Transformers

Nathan Sprague

LLM Capability Trend

Naively extrapolating the trend in horizon length implies that AI will reach a time horizon of >1 month (167 work hours) between late 2028 and early 2031 (Figure 12).

Kwa, Thomas, et al. "Measuring ai ability to complete long tasks." arXiv preprint arXiv:2503.14499 (2025) (METR – Model Evaluation and Threat group)

Counterpoint...

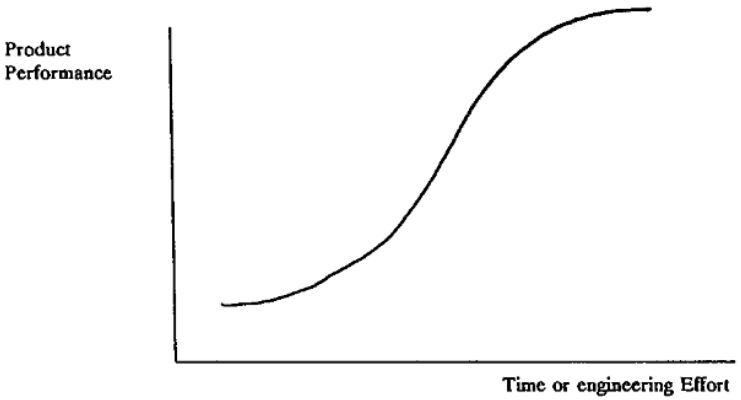
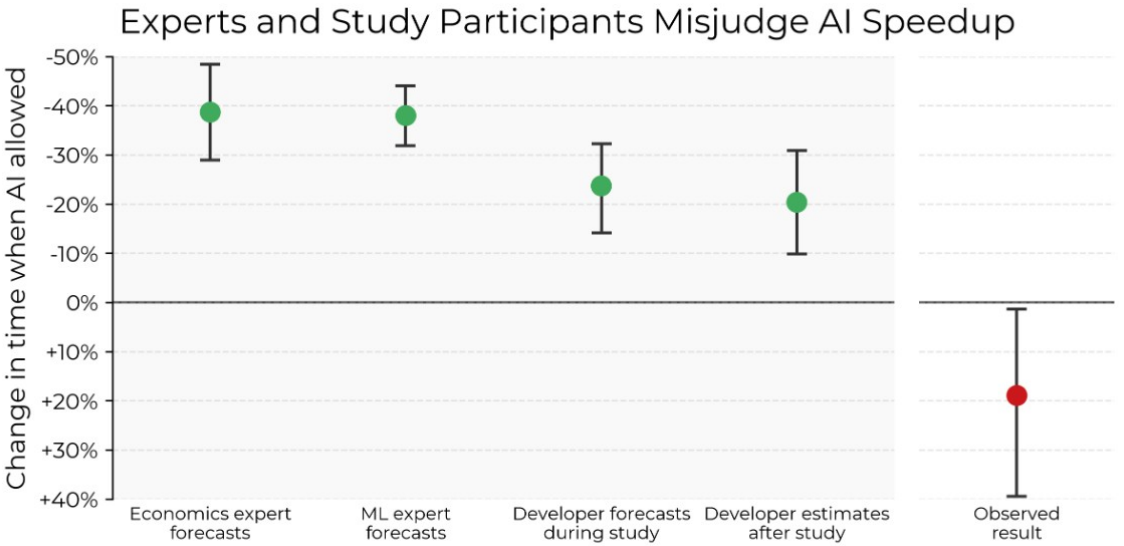


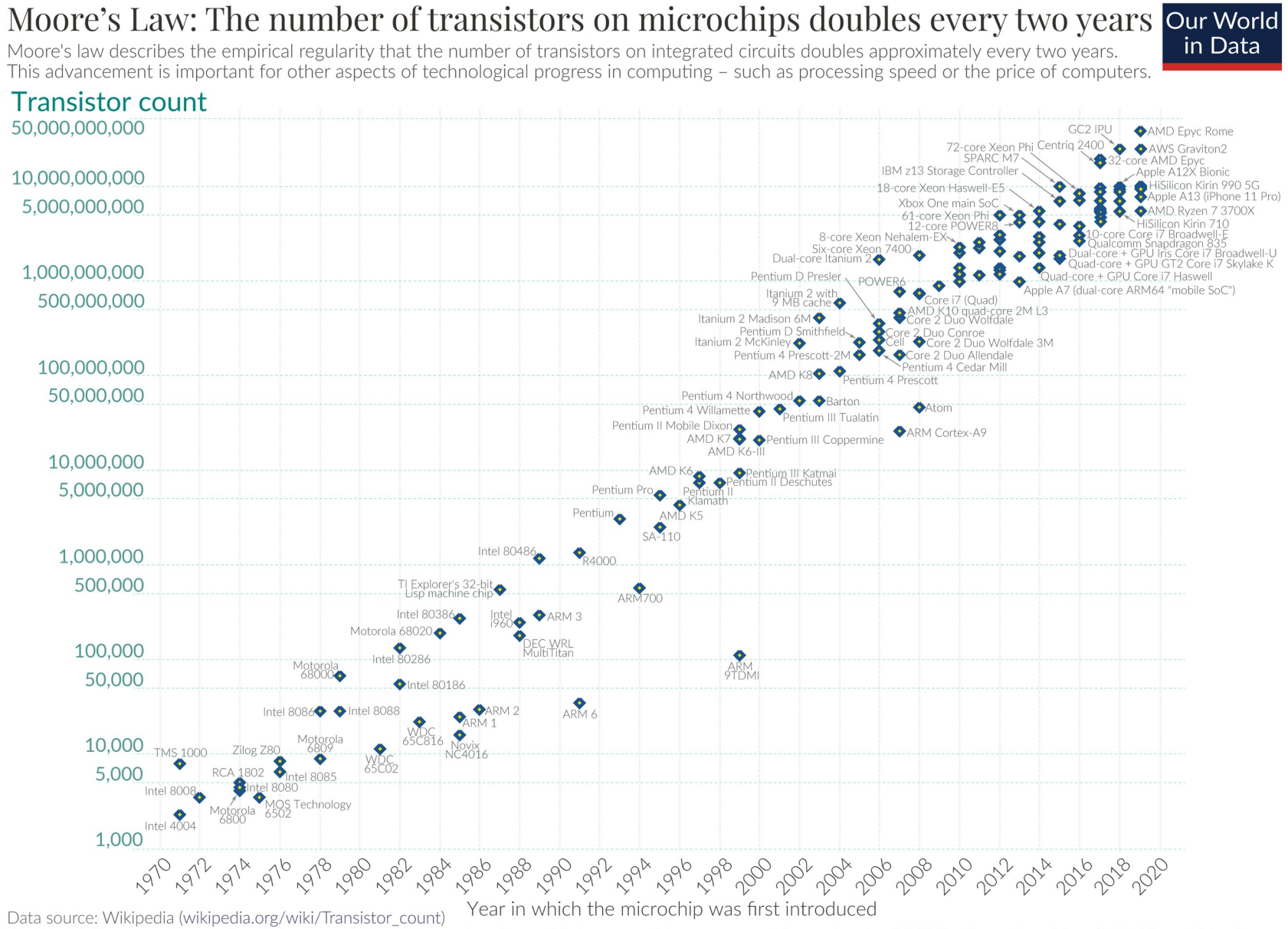
FIGURE 1. The Technology S-Curve

Christensen, Clayton M. "Exploring the limits of the technology S-curve. Part I: component technologies." Production and operations management 1.4 (1992): 334-357.



Becker, Joel, et al. "Measuring the impact of early-2025 ai on experienced open-source developer productivity." arXiv preprint arXiv:2507.09089 (2025).

