Reasoning is all the rage these days. If you want to save some time and get to the crux of how to enable reasoning in LLMs, here's a list of 10 recent papers that I find most informative, along with my notes:

(Full thread in doc: https://docs.google.com/document/d/1TW7wEUgo61FZnPckZMploGTdB0eNcemiDPDqdmzsCvA/edit?usp=sharing)

1/

DeepSeek-R1 (https://arxiv.org/abs/2501.12948) @deepseek_ai

R1-Zero:

- RL on the base model with GRPO
- Rule-based reward (answer accuracy & format)
- Reasoning emerges and CoT lengths increase throughout RL training

R1:

- SFT on long CoT data first
- Then do reasoning RL, also included language consistency reward to avoid language mixing in reasoning
- Gather new reasoning SFT data (800k) from the RL checkpoint w/ rejection sampling and more diverse prompts
- SFT with new data + RL w/ reasoning & general prompts

Distillation:

- Finetune Qwen & Llama on the 800k SFT data
- For smaller models, distillation can be better than RL. E.g., DeepSeek-R1-Distill-Qwen-32B > DeepSeek-R1-Zero-Qwen-32B ~ QwQ-32B-Preview

2/

s1 (https://arxiv.org/abs/2501.19393) @Muennighoff @ZitongYang0 @WeijiaShi2 @XiangLisaLi2

- First collected 59K reasoning SFT data on diverse math & reasoning questions with Gemini Flash Thinking
- Then filter down to 1K samples optimizing difficulty & diversity (s1K)
- SFT on Qwen2.5-32B-Instruct to get s1-32B
- Budget forcing during test time: end thinking by appending end-of-thinking token; or extend thinking by appending “Wait” to reasoning trace.
- Results: s1-32B shows large gains compared to Qwen2.5-32B-Instruct on AIME (26.7->56.7), MATH (84.0->93.0), GPQA (49.0->59.6); extending reasoning length by budget forcing shows clear positive scaling trends.

In a similar vein, LIMO (https://arxiv.org/abs/2502.03387 @BLeavesYe @Z_Huang_02 @stefan_fee) curated 817 reasoning SFT examples to finetune Qwen2.5-32B-Instruct and beat QwQ-32B-preview.

3/

Demystifying Reasoning (https://arxiv.org/abs/2502.03373) @eddy_data3 @tongyx361 @xiangyue96

Some main findings:

Experiment #1: SFT w/ long CoT (from QwQ-32B-Preview) vs short CoT (Qwen2.5-Math-72B-Instruct) on the base model Llama-3.1-8B.
Results: SFT with long CoT can scale up to a higher performance upper limit than short CoT.

Experiment #2: RL (PPO w/ rule-based reward with MATH training prompts) on top of the SFT checkpoints from Experiment #1.
Results: Models initialized with long CoT SFT can usually be further significantly improved by RL, while models initialized with short CoT SFT see little gains.

Experiment #3: RL with Cosine Reward (shorter correct CoTs receive higher rewards than longer correct CoTs, shorter wrong CoTs receive higher penalties than longer wrong CoTs).
Results: Cosine Reward significantly stabilized the length scaling and training accuracy.

Experiment #4: RL with Cosine Reward + Repetition Penalty to avoid length reward hacking (longer CoT by just repeating).
Results: The repetition penalty resulted in better downstream task performance and shorter CoTs.

Experiment #5: Adding noisy verifiable data (WebInstruct) to SFT & RL.
Results: Adding noisy but diverse data to SFT can help (mostly on general benchmarks like MMLU-Pro); RL w/ noisy data w/ rule-based reward on extracted short answers or model-based reward (on free-form responses) can have moderate gains esp. with proper filtering to find more easily verifiable data.

4/

SFT Memorizes, RL Generalizes (https://arxiv.org/abs/2501.17161)
@TianzheC @simon_zhai

How well does SFT vs RL generalize to OOD settings? Two eval tasks:

- GeneralPoints (generalized points 24 game): The OOD variation is to change whether JQK is interpreted as 11/12/13 or all 10.
- V-IRL (spatial navigation): The OOD variation is to switch between absolute/relative orientation action space.

Experiments:

- SFT / RL on Llama-3.2-Vision-11B (RL is initialized with SFT)
- RL consistently improves OOD performance, while SFT consistently exhibits performance degradation across all OOD evaluations

- Conclusions hold for vision-language settings as well (see paper for details)

On more realistic tasks, see the comparison of SFT vs RL models on AIME 2025: https://x.com/WenhuChen/status/1888691381054435690 and https://x.com/BLeavesYe/status/1888644837278437573

But also note the potential data contamination caveat: https://x.com/DimitrisPapail/status/1888325914603516214

5/

AceCoder (https://arxiv.org/abs/2502.01718) @DongfuJiang @WenhuChen

A lot of success on math reasoning with rule-based reward, what about coding?

Reward Model:
- First curated a coding dataset AceCode-89K. Generate and quality-filter test cases for each problem.
- Sample multiple programs for each problem, use test-case pass rate to construct preference pairs.
- Train the reward model on these preference pairs.

BoN & RL:
- Directly using the RM during inference for Best-of-N can boost performance
- RL with rule-based reward (pass rate) or model-based reward (RM) has some gains. The gain is pretty big directly doing RL on a base model without SFT (Qwen2.5-Coder-7B-Base), esp with rule-based reward.

6/

There May Not Be Aha Moment (https://oatllm.notion.site/oat-zero) @zzlccc @Cameron_Chann @liwenjun2016 @TianyuPang1

Some observations on the actual reasoning traces:

- Base models (Qwen, DeepSeek, Llama) can exhibit some self-reflection patterns even without any post-training, esp. Qwen-2.5 models.
- But these self-reflections by the base models are often wrong/superficial.
- RL training dynamics:
    - In the first phase, the format reward dominates, the lengthy incorrect responses are suppressed, hence the average response length drops
    - In the second phase, model starts to climb on the correctness reward by outputting more retries, hence a length increase in correct responses.
    - Throughout RL, more superficial self-reflection turned into effective self-reflection.

7/

Other R1/R1-Zero style experiments:

- From @jiayi_pirate and team:
  https://x.com/jiayi_pirate/status/1882839370505621655
  Qwen-2.5 models on the simple countdown and multiplication tasks; works for either Base or Instruct models with size >= 1.5B; and works with any of PPO, GRPO, and PRIME.
- From @junxian_he and team:
  https://x.com/junxian_he/status/1883183099787571519
  PPO (rule-based reward) on Qwen2.5-Math-7B-Base with 8K training data from MATH; big gains even without any SFT before RL.
- From @michaelzluo @sijun_tan and team:
  https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2
  GRPO on Deepseek-R1-Distilled-Qwen-1.5B; increase context length (8K->16K->24K) in later stages of training; big gains over the distilled baseline model, surpassing o1-preview on AIME 2024 and MATH 500.

8/

Apart from natural language CoT, another alternative is to reason through latent CoT.

Chain of Continuous Thought (COCONUT; https://arxiv.org/abs/2412.06769) @Ber18791531 @tydsh

- Directly feed the last hidden state as the input embedding for the next (thinking) token
- Use language CoT data for multi-stage training: the initial stage trains on full language CoT; afterwards at the k-th stage, the first k reasoning steps in the CoT are replaced with k × c continuous thinking tokens (c is a hyper-parameter).
- Training objective is NLL loss on the language CoT and answer tokens. GPT-2 as base models.
- During inference, insert <bot> after the input question to start latent thinking and pad thinking tokens until a specified length.
- Results: on GSM8K, COCONUT is much better than no-CoT but worse than language CoT (with much fewer thinking tokens); on synthetic logical reasoning, COCONUT can be even better than language CoT.

9/

Recurrent Depth Transformer (https://www.arxiv.org/abs/2502.05171) @jonasgeiping @tomgoldsteincs

- Architecture: first the embedding block, then the recurrent block, and lastly the un-embedding block. Each block has multiple Transformer layers.
- The model repeatedly applies the recurrent blocks multiple times, each time taking the previous latent state and the input embedding as input, producing the new latent state as output.
- During training, randomly sample the recurrent iterations (log-normal Poisson distribution) for every input to enable extrapolation during inference.
- Trained on 800B tokens with a data mixture skewed towards reasoning. {2,4,4} layers for the embedding / recurrent / un-embedding blocks; 3.5B parameters in total; mean recurrence during training = 32.

- Results: match/beat OLMo-7B on math/coding benchmarks; significantly better than non-recurrent baseline when controlled for training tokens; although performance seems to saturate beyond 32 recurrences (Figures 7/8/9) during inference, which is the training-time average recurrence.

10/

Shout out to @xiangyue96 @jiayi_pirate @StevenyzZhang @neilbband for helpful discussion!

11/11

Additional References:

- OpenThinker-32B: https://www.open-thoughts.ai/blog/scale
- Structure, not content, is what matters for reasoning distillation: https://arxiv.org/abs/2502.07374
- Hierarchical LLM Reasoning: https://arxiv.org/abs/2502.06772
- 1B reasoning LLM: https://ryanliu112.github.io/compute-optimal-tts/
- GRPO vs PPO on Tulu: https://x.com/vwxyzjn/status/1889728091401973968?s=46