CSE 493s/599s Lecture 19.

Sewoong Oh



Lecture notes

- These lecture notes are based on other courses in LLMs, including
 - CSE493S/599S at UW by Ludwig Schmidt: <u>https://mlfoundations.github.io/advancedml-sp23/</u>
 - EE-628 at EPFL by Volkan Cevher: <u>https://www.epfl.ch/labs/lions/teaching/ee-628-training-large-language-models/ee-628-slides-2025/</u>
 - ECE381V Generative Models at UT Austin by Sujay Sanghavi
 - and various papers and blogs cited at the end of the slide deck

Outline

- Language models
- General LLM framework
 - Token processing
 - Sequence mixing
 - Prediction
- Prompting techniques at inference time
 - In-context learning
 - Chain-of-thought prompting
- Fine-tuning
- Alignment
- Mixture-of-Experts (MoE)

Mixture-of-Experts (MoE)



Fig. 1. A chronological overview of several representative mixture-of-experts (MoE) models in recent years. The timeline is primarily structured according to the release dates of the models. MoE models located above the arrow are open-source, while those below the arrow are proprietary and closed-source. MoE models from various domains are marked with distinct colors: Natural Language Processing (NLP) in green, Computer Vision in yellow, Multimodal in pink, and Recommender Systems (RecSys) in cyan.



• It is common for feed-forward layers to account for

Figure from http://arxiv.org/abs/2409.02060 [Muennighoff et al. 2024]

• One can benefit from scaling up to larger models without increasing as much inference time with **Mixture-of-Experts (MoE)**



• **Dense MoE** gives every expert non-zero weights

•
$$y = \sum_{i=1}^{N} G(x)_i E_i(x)$$

- Each expert is a small FFN:
 - $E_i(x) = W_2 \sigma(W_1 x)$
- Router is a dense softmax
 - $G(x) = \text{Softmax}(W_G x)$



• **Sparse MoE** selects top-*k* experts and is now popular

$$y = \sum_{i=1}^{N} G(x)_i E_i(x)$$

- Each expert is a small FFN:
 - $E_i(x) = W_2 \sigma(W_1 x)$
- Router is a softmax over top-*k* gating

•
$$G(x) = \text{Softmax}(\text{Top-}k(W_G x))$$

. Top- $k(v) = \begin{cases} v_i, v_i \text{ is in the Top } k \\ -\infty, \text{ otherwise} \end{cases}$



- v = [-7, 3, 8, -2]
- **Top**- $k(v) = [-\infty, 3, 8, -\infty]$

Softmax(**Top-**
$$k(v)$$
)
= $[0, \frac{e^3}{e^3 + e^8}, \frac{e^8}{e^3 + e^8}, 0]$

- Many variations of MoE architecture exist
- If left unchecked, the expert gate tends to concentrate on a small number of experts that are popular early in training.
- Proposed solution in OIMoE [Muennighoff et al. 2024]:

$$L = L_{\rm CE} + \alpha L_{\rm LB}$$

 Load balancing regularizer: balances experts within a batch

$$L_{\rm LB} = N \sum_{i=1}^{N} f_i P_i$$



 f_i = fraction of tokens routed to expert i

 P_i = total routing probability allocated to E_i in current batch

• Many variations of regularizers exist to help with training.

- The number of experts chosen, k, is typically small
 - Mixtral (MoE from mistral): k = 1, N = 8
 - OIMoE (MoE from AI2): k = 8, N = 64
- For MoEs the gain is in inference-time computation
 - GPU memory requirement ~= # of total parameters
 - FLOPS computation requirement ~= # of active parameters

where **# of active parameters** is the count of parameters that are selected by the router



Figure 3: Results on MMLU, commonsense reasoning, world knowledge and reading comprehension, math and code for Mistral (7B/8x7B) vs Llama 2 (7B/13B/70B). Mixtral largely outperforms Llama 2 70B on all benchmarks, except on reading comprehension benchmarks while using 5x lower active parameters. It is also vastly superior to Llama 2 70B on code and math.

Figure from http://arxiv.org/abs/2401.04088

 MoEs achieve the frontier of inference-cost vs. performance trade-off for open-source models



Figure 1: Performance, cost, and degree of openness of open MoE and dense LMs. Model names contain rounded parameter counts: model-active-total for MoEs and model-total for dense LMs. #ckpts is the number of intermediate checkpoints available. We highlight MMLU as a summary of overall performance; see §3 for more results. OLMOE-1B-7B performs best among models with similar active parameter counts and is the most open MoE.

Figure from http://arxiv.org/abs/2409.02060



MoEs achieve favorable training-cost vs. performance trade-off

Figure 3: Evaluation of OLMOE-1B-7B and the current best OLMo models during pretraining. OLMOE-1B-7B differs from the OLMo models in its MoE architecture, several training hyperparameters, and its training dataset, see §2. A version of this plot with tokens as the x-axis and markers where annealing starts is in Appendix E. More results, logs, and configurations: https://wandb.ai/ai2-llm/olmoe/ reports/Plot-OLMoE-1B-7B-vs-OLMo-7B-vs-OLMo-1B--Vmlldzo40TcyMjEz

Figure from http://arxiv.org/abs/2409.02060

• Earlier works in MoE designed for RNNs favored very large number of experts



More recent MoEs for transformers require comparably smaller number of experts.



Figure 5: Expert granularity. We vary the number of experts in tandem with the FFN dimension to ensure that active and total parameters and thus compute cost remain the same. For example, for 64 experts, the FFN dimension is 1,024 and 8 experts are activated, while for 32 experts it is 2,048 with 4 activated experts. More results, logs, and configurations: https://wandb.ai/ai2-llm/olmoe/reports/Plot-Granularity--Vmlldzo40TIx0TE4

Figure from http://arxiv.org/abs/2409.02060

Mixture-of-Experts (MoE)

- How to improve performance of LMs?
- Scaling Pre-training

[2020-] Scaling pretraining: larger model, larger dataset

• Scaling Post-training

• Test-time scaling





Scaling Instruction-Finetuned Language Models [Chung et al., 2022]

lest time sealing



[Now] Test-time scaling: increase compute at generation time

Scaling Laws for Neural Language Models [Kaplan et al., 2020]

Test-time compute vs. accuracy ([OpenAI, 2024])

[Welleck et al. 2024]

• Goal: design a system G that generates acceptable sequences:

$$\arg\max_{G} \mathbb{E}_{y \sim G} A(y)$$

where acceptable, $A(\cdot)$, is measured by correctness, human preference, etc.

• We have a retrained model that can sample from $y \sim P_{\theta}(y | x)$, which may produce unacceptable samples.



• If we have an oracle that can verify correctness of an output, one can repeat

x:

z:

y:

•
$$z \sim P_{\theta}(z \mid x)$$

•
$$y \sim P_{\theta}(y \mid x, z)$$

 stop if oracle says it is correct Input: Let $f(r) = \sum_{j=2}^{2008} \frac{1}{j^r} = \frac{1}{2^r} + \frac{1}{3^r} + \dots + \frac{1}{2008^r}$. Find $\sum_{k=2}^{\infty} f(k)$. LLEMMA 34B solution: We have $\sum_{k=2}^{\infty} f(k) = \sum_{k=2}^{\infty} \sum_{i=2}^{2008} \frac{1}{j^k} = \sum_{i=2}^{2008} \sum_{k=2}^{\infty} \frac{1}{j^k} = \sum_{i=2}^{2008} \frac{1}{j^2} \sum_{k=0}^{\infty} \frac{1}{j^k} = \sum_{i=2}^{2008} \frac{1}{j^2} \frac{1}{1 - \frac{1}{j^2}}$ $=\sum_{i=2}^{2008}\frac{1}{j(j-1)}=\sum_{i=2}^{2008}\left(\frac{1}{j-1}-\frac{1}{j}\right)$ $=\left(\frac{1}{1}-\frac{1}{2}\right)+\left(\frac{1}{2}-\frac{1}{3}\right)+\cdots+\left(\frac{1}{2007}-\frac{1}{2008}\right)$ $= 1 - \frac{1}{2008}$ 2007 = 2008 Final Answer: The final answer is $\frac{2007}{2008}$.

• When oracle is correct, this strategy increases accuracy.



¹Adapted from [Brown et al., 2024]. See also [Li et al., 2022, Cobbe et al., 2021, Jiang et al., 2023]

 In general, a meta-generator G uses multiple generative model calls and other tools, hyper-parameters, prompts:

$$y \sim G(y \mid x; g_1, \dots, g_G, \phi)$$

- Design choices:
 - *G* : strategy for calling generators
 - g_1, \ldots, g_G : choices of generators
 - ϕ :other models, number of tokens, etc.

- Meta-generation strategies include:
 - Chain
 - Parallel
 - Tree search
 - Refinement/self-correction



• Chained meta-generators call the LM repeatedly, with augmented prompts.

$$y_1 \sim g_1(x)$$

 $y_2 \sim g_1(x, y_1)$
 $y_3 \sim g_3(x, y_2)$
:



• Chain-of-Thought is an example:



• This auto-regressive decoding of CoT can be represented by generate thought: $z \sim P_{\theta}(z \mid x)$ generate answer: $y \sim P_{\theta}(y \mid x, z)$

• Search is an example of multiple step chained meta-generation, with access to tools like search engine calls:

GPT-3
Question: In what year was the current tallest wooden lattice tower completed? Are follow up questions needed here: Yes. Follow up: What is the current tallest wooden lattice tower?
Search Engine Response: Radio Tower Gliwice
GPT-3 Intermediate answer: Radio Tower Gliwice. Follow up: When was Gliwice Radio Tower completed?
Search Engine Response: 1935
GPT-3 Intermediate answer: 1935. So the final answer is: 1935.
Self-Ask [Press et al., 2023]

• This can be represented as a program, which typically improves the reasoning:

```
def search(x: Example) -> Example:
    x.hop1 = generate(hop_template)(x).pred
    x.psg1 = retrieve(x.hop1, k=1)[0]
    x.hop2 = generate(hop_template)(x).pred
    x.psg2 = retrieve(x.hop2, k=1)[0]
    return x
def predict(x: Example) -> Example:
    x.context = [x.psg1, x.psg2]
    x.pred = generate(qa_template)(x).pred
    return x
```

Demonstrate-Search-Predict (DSP) [Khattab et al., 2022]

- Many other strategies
 - Re-write input before feeding in to LM for an answer: System-2 attention [Weston and Sukhbaater, 2023]
 - Sketch proof, fill gaps, check proof Draft-sketch-prove [Jiang et al., 2023]



• Chained meta-generators only utilize/control the input space.

- Parallel meta-generators generate multiple outputs to select from.
 - Generate candidates: $\{y^{(1)}, \dots, y^{(m)}\} \sim G(\ \cdot \ | \ x)$
 - Aggregate:
 - $y = h(y^{(1)}, ..., y^{(m)})$



- Best-of-*m* sampling and rejection sampling
 - Aggregate: with a reward model:

$$y \leftarrow \arg \max_{y^{(1)}, \dots, y^{(m)}} A(y)$$



• This can suffer from over optimizing to the reward $A(\cdot)$



⁷Plot adapted from *Training Verifiers to Solve Math Word Problems* [Cobbe et al., 2021]

- Majority voting and self-consistency does not require external reward model.
 - Aggregation:

$$y \leftarrow \arg\max_{a} \sum_{i=1}^{m} \mathbb{I}\{y^{(i)} = a\},\$$



- You can combine best-of-*m* and voting:
- Weighted aggregation:

$$y \leftarrow \arg\max_{a} \sum_{i=1}^{m} A(y^{(i)}) \cdot \mathbb{I}\{y^{(i)} = a\}$$



- Weighted voting can out-perform best-of-*m*
 - This happens when weighted maximum reward is more likely to give the correct answer



¹⁰[Sun et al., 2024] Easy-to-Hard Generalization: Scalable Alignment Beyond Human Supervision.
Z. Sun, L. Yu, Y. Shen, W. Liu, Y. Yang, S. Welleck, C. Gan. NeurIPS 2024.

• For large *m*, this converges to maximum conditional expectation:

$$y \leftarrow \arg\max_{a} \sum_{z} V(x, z, y = a)g(z, y = a \mid x)$$

- One can potentially further improve the performance by
 - using better reward, and
 - using a better generator that reasons better



• Parallel meta-generators use the reward in the end.

12: [Zhang et al., 2024]

• **Tree search meta-generators** can evaluate reward on a sequence of intermediate states and the final outcome.



• **Tree search meta-generators** can evaluate reward on a sequence of intermediate states and the final outcome.



¹³[Uesato et al., 2022, Lightman et al., 2024, Wang et al., 2024a]

• Selects which state to explore next by taking the reward into account.



¹⁴[Wu et al., 2024b] Inference Scaling Laws: An Empirical Analysis of Compute-Optimal Inference.

• Aggregates the final answer using, e.g., weighted voting



• Key idea in tree search meta-generator is leveraging intermediate states and rewards to back-track/explore.



¹⁵[Wu et al., 2024b] Inference Scaling Laws: An Empirical Analysis of Compute-Optimal Inference.

Tree search meta-generators require good reward functions over well-defined states.

• **Refinement/self-correction meta-generators** use past generation and correct mistakes.



• Quality of the feedback is critical.



• There are extrinsic/intrinsic feedback.



Intuition: adds new information, can detect and localize errors



• There are extrinsic/intrinsic feedback.



Easy to evaluate but results are mixed.

• One can train a corrector based on samples generated and optimized to maximize reward.



¹⁸E.g., Self-corrective learning [Welleck et al., 2023], SCoRe [Kumar et al., 2024].



First 1/2 of CSE492S/599S Advanced ML

Part I: Foundations

- A gentle start
- A formal learning model
- Learning via uniform convergence
- The bias-complexity trade-off
- The VC-dimension
- Non-uniform learnability
- The runtime of learning

Shai Shalev-Shwartz and Shai Ben-David UNDERSTANDING MACHINE LEARNING FROM THEORY TO ALGORITHMS

Part II: From Theory to Algorithms

- Linear predictors
- Boosting
- Model selection and validation
- Convex learning problems
- Regularization and stability
- Stochastic gradient descent
- Support vector machines
- Kernel methods
- Multiclass, ranking, and complex prediction problems
- Decision trees
- Nearest neighbor
- Neural networks

Part III: Additional Learning Models

- Online learning
- Clustering
- Dimensionality reduction
- Generative models
- Feature selection and generation

Part IV: Advanced Theory

- Rademacher complexities
- Covering numbers
- Proof of the fundamental theorem of learning theory
- Multiclass learnability
- Compression bounds
- PAC-Bayes

Second 1/2 of CSE492S/599S Advanced ML

- Language models
- General LLM framework
 - Token processing
 - Sequence mixing
 - Prediction
- Prompting techniques at inference time
 - In-context learning
 - Chain-of-thought prompting
- Fine-tuning
- Alignment
- Mixture-of-Experts (MoE)
- Test-time compute

Sources

- Other courses in LLMs that the lecture slides are based on
 - CSE493S/599S at UW by Ludwig Schmidt: <u>https://mlfoundations.github.io/advancedml-sp23/</u>
 - EE-628 at EPFL by Volkan Cevher: https://www.epfl.ch/labs/lions/teaching/ee-628-training-large-language-models/ee-628-slides-2025/
 - https://sharif-llm.ir/assets/lectures/Chain-of-Thought-Prompting.pdf
 - https://www.cs.princeton.edu/courses/archive/fall22/cos597G/lectures/lec09.pdf
- Useful blog posts
 - https://azizbelaweid.substack.com/p/complete-summary-of-absolute-relative
 - https://blog.dust.tt/speculative-sampling-llms-writing-a-lot-faster-using-other-llms/
 - <u>https://gordicaleksa.medium.com/eli5-flash-attention-5c44017022ad</u>
 - <u>https://medium.com/@dillipprasad60/qlora-explained-a-deep-dive-into-parametric-efficient-fine-tuning-in-large-language-models-llms-c1a4794b1766</u>
 - https://cmu-l3.github.io/neurips2024-inference-tutorial/
- Dan Jurafsky and James H. Martin. Speech and Language Processing (3rd ed. draft). draft, third edition, 2023.
- Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space", In International Conference on Learning Representations, 2013.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation", Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.
- Ofir Press, Noah A. Smith1,3 Mike Lewis2, "Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation", In International Conference on Learning Representations, 2022
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, "Attention Is All You Need", In Neural Information Processing Systems, 2017
- Beitong Zhou, Cheng Cheng, Guijun Ma, and Yong Zhang. "Remaining useful life prediction of lithium-ion battery based on attention mechanism with positional encoding", In IOP Conference Series: Materials Science and Engineering, 2020.
- Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. "On the difficulty of training recurrent neural networks." In International Conference on Machine Learning, 2013

- Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory." In Neural Computation, 9(8):1735–1780, 11 1997.
- Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. "Learning phrase representations using rnn encoder-decoder for statistical machine translation", In ACL 2014
- Andrey Andreyevich Markov. "Essai d'une recherche statistique sur le texte du roman. 'Eugene Onegin' illustrant la liaison des epreuve en chain". In: Izvistia Imperatorskoi Akademii Nauk (Bulletin de l'Académie Impériale des Sciences de St.-Pétersbourg). 6th ser, 7:153–162, 1913.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, Yejin Choi, "The Curious Case of Neural Text Degeneration", In International Conference on Learning Representations, 2020
- Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre and John Jumper, "Accelerating Large Language Model Decoding with Speculative Sampling" In, ACL-findings, 2024
- Sergey loffe, Christian Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", In International Conference on Machine Learning, 2015
- Shibani Santurkar MIT shibani@mit.edu Dimitris Tsipras MIT tsipras@mit.edu Andrew Ilyas MIT ailyas@mit.edu Aleksander Madry, "How Does Batch Normalization Help Optimization?", In Advances in Neural Information Processing Systems 31 (NeurIPS 2018)
- Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton, "Layer Normalization ", In 2016
- Tianyu Gao, Adam Fisch, Danqi Chen, "Making Pre-trained Language Models Better Few-shot Learners", In ACL, 2021
- Sewon Min1,2 Xinxi Lyu1 Ari Holtzman1 Mikel Artetxe2 Mike Lewis2 Hannaneh Hajishirzi1,3 Luke Zettlemoyer, "rethinking the role of demonstrations what makes in conte..."
- Hila Gonen1,2 Srini Iyer2 Terra Blevins1 Noah A. Smith1,3 Luke Zettlemoyer1, "Demystifying Prompts in Language Models via Perplexity Estimation"
- E Akyürek, B Wang, Y Kim, J Andreas, "In-context language learning: Architectures and algorithms", 2024
- What learning algorithm is in-context learning? Investigations with linear models Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, Denny Zhou, 2022
- Ziqian Lin, Kangwook Lee, "Dual Operating Modes of In-Context Learning", 2024
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models", In NeurIPS 2022
- Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, Yusuke Iwasawa, "Large Language Models are Zero-Shot Reasoners", In NeurIPS 2022
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, Denny Zhou, "Self-Consistency Improves Chain of Thought Reasoning in Language Models", In ICLR 2023

- Shunyu Yao · Jeffrey Zhao · Dian Yu · Nan Du · Izhak Shafran · Karthik Narasimhan, Yuan Cao, "ReAct: Synergizing Reasoning and Acting in Language Models", In ICLR 2025
- Satyapriya Krishna1, Kalpesh Krishna2, Anhad Mohananey†2, Steven Schwarcz2, Adam Stambler2, Shyam Upadhyay2, Manaal Faruqui*3, "Fact, Fetch, and Reason: A Unified Evaluation of Retrieval-Augmented Generation"
- Salaheddin Alzubi, Creston Brooks, Purva Chiniya, Edoardo Contente, Chiara von Gerlach, Lucas Irwin, Yihan Jiang, Arda Kaz, Windsor Nguyen, Sewoong Oh, Himanshu Tyagi, Pramod Viswanath, "Open Deep Search: Democratizing Search with Open-source Reasoning Agents", <u>https://arxiv.org/abs/2503.20201</u>
- Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, Heng Ji, "Executable Code Actions Elicit Better LLM Agents", In *ICML* 2024
- Peter Shaw, Jakob Uszkoreit Ashish Vaswani, "Self-Attention with Relative Position Representations", 2018
- Ofir Press1,2 Noah A. Smith1,3 Mike Lewis2, "TRAIN SHORT, TEST LONG: ATTENTION WITH LINEAR BIASES ENABLES INPUT LENGTH EXTRAPOLATION", 2022
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, Luke Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs", In NeurIPS 2023
- Fanxu Meng1,2, Zhaohui Wang1, Muhan Zhang1,2*, "PiSSA: Principal Singular Values and Singular Vectors Adaptation of Large Language Models" In *NeurIPS* 2024
- Hanqing Wang, Yixia Li, Shuo Wang, Guanhua Chen, Yun Chen, "MiLoRA: Harnessing Minor Singular Components for Parameter-Efficient LLM Finetuning", In NAACL 2025
- Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, Jacek Tabor, "LoRA-XS: Low-Rank Adaptation with Extremely Small Number of Parameters"
- Bingcong Li, Liang Zhang, Aryan Mokhtari, Niao He, "On the Crucial Role of Initialization for Matrix Factorization.", In ICLR 2025
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. "Scaling laws for neural language models", 2020.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack William Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, Advances in Neural Information Processing Systems, 2022.

- Sean McLeish, John Kirchenbauer, David Yu Miller, Siddharth Singh, Abhinav Bhatele, Micah Goldblum, Ashwinee Panda, and Tom Goldstein. "Gemstones: A model suite for multi-faceted scaling laws", 2025
- Sean Welleck Sean Welleck1 Amanda Bertsch Amanda Bertsch1 Matthew Finlayson Matthew Finlayson2 Alex Xie Alex Xie1 Graham Neubig Graham Neubig1 Konstantin Golobokov Konstantin Golobokov5 Hailey Schoelkopf3 Ilia Kulikov Ilia Kulikov4 Zaid Harchaoui Zaid Harchaoui5, "Neurips 2024 Tutorial: Beyond Decoding: Meta-Generation Algorithms for Large Language Models"