Counter-Counterpoint…

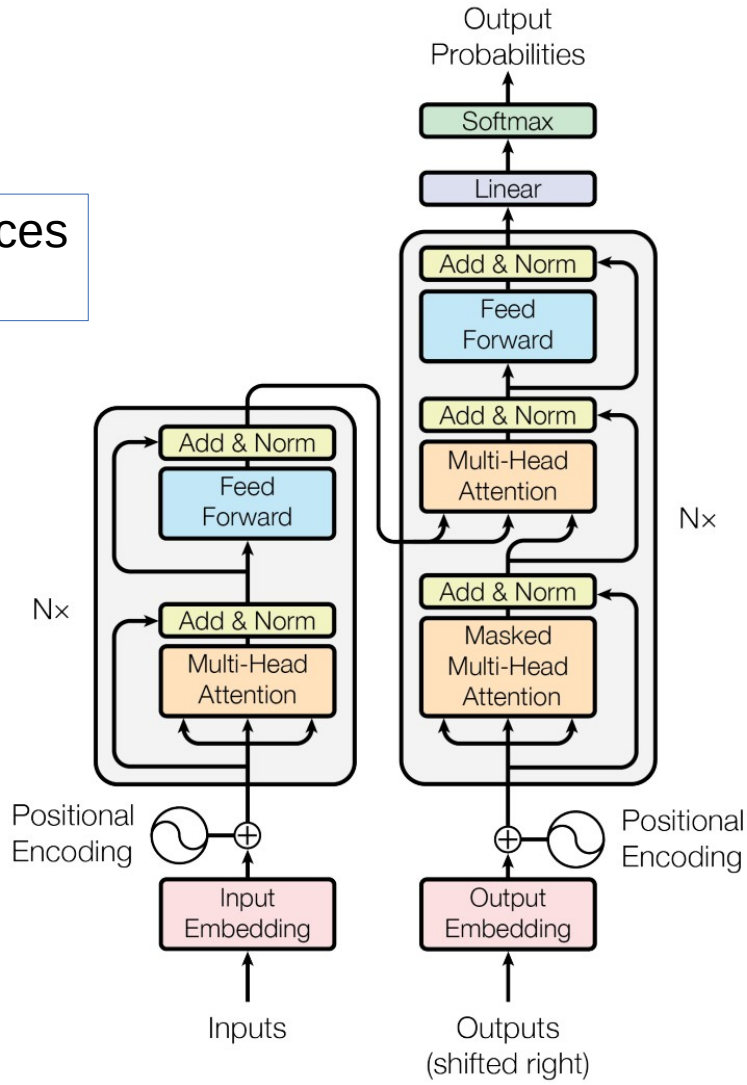


Transformer Networks

- Introduced in:

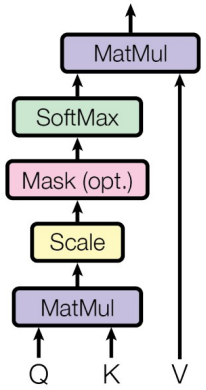
A. Vaswani et al., *Attention is All you Need*, in Advances in Neural Information Processing Systems 30, 2017.

  - (200,000 citations in Google Scholar)

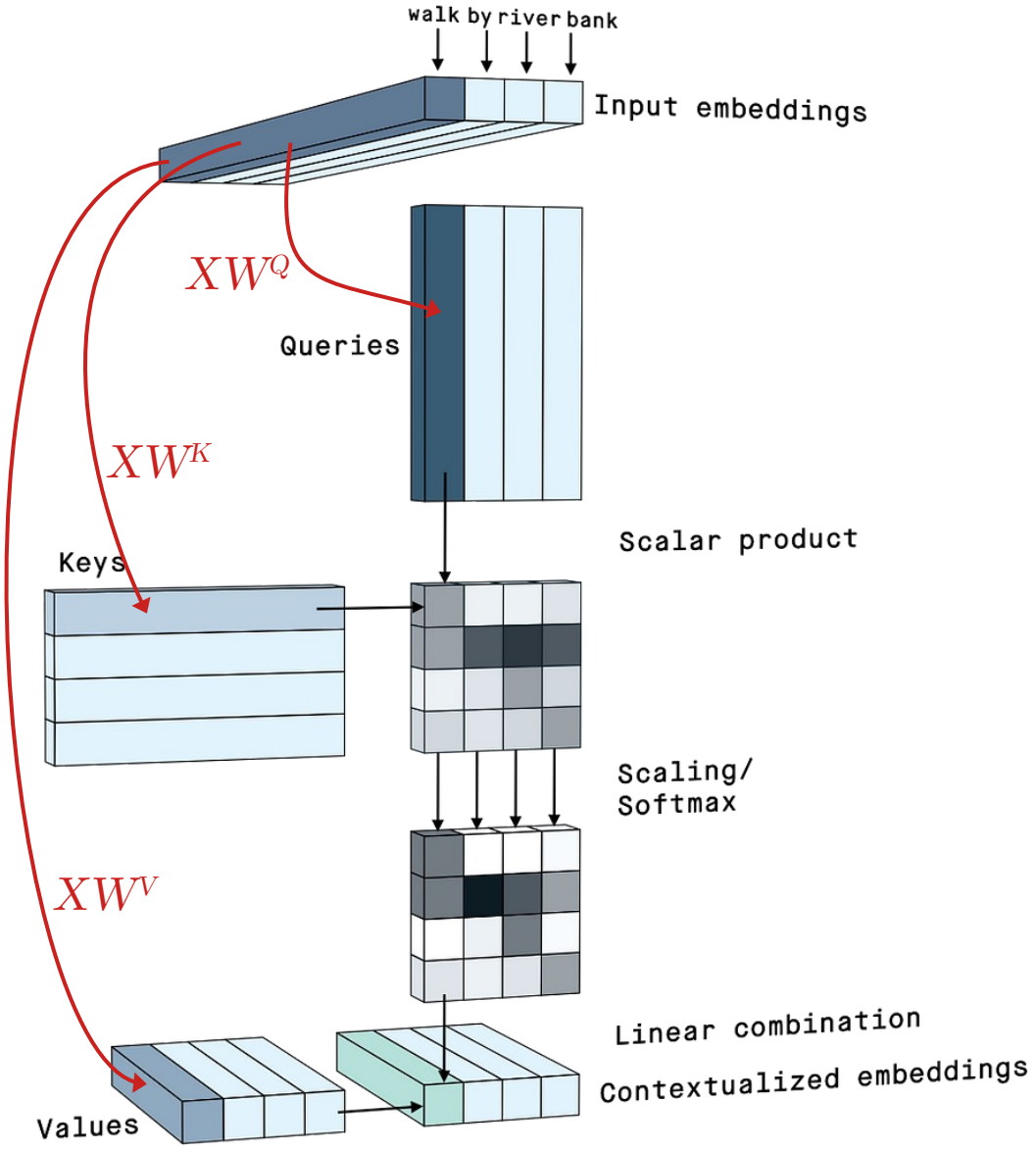


Transformer Self Attention

Scaled Dot-Product Attention



$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

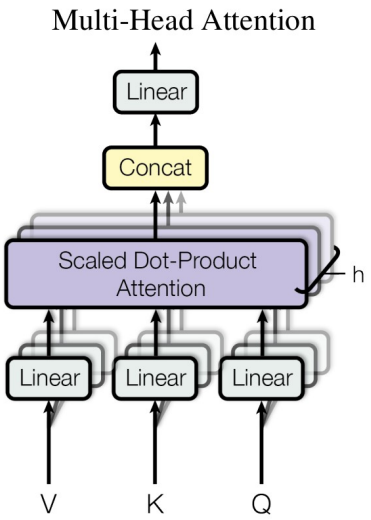


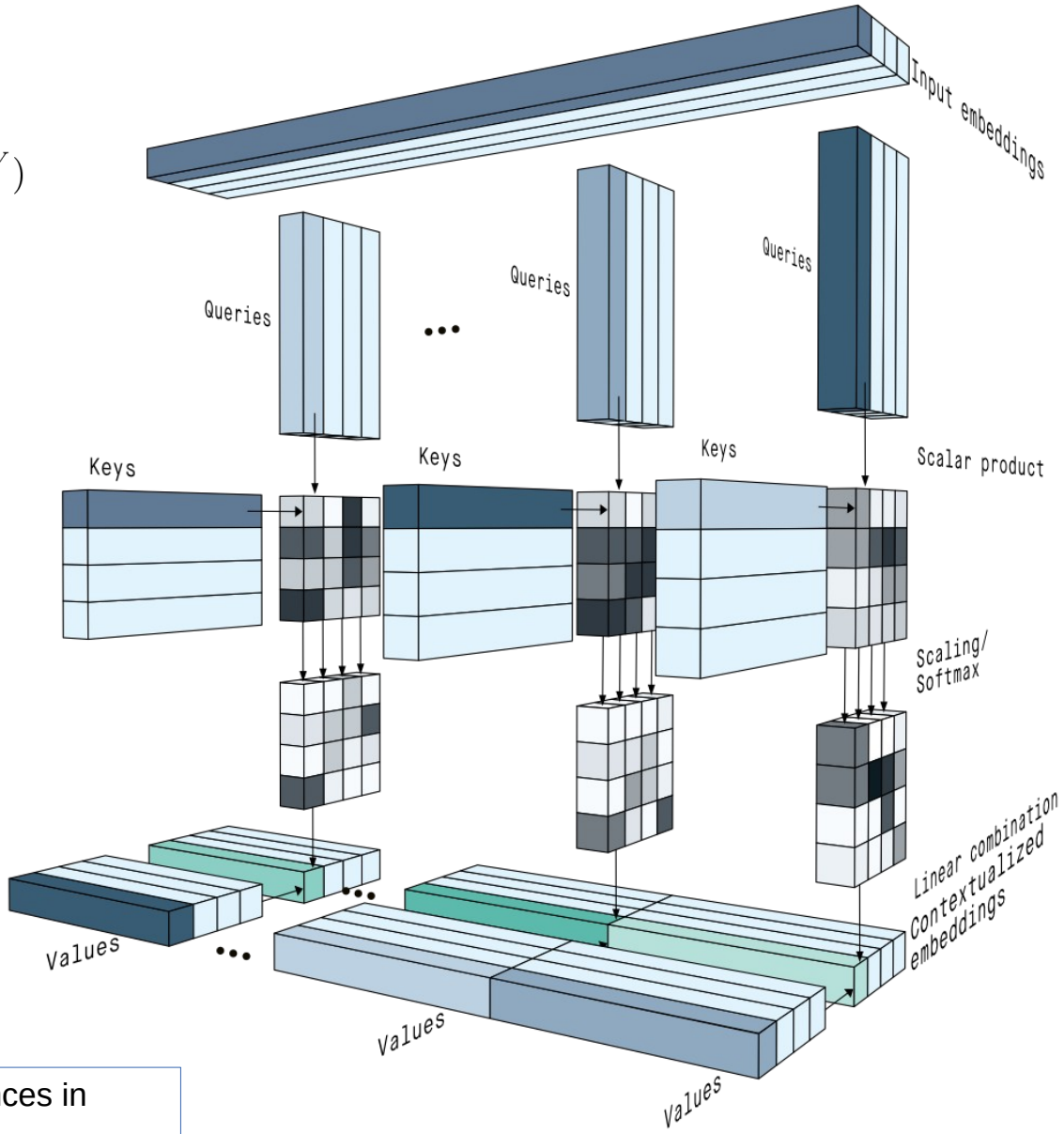
https://towardsai.net/p/nlp/getting-meaning-from-text-self-attention-step-by-step-video

A. Vaswani et al., “Attention is All you Need,” in Advances in Neural Information Processing Systems 30, 2017.

Transformer: Multi-Head Attention

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



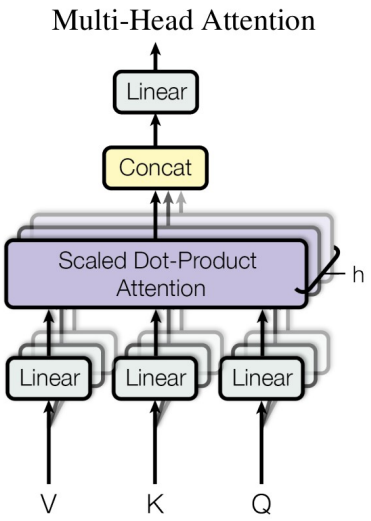


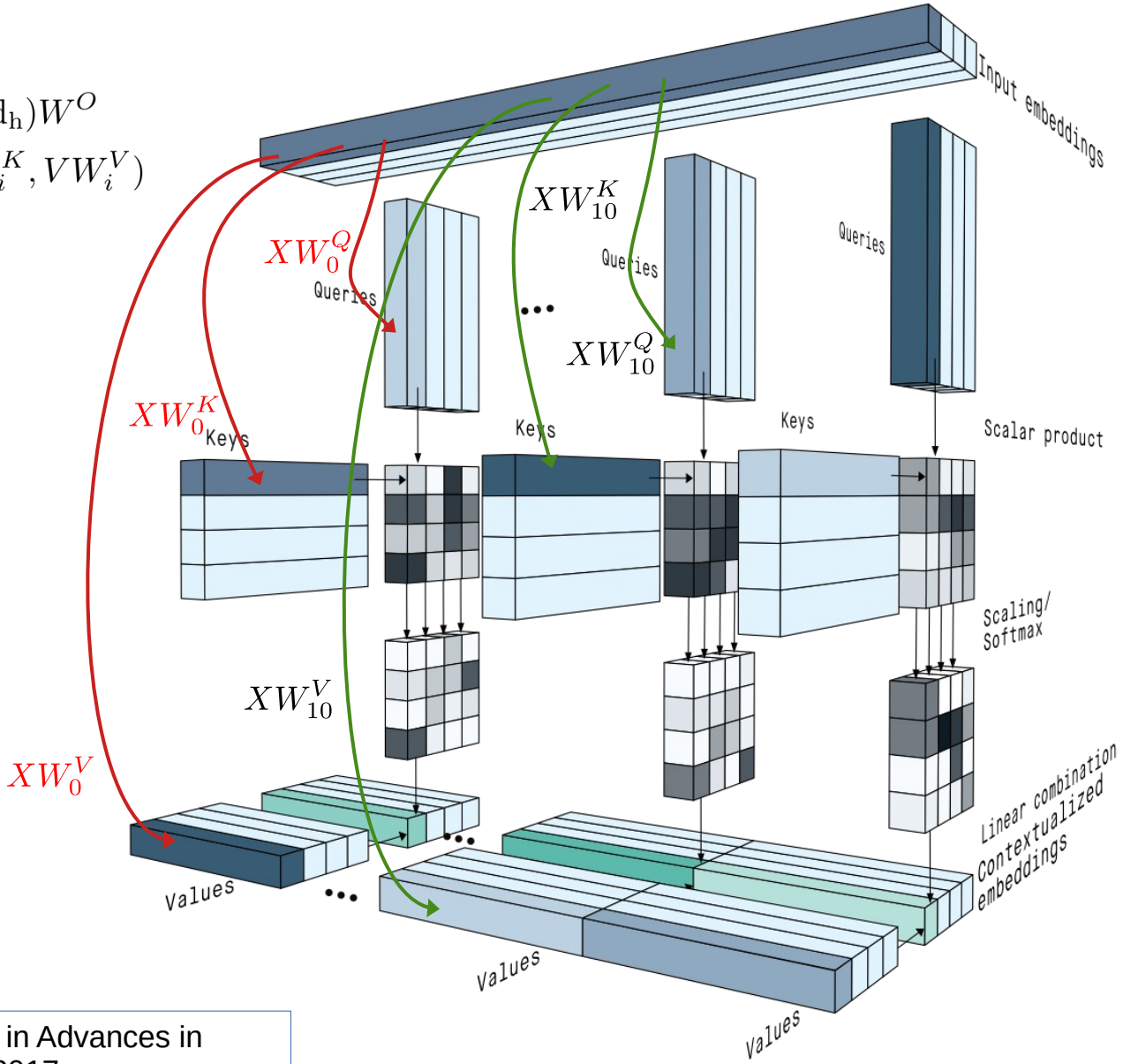
A. Vaswani et al., “Attention is All you Need,” in Advances in Neural Information Processing Systems 30, 2017.

https://towardsai.net/p/nlp/getting-meaning-from-text-self-attention-step-by-step-video

Transformer: Multi-Head Attention

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

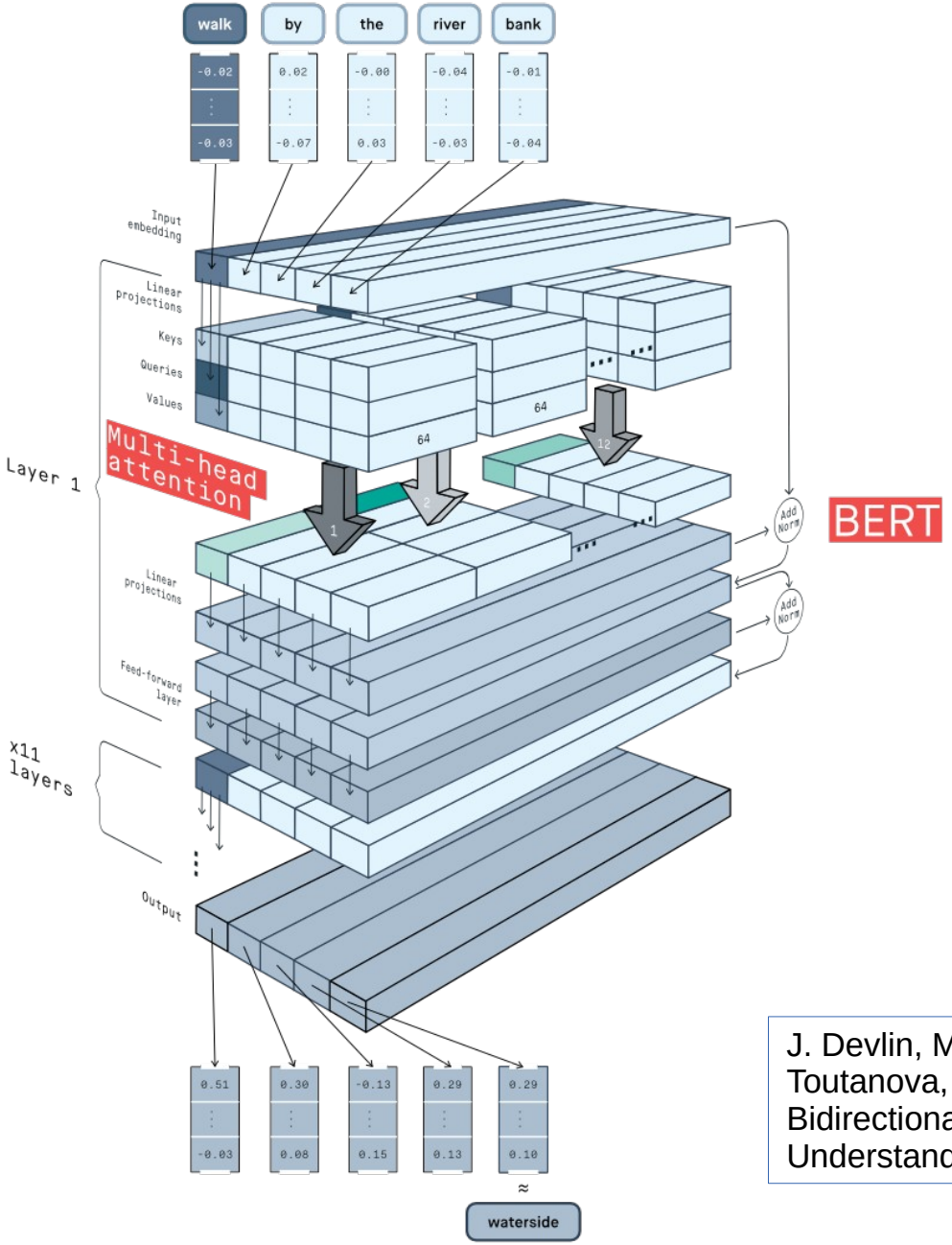




A. Vaswani et al., “Attention is All you Need,” in Advances in Neural Information Processing Systems 30, 2017.

https://towardsai.net/p/nlp/getting-meaning-from-text-self-attention-step-by-step-video

Full Transformer (Encoder Only)

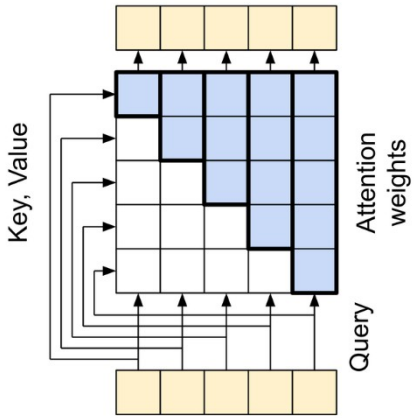


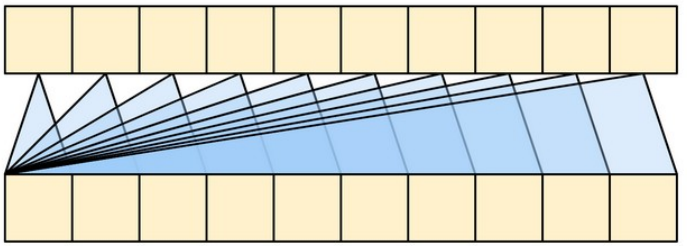
https://towardsai.net/p/nlp/getting-meaning-from-text-self-attention-step-by-step-video

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” 2019

Encoder/Decoder

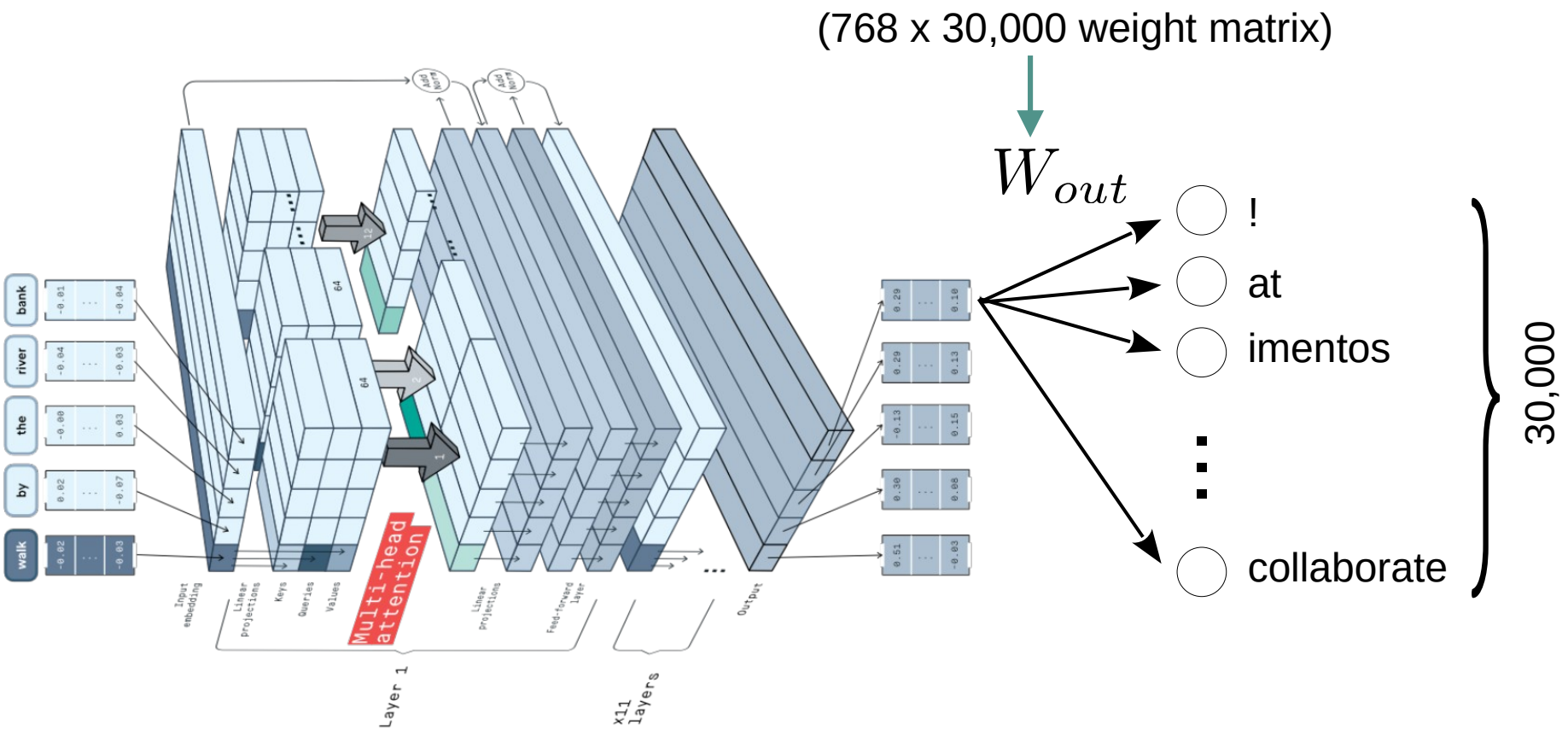
- Model above (BERT) is **Encoder-Only**:
  - Sequence in, embeddings out
- We can also have a **Decoder-Only** model (e.g. GPT)
  - Embedding in, sequence out
  - Intended for auto-regressive prediction: Each new token generated is fed back in before predicting the next token.
  - Training involves “masked self-attention”:



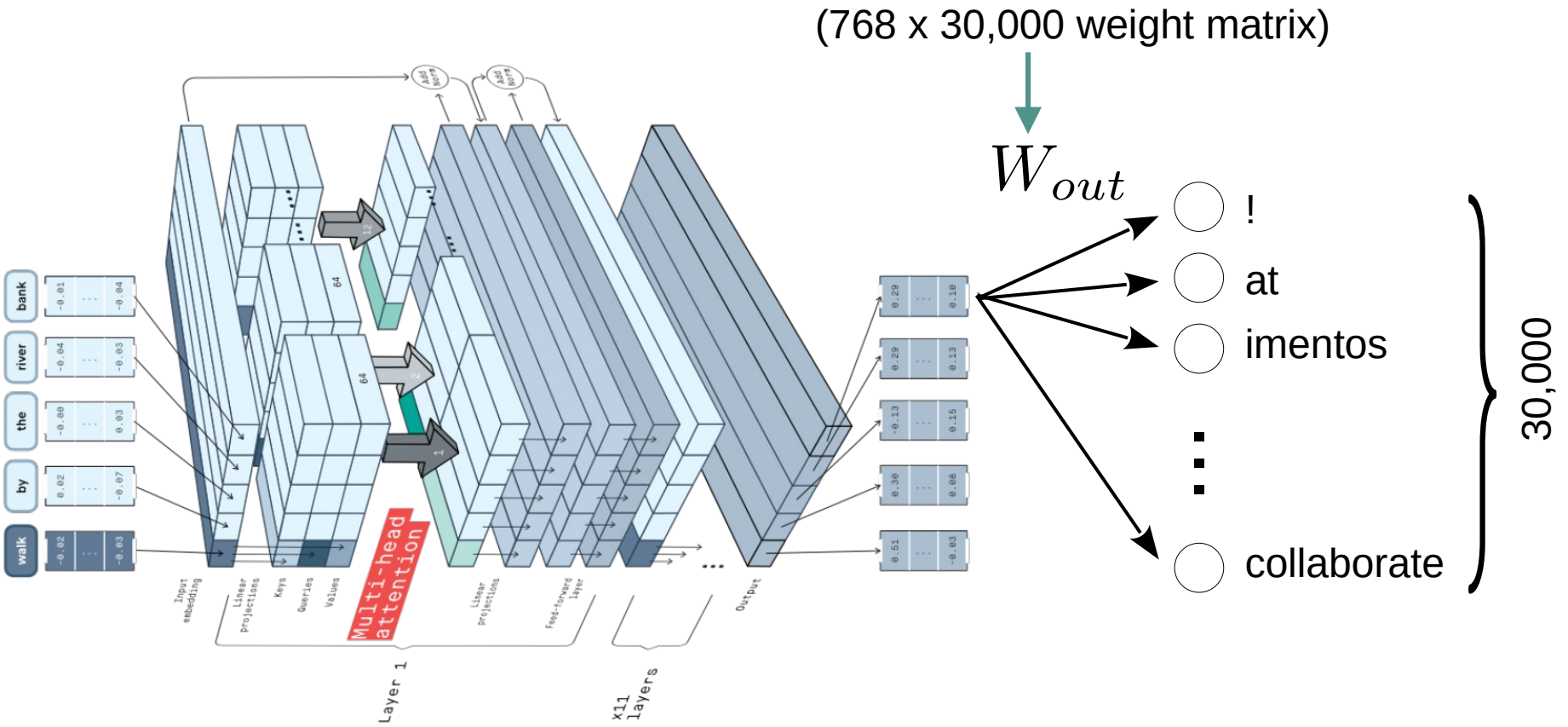


https://www.tensorflow.org/text/tutorials/transformer#the_causal_self_attention_layer

Full Transformer (GPT-Style Decoder Only)



Full Transformer (GPT-Style Decoder Only)



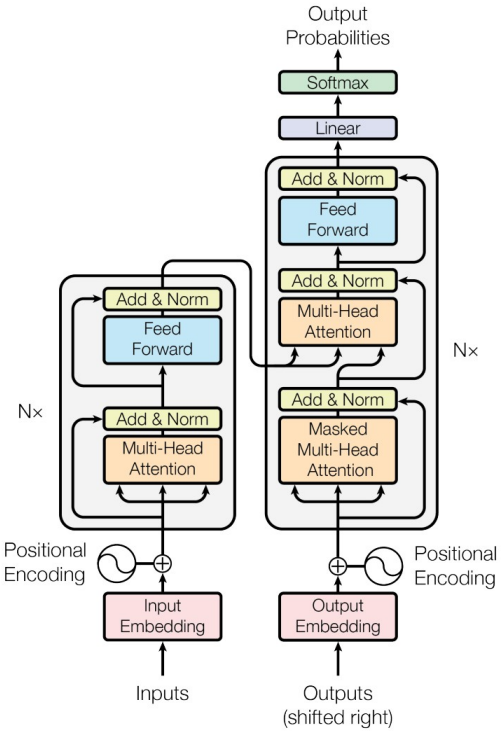
Train using gradient descent:

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$$

$$\mathcal{L}(\theta) = -\sum_{i=1}^{K} y_i \log p_\theta(y_i \mid x)$$

Encoder/Decoder

- Original Formulation was Encoder/Decoder
  - Sequence in, Different Type of Sequence out.
  - E.g. Machine Translation.



GPT Development

- Generative Pre-trained Transformers (GPT)
  - "Decoder only" model – Trained on a next-word prediction task. Future words are masked.
- GPT-1: Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*.
  - Train on a large text corpus, then fine-tune on specific tasks.

GPT Development

- Generative Pre-trained Transformers (GPT)
  - "Decoder only" model – Trained on a next-word prediction task. Future words are masked.
- GPT-1:

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*.

  - Train on a large text corpus, then fine-tune on specific tasks.
- GPT-2:

Radford, A. Wu., Child, R. Luan, D. Amodei, D. and Sutskever, I. (2019) *Language Models are Unsupervised Multitask Learners.*

  - No fine tuning. Instead, feed in several sample input/output pairs.

GPT Development

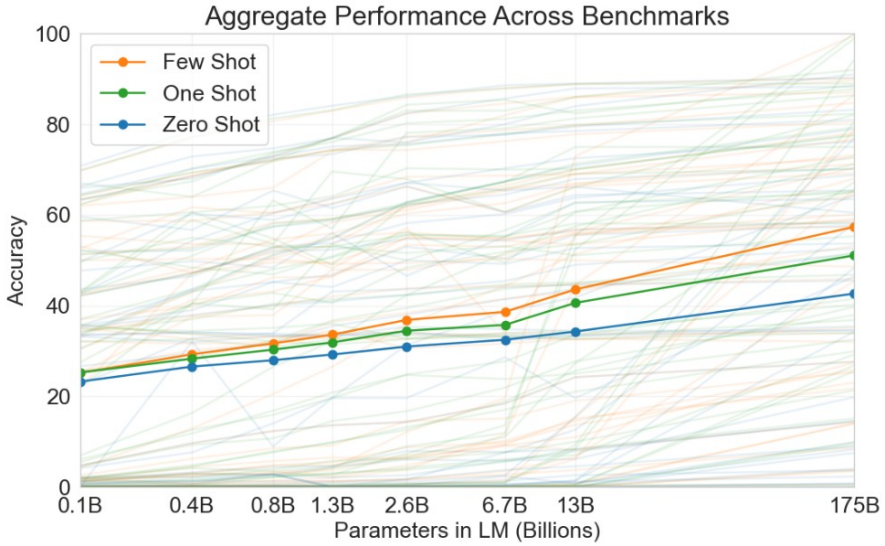
- Generative Pre-trained Transformers (GPT)
  - "Decoder only" model – Trained on a next-word prediction task. Future words are masked.
- GPT-1:

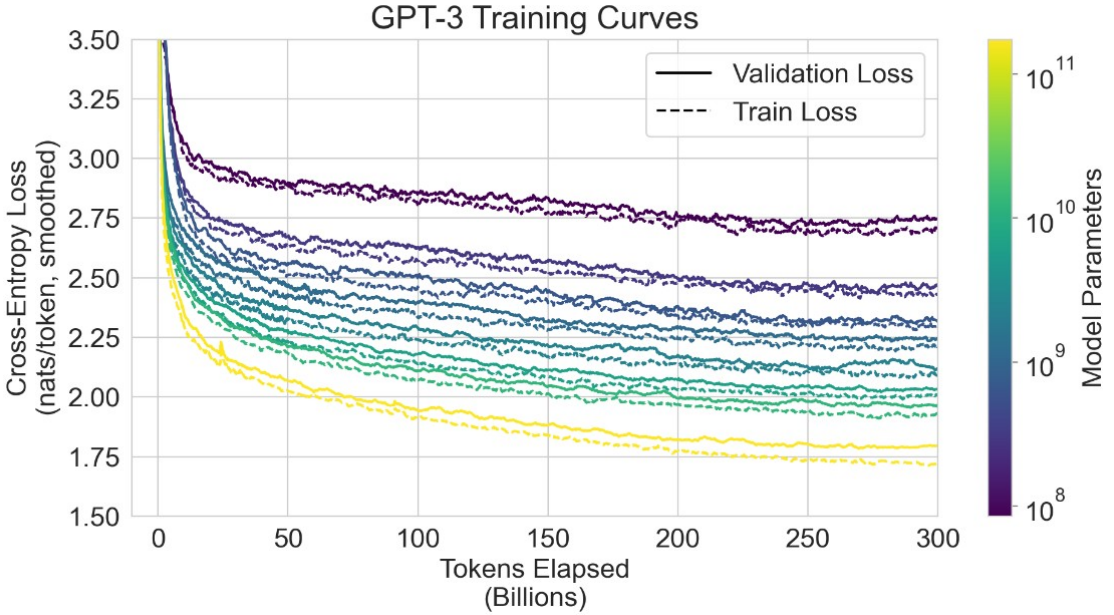
  Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*.
  - Train on a large text corpus, then fine-tune on specific tasks.
- GPT-2:

  Radford, A. Wu., Child, R. Luan, D. Amodei, D. and Sutskever, I. (2019) *Language Models are Unsupervised Multitask Learners.*
  - No fine tuning. Instead, feed in several sample input/output pairs.
- GPT-3:

  Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020)
  - "Zero-Shot Learning": Just ask for what you want.

GPT-3





Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020)

See also:

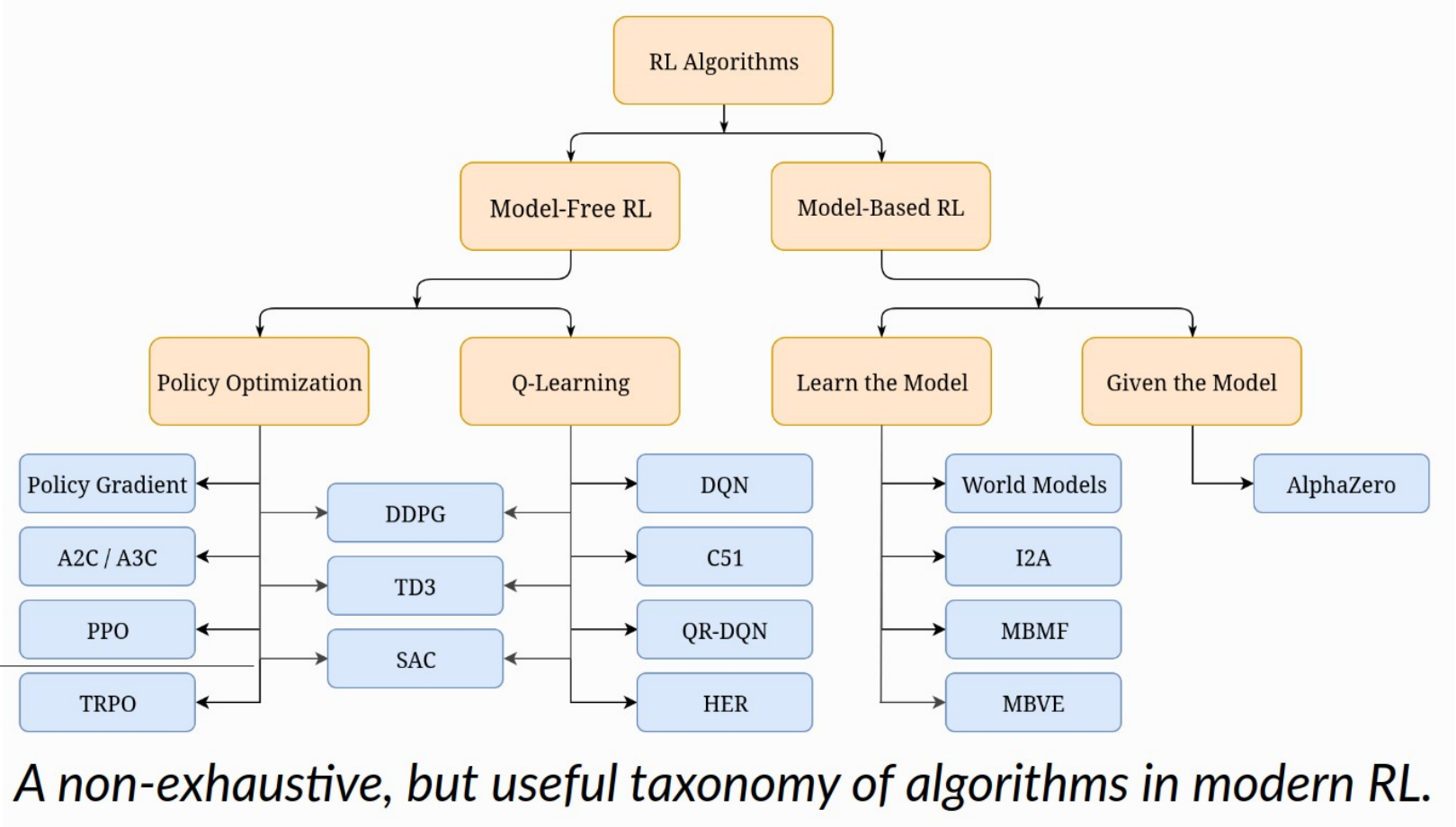
Kaplan, Jared, et al. "Scaling laws for neural language models." arXiv preprint arXiv:2001.08361 (2020).

GPT-4 (ChatGPT)

“This report focuses on the capabilities, limitations, and safety properties of GPT-4. GPT-4 is a Transformer-style model [39] pre-trained to predict the next token in a document, using both publicly available data (such as internet data) and data licensed from third-party providers. The model was then fine-tuned using Reinforcement Learning from Human Feedback (RLHF) [40]. Given both the competitive landscape and the safety implications of large-scale models like GPT-4, this report contains no further details about the architecture (including model size), hardware, training compute, dataset construction, training method, or similar.”

GPT-4 Technical Report, OpenAI (2023)

Reinforcement Learning



A non-exhaustive, but useful taxonomy of algorithms in modern RL.

https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html

Policy Gradients

- Goal is to maximize the expected return:

$$J(\pi_\theta) = \mathrm{E}_{\tau \sim \pi_\theta}[R(\tau)]$$

- We would like to use gradient ascent:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi_\theta)$$

Williams, Ronald J. "Simple statistical gradient-following algorithms for connectionist reinforcement learning." Machine learning 8.3 (1992): 229-256.

Policy Gradients

- Goal is to maximize the expected return:

$$J(\pi_\theta) = \mathrm{E}_{\tau \sim \pi_\theta}[R(\tau)]$$

- We would like to use gradient ascent:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi_\theta)$$

- This turns out to be the derivative:

$$\nabla_\theta J(\pi_\theta) = \mathrm{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau) \right]$$

- Which we can estimate from samples as:

$$\hat{g} = \frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, R(\tau)$$

- Turns out to be impractical :(

Williams, Ronald J. "Simple statistical gradient-following algorithms for connectionist reinforcement learning." Machine learning 8.3 (1992): 229-256.

Toward Practical Policy Optimization

- Trust Region Policy Iteration (TRPO)

Schulman, John, et al. "Trust region policy optimization." International conference on machine learning. PMLR, 2015.

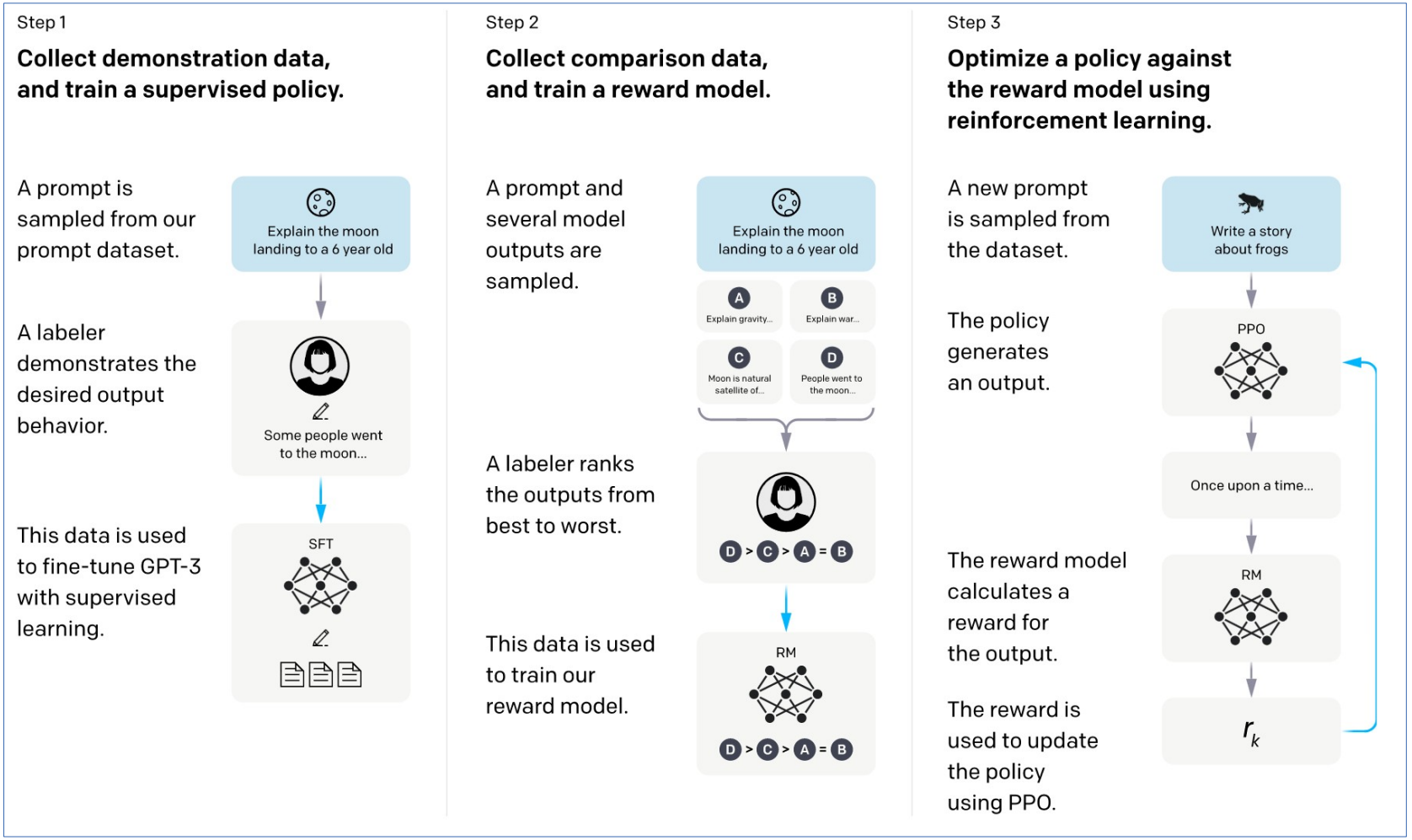
- Proximal Policy Iteration (PPO)
  - GPT3

Schulman, John, et al. "Proximal policy optimization algorithms." arXiv preprint arXiv:1707.06347 (2017).

- Group Relative Policy Optimization (GRPO)
  - DeepSeek-R1

Shao, Zhihong, et al. "Deepseekmath: Pushing the limits of mathematical reasoning in open language models." arXiv preprint arXiv:2402.03300 (2024).

Reinforcement Learning from Human Feedback



Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., & Lowe, R. (2022). *Training language models to follow instructions with human feedback*. Advances in Neural Information Processing Systems, 35.

Policy Gradients

- Goal is to maximize the expected return:

$$J(\pi_\theta) = \mathrm{E}_{\tau \sim \pi_\theta}[R(\tau)]$$

- We would like to use gradient ascent:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi_\theta)$$

Williams, Ronald J. "Simple statistical gradient-following algorithms for connectionist reinforcement learning." Machine learning 8.3 (1992): 229-256.

Policy Gradients

- Goal is to maximize the expected return:

$$J(\pi_\theta) = \mathrm{E}_{\tau \sim \pi_\theta}[R(\tau)]$$

- We would like to use gradient ascent:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi_\theta)$$

- This turns out to be the derivative:

$$\nabla_\theta J(\pi_\theta) = \mathrm{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau) \right]$$

- Which we can estimate from samples as:

$$\hat{g} = \frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, R(\tau)$$

- Turns out to be impractical :(

Williams, Ronald J. "Simple statistical gradient-following algorithms for connectionist reinforcement learning." Machine learning 8.3 (1992): 229-256.

Toward Practical Policy Optimization

- Trust Region Policy Iteration (TRPO)

Schulman, John, et al. "Trust region policy optimization." International conference on machine learning. PMLR, 2015.

- Proximal Policy Iteration (PPO)
  - GPT3

Schulman, John, et al. "Proximal policy optimization algorithms." arXiv preprint arXiv:1707.06347 (2017).

- Group Relative Policy Optimization (GRPO)
  - DeepSeek-R1

Shao, Zhihong, et al. "Deepseekmath: Pushing the limits of mathematical reasoning in open language models." arXiv preprint arXiv:2402.03300 (2024).

DeepSeek-R1

- Based on DeepSeek-V3
  - 671B parameter model
  - Trained on 14.8 trillion tokens

Liu, Aixin, et al. "Deepseek-v3 technical report." arXiv preprint arXiv:2412.19437 (2024).

- Deepseek-R1 created by using GRPO to train on a large collection of queries with known answers.

Supplementary Table 1: Description of RL Data and Tasks.

Data Type	# Prompts	Question Type	Output Type
Math	26K	Quantitative Reasoning	Number/Expression/Equation
Code	17K	Algorithm and Bug Fixing	Code Solution
STEM	22K	Multi-Choice	Option
Logic	15K	Choice/Quantitative Reasoning	Option/Number
General	66K	Helpfulness/Harmlessness	Ranked Responses

  - Reward based on correct final answer

Guo, Daya, et al. "Deepseek-r1 incentivizes reasoning in llms through reinforcement learning." Nature 645.8081 (2025): 633-638.

DeepSeek-R1-Zero Prompt

```
A conversation between User and Assistant. The User asks a question
and the Assistant solves it. The Assistant first thinks about the
reasoning process in the mind and then provides the User with the
answer. The reasoning process and answer are enclosed within
<think>...</think> and <answer>...</answer> tags, respectively, that
is, <think> reasoning process here </think> <answer> answer here
</answer>.

User: prompt.

Assistant:
```

DeepSeek-R1

- Deepseek Training:

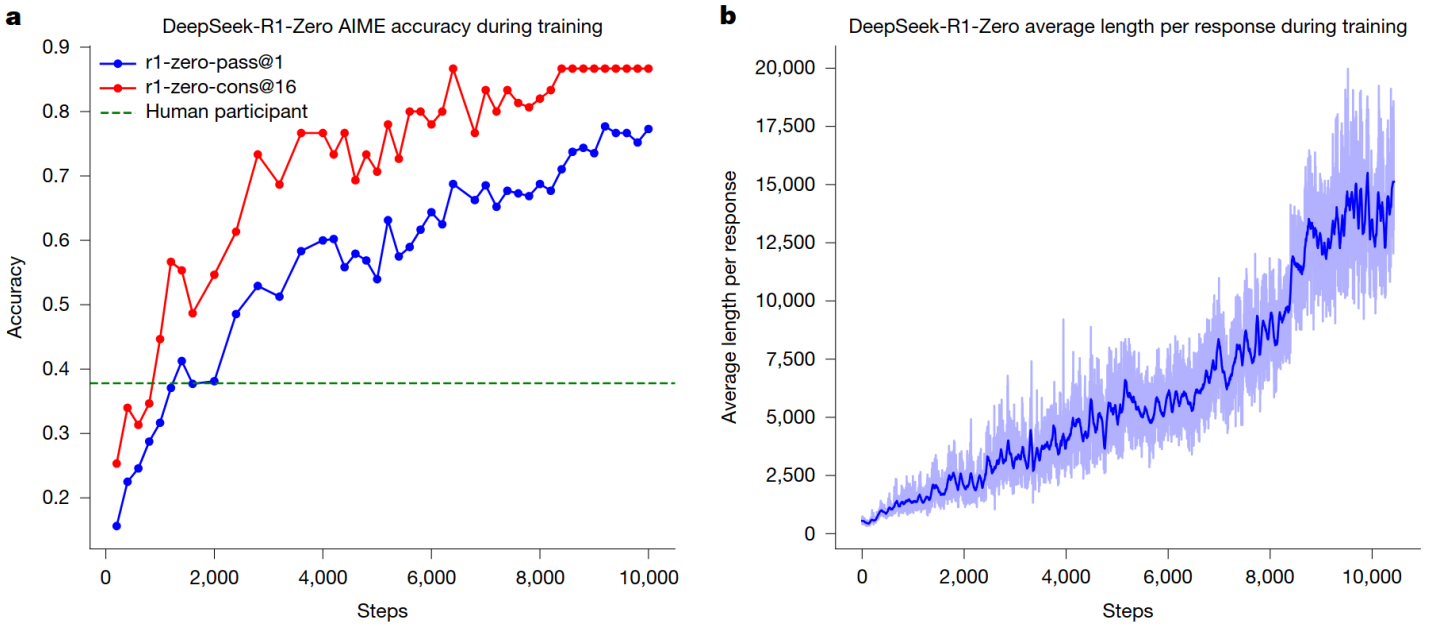


Fig. 1 | Accuracy and output length of DeepSeek-R1-Zero throughout the training process. a, AIME accuracy of DeepSeek-R1-Zero during training. AIME takes a mathematical problem as input and a number as output, illustrated in Extended Data Table 1. pass@1 and cons@16 are described in Supplementary Information, section 4.1. The baseline is the average score achieved by human participants in the AIME competition. **b**, The average response length of DeepSeek-R1-Zero on the training set during the RL process. DeepSeek-R1-Zero naturally learns to solve reasoning tasks with more thinking time. Note that a training step refers to a single policy update operation.

pass@1 probability of passing in one attempt
cons@16 probability of passing by making 16 attempts and taking the consensus

DeepSeek-R1 Full Training

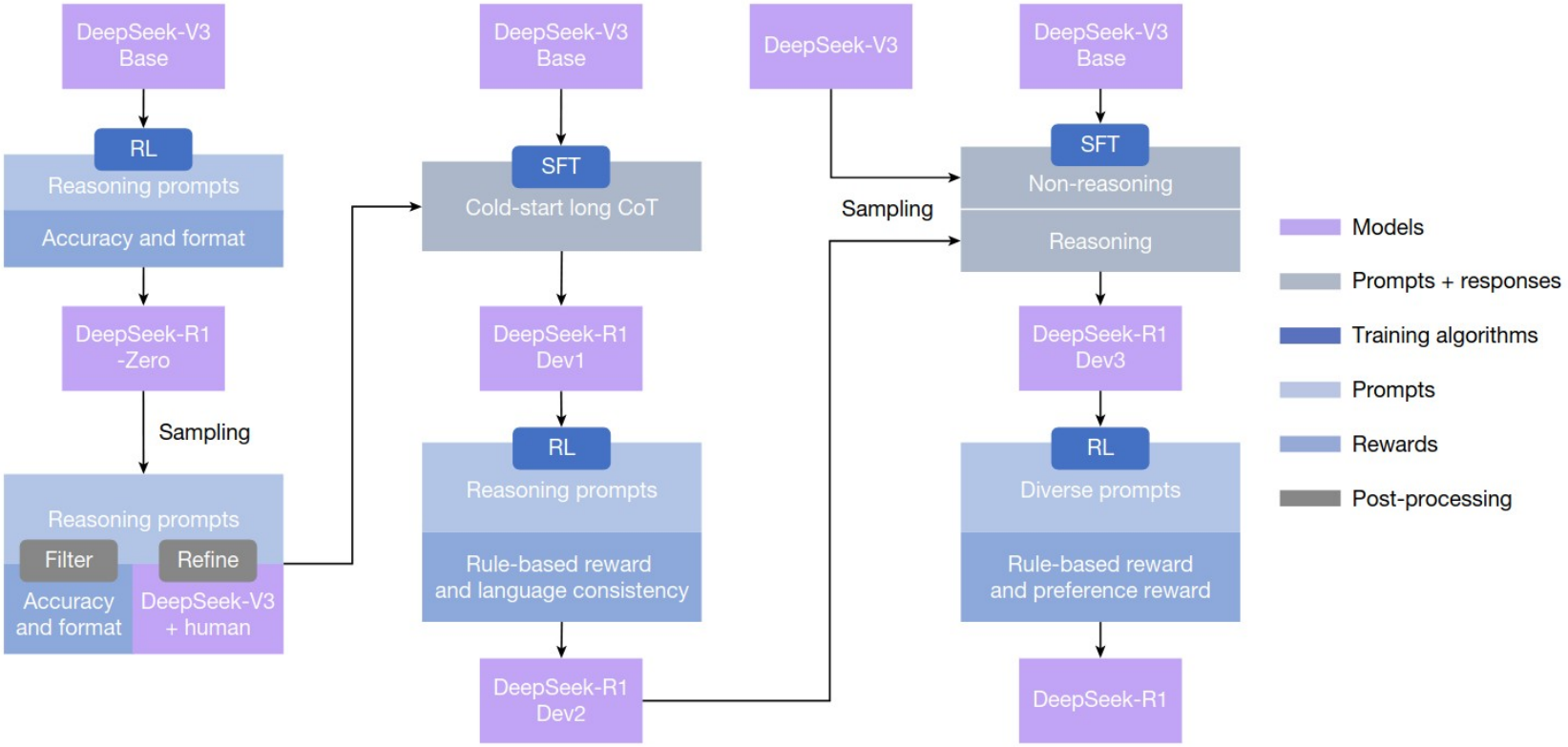


Fig. 2 | The multistage pipeline of DeepSeek-R1. A detailed background on DeepSeek-V3 Base and DeepSeek-V3 is provided in Supplementary Information, section 1.1. The models DeepSeek-R1 Dev1, Dev2 and Dev3 represent intermediate checkpoints in this pipeline.

Getting The Behavior We Want: More Tricks

- Prompt Design /Prompt Engineering
  - Examples:
    - "You are a 17th century pirate captain. Answer subsequent questions using an appropriate style and with only the knowledge that such an individual would have."
    - "Here is a copy of the Syllabus for CS 445. Please summarize the AI usage policy."
  - Downsides:
    - Computationally expensive to include large amounts of context/instructions as part of the model input
    - Amount of required information may exceed the context window.
- Chain of Thought Prompting – Asking the model to "show its work" leads to more correct answers.

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. "Chain-of-thought prompting elicits reasoning in large language models." Advances in neural information processing systems 35 (2022): 24824-24837.

- Fine Tuning
  - Take a pre-trained model and train it on a curated dataset that has the desired characteristics.
  - Downsