1] and
mountain[i] > mountain[i + 1]:
            peaks.append(i)
    return peaks
```

please draw a llama climbing along the "walking into the sky" on the campus of CMU

please draw an image for my slides about ”The power of Large Language Models"

Quiz

- Please pick a simple daily task or a question, create a prompt to ask an LLM (ChatGPT/Bard/Microsoft Co-Pilot/Claude), examine the answer.

Outline

- Capabilities of LLMs
- Mathematical Foundations
- Challenges in LLM Systems
- Logistics

The Power of Predicting Next Token

$Prob.(next\_word|prefix)$

Santa Barbara has very nice _____

beach	0.5
weather	0.4
snow	0.01

Pittsburgh is a city of _____

bridges	0.6
corn	0.02

Mathematics of Language Model

Probability(“Pittsburgh is a city of bridges”)
$= P(\text{“}Pittsburgh\text{”}) \cdot P(\text{“}is\text{”} | \text{“}Pittsburgh\text{”})$
$\cdot P(\text{“}a\text{”} | \text{”}Pittsburgh\ is\text{”}) \cdot P(\text{“}city\text{”} | \ldots) \cdot P(\text{“}of\text{”} | \ldots)$
$\cdot P(\text{“}bridges\text{”} | \ldots)$

$$\text{Prob.}\,(x_{1..T}) = \prod_{t=1}^{T} \underline{\mathrm{P}(\mathrm{x}_{t+1} | \mathrm{x}_{1..t})}$$

Predicting using Neural Nets
(Transformer network)

Why is ChatGPT changing AI landscape

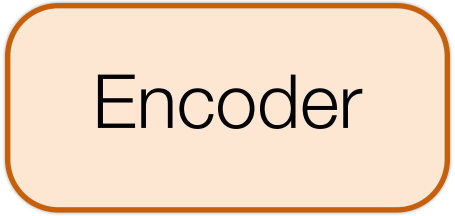
- Pre-training on very large raw data (300B tokens) + small human feedback
- Instruction following – easy to use through natural instruction
- In-context learning – Generalize well to versatile tasks, by showing a few examples at use time.

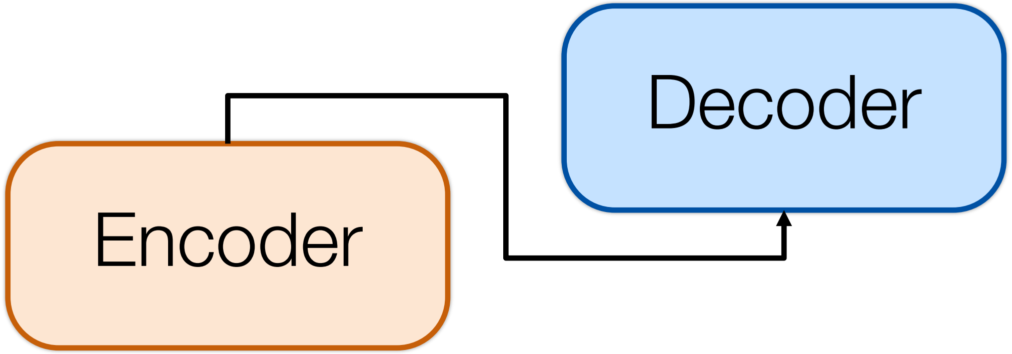
Type of Language Models

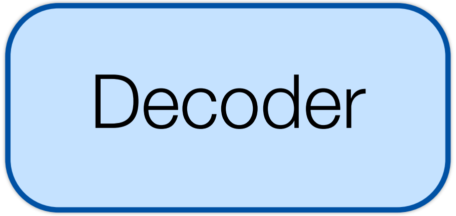
Encoder-only
Masked LM
Non-autoregressive

Encoder-decoder

Decoder-only
Autoregressive

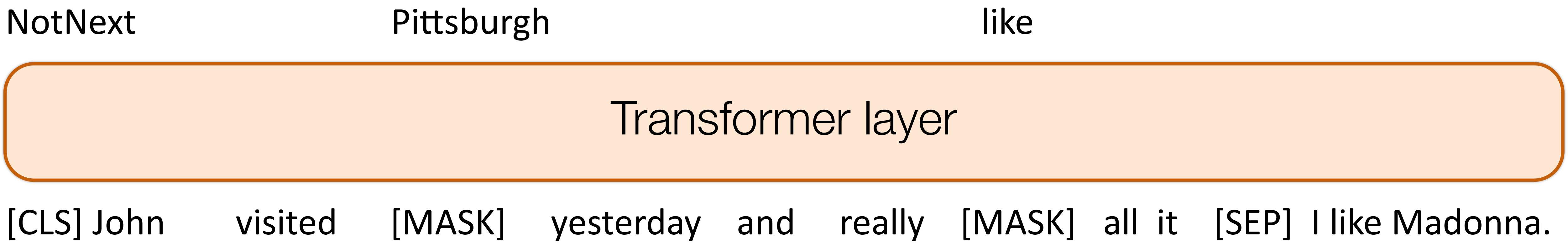






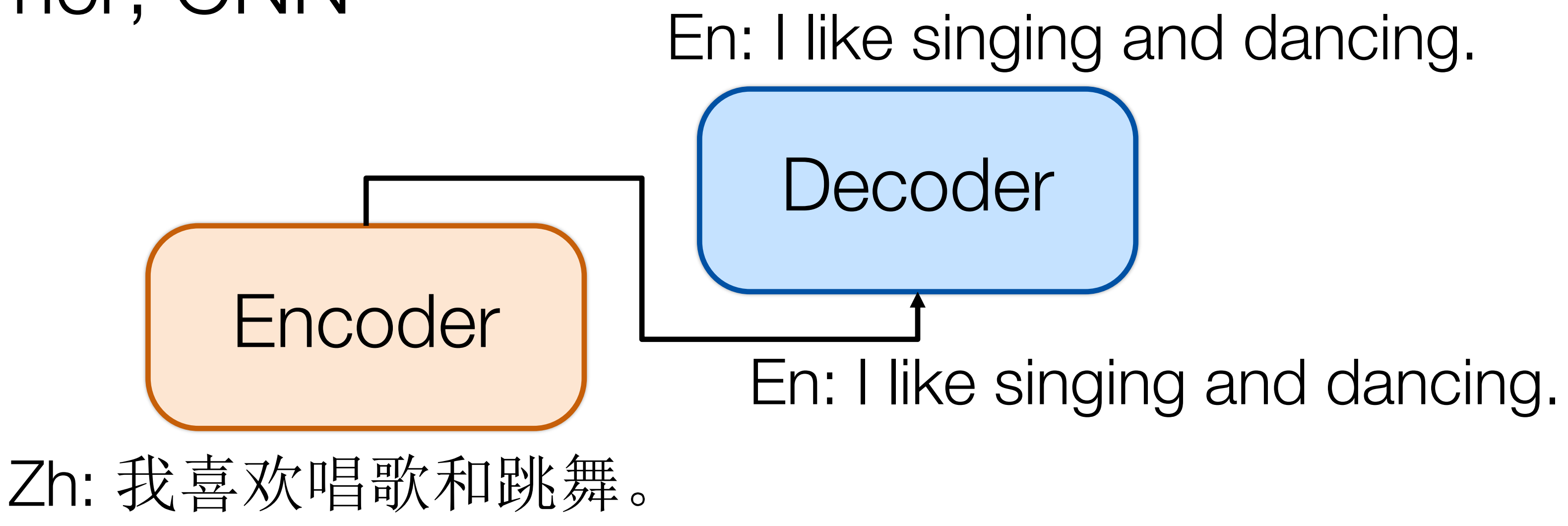
Encoder-only Language Model

- Masked Prediction: BERT $P(x_{mask}|x)$
- Can also be used to generate language
  - Need special network, e.g. REDER (reversible duplex model)



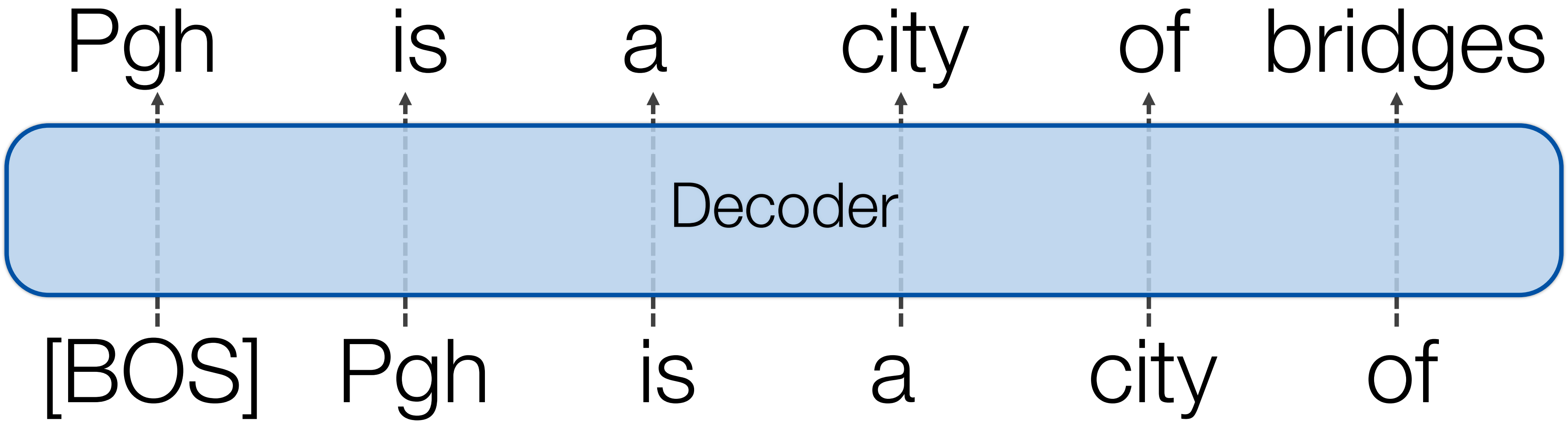
Encoder-Decoder Language Model

- Model the probability $P(Y|X)$
- Encoder may choose Bi-LSTM, Transformer, CNN
- Decoder may choose LSTM, Transformer or Non-auto Transformer, CNN



Decoder-only Language Model

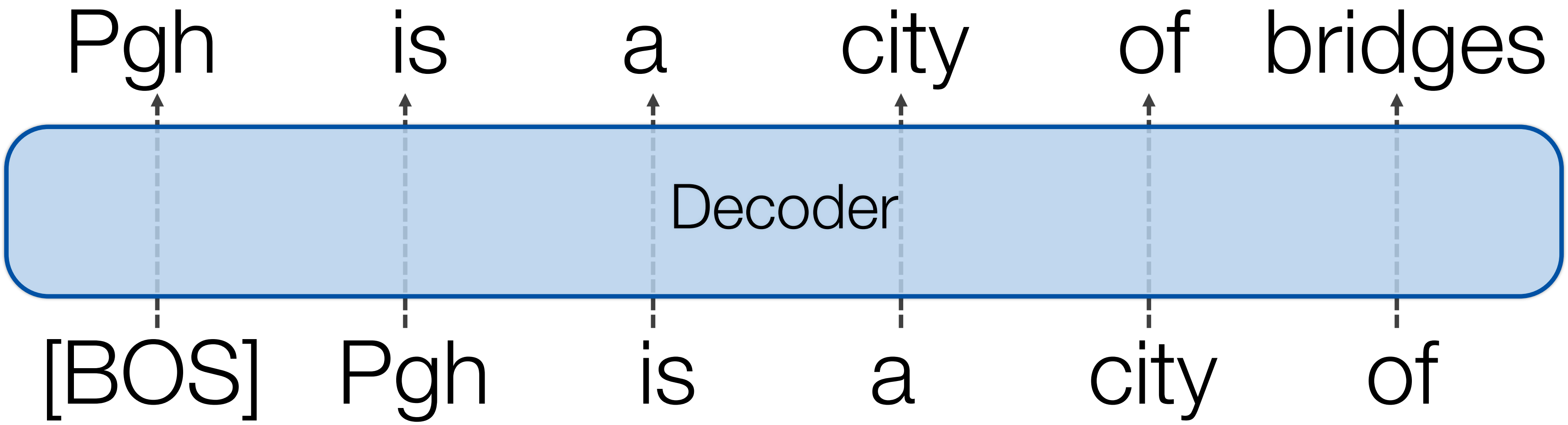
- Causal Language Model: $P(X) = \prod_{n=1}^{N} P(x_n | x_{1..n-1})$
- Model choice: Transformer or (uni-directional) LSTM



Training Objective

- Cross entropy loss

$$CE = -\frac{1}{N}\sum_{n=1}^{N} \log P_\theta(x_n|x_{<n})$$



Measuring Performance of Language Model

- Perplexity: for open-end generation

$$PPL = \exp -\frac{1}{N}\sum_{n=1}^{N} \log P_{LM}(x_n|x_{<n})$$

- Reference-based metric:
  - SEScore2/COMET/BLEURT: model-based score
  - BLEU (or ROUGE): based on n-gram matching, old but classic
  - InstructScore: explainable score [Xu et al 2023]

Measuring Performance of Language Model

- Using downstream task performance metrics
  - Named entity recognition: F1
  - Question Answering: accuracy, matching rate
  - Code generation: pass-rate @k
  - Retrieval: ndcg
  - Summarization: ROUGE
  - Translation: COMET/SEScore2/BLEU

Performance of LLM

- GPT3 trained on 300B tokens
- Both model scale and data are important

Perplexity

Setting	PTB
SOTA (Zero-Shot)	35.8[a]
GPT-3 Zero-Shot	**20.5**

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

Computation in Language Models

- Common network layer
  - Multi-head Attention
  - Layer norm
  - Dropout
  - Linear layer
  - Nonlinear activation: Tanh, RELU, GELU
  - Softmax
- Low level operators:
  - Matrix/Tensor multiplication
  - Reduction: summation, avg
  - Map: element-wise applying
  - Memory movement

Outline

- Capabilities of LLMs
- Mathematical Foundations
- Challenges in LLM Systems
- Logistics

Modern AI Research

- General
  - Instead of focus on one AI model, develop generic models that work for many tasks
- Instructibility
  - Interactive AI that is able to take natural or infinite commands
- Alignment
  - Not a single explicit objective, but also its behavior aligned with human/society/cultural norm

Big Ideas in Systems Research

- Problem Formulation
  - Using less resources (CPU/GPU/memory/networking bandwidth) to compute certain (or bigger) workload faster
- Abstraction
  - Designing useful building blocks that hide complexity from application developers (modularity, layering, optimization, etc)
- Tradeoffs
  - What are the fundamental constraints?
  - How can you reach new points in the trade-off space?

What is AI-Systems Research?

- Good AI and Systems research
  - Provides insights to both communities
  - Builds on big ideas in prior AI/ML and Systems Research
- Leverages understanding of both AI and Sys domains
  - Studies statistical and computational tradeoffs
  - Identify essential abstractions to bridge AI and Systems
  - Reframes systems problems as learning problems
- More than just useful open-source software!
  - But software impact often matters…

System Performance

- The elephant in the room given “scale is all you need”!
- Lots of levels to work at:
  - Model architecture
  - Algorithms
  - Software optimization (mostly instructions & data movement)
  - Hardware
  - Larger systems (e.g. a datacenter serving multiple models)

What Matters for System Performance?

Table 1. Speedups from performance engineering a program that multiplies two 4096-by-4096 matrices. Each version represents a successive refinement of the original Python code. "Running time" is the running time of the version. "GFLOPS" is the billions of 64-bit floating-point operations per second that the version executes. "Absolute speedup" is time relative to Python, and "relative speedup," which we show with an additional digit of precision, is time relative to the preceding line. "Fraction of peak" is GFLOPS relative to the computer's peak 835 GFLOPS. See Methods for more details.

Version	Implementation	Running time (s)	GFLOPS	Absolute speedup	Relative speedup	Fraction of peak (%)
1	Python	25,552.48	0.005	1	–	0.00
2	Java	2,372.68	0.058	11	10.8	0.01
3	C	542.67	0.253	47	4.4	0.03
4	Parallel loops	69.80	1.969	366	7.8	0.24
5	Parallel divide and conquer	3.80	36.180	6,727	18.4	4.33
6	plus vectorization	1.10	124.914	23,224	3.5	14.96
7	plus AVX intrinsics	0.41	337.812	62,806	2.7	40.45

Leiserson et al, There's Plenty of Room at the Top, Science, 2020

What Matters for System Performance?

- The most expensive thing physically is data movement, so that tends to matter most
  - Model is also data
  - Processor core to cache
  - Cache, to other cache levels, to memory
  - Memory to storage & network
  - This is why we worry about batching, block sizes, precision, etc!
- Of course, other things matter too:
  - Utilization: keep all parts of system working all the time
  - Amdahl's Law: diminishing returns from optimizing 1 component
  - Instruction count

Reliability

- Large, parallel systems are more likely to experience failures and stragglers
  - What if one step on one gpu with a batch fail to produce gradient?
- Lots of techniques to reduce rate and impact of failures
  - Generally easier to optimize mean time to repair (MTTR) than mean time between failures (MTBF)
- ML gives more flexibility because it’s approximate

Programming Models

- What are good abstractions to build AI applications?
  - Upper level: integrating models into a product system, tracking and improving product quality over time
  - Middle level: building model training and inference software, streaming dataflow
  - Lower level: building GPU kernels, compilers, etc
- A good abstraction successfully frees the programmer of one or more concerns (e.g., performance or failures) while supporting a wide range of apps on top

Security

- Also a “systems” concern about abstractions and their potential misuse
- With DL today, one can fairly easily:
  - Recover the training data that went into models
  - Find adversarial inputs that push models to a target output
  - Bypass current methods to train models to be safe
- LLMs are especially troublesome:
  - They sort of do “code execution” (instruction following)
  - People are trying to let them take actions (plugins, agents)

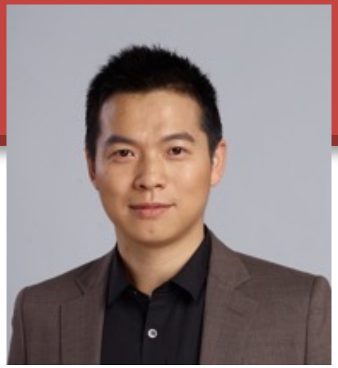
What makes a great (accepted) paper?

- Novel problem settings, benchmarks, and metrics
- A substantial claim
  - Surprising discovery
  - State of the art results: accuracy, quality, complexity
- Innovation in methods: network architecture, training methodology, augmentation, objectives…
- Theoretical results that provide a deeper understanding
- Narrative and framing in prior work and current trends

Outline

- Capabilities of LLMs
- Mathematical Foundations
- Challenges in LLM Systems
- Logistics

Prof. Lei Li



Carnegie Mellon University
Language Technologies Institute

Research Area:
Large language models
Machine Translation
AI drug discovery

2011	2014	2016	2021	2023
CMU PhD SCS, CSD	Berkeley UNIVERSITY OF CALIFORNIA PostDoc EECS	Baidu 百度 Principal Scientist Institute of Deep Learning	ByteDance Founding Director AI Lab	Assist. Prof. CS UC SANTA BARBARA Carnegie Mellon University

Highlight of Past Work

- led and developed ByteDance's machine translation system VolcTrans and AI writing system Xiaomingbot, which have been deployed in products (Toutiao, Douyin, Tiktok, Lark), serving over a billion users.

- Deep learning system research
  - LightSeq: An acceleration library for Transformer models

- Machine translation: LegoMT, VOLT(ACL21 best paper), mRASP, GLAT, WACO, InstructScore

- Large Language Models: SALAM, ToolDec, ALGO, Ginsew

- AI for Science: drug discovery, protein design

Learning Objective

- Learn basic knowledge and techniques to implement an efficient LLM system
- Learn about key ideas in latest LLM systems
    - Learn how to read, evaluate, peer review a (AI sys) paper
- Identify and solve impactful problems for LLM systems
    - and future AI systems

Enrollment

- We have 140 slots based on the room size
- Many on the waitlist:
  - Students drop after two weeks.
  - Please stay if you plan to complete the class
- Audit (=listening to lectures only, no homework):
  - Sorry, no auditing option (university requires enrolling all students on waitlist)
  - It does not help you learn this course if you do not complete homework

Reasons not to take this class … ☹️

If you plan to drop this class, please drop it soon so others can enroll!

- If you want to learn how to train DL models and use TensorFlow, PyTorch
  - Take 11685/11785
- If you want learn how to use LLM but not interested in the underlying system
  - Take 11667
- You are unable to attend lecture, complete homework, project
  - Warning: heavy workload and system implementation

Prerequisites

In order to take this course, you need to be proficient with:

- Systems programming (e.g., 15-213)
- Linear algebra (e.g., 21-240 or 21-241)
- Other mathematical background: e.g., calculus, probability
- Python and C++ development
- Basic prior experience with ML/DL (e.g. 11785 or 11711)

Organization of Lecture Topics

- Half of lectures are about basic techniques for building an LLM sys
  - Taught by the Instructor
- Later half
  - 1 paper assigned for each session
  - Everyone needs to read the paper & critiquing the required paper before class!
  - 45mins presentation by student group (signup link out in week 2)
  - 25mins in-class discussion on the strengths and weakness, impact, future directions
- https://llmsystem.github.io/llmsystem2024spring/

Grading

- 5 Assignments to implement an LLM system
    - Two weeks for each, (10% each)
- Paper Reading/Review (10% total), everyone
- Paper presentation
    - Presented in class (10% + 5%), graded by organization, design, clarity, interaction.
    - Recorded presentation (10%)
- Project (30% or 40%)

Assignments

- The course will consist of five programming-based homework assignments
- Homeworks are done individually
- Through the assignments you will develop the main components for miniTorch and train a Transformer-based LLM

Project

- Team of 2~3

- Two types of projects:
  - Implement a LLM system paper as an additional component to miniTorch (should not use external PyTorch/Tensorflow code)
  - Research oriented project that solves a challenge in LLM system (with the goal of submitting to a conference)

- Need to submit Project Proposal/Mid-term/Final Report
  - Please include: project description, proposed method, data, evaluation procedure/metric, computation, other resources, preliminary results.

- Poster presentation (ideally with demo) in-class

Paper Presentation

- You will be organized into 15 student groups
  - One assigned paper for presentation
  - Each group decide up to 2 presenters to the whole class
  - The rest team member each record a presentation video.

Computing Resources

- We will allocate PSC credits
- We will also distribute complementary AWS credits for course project (will not be available immediately)
- Additionally, you may use colab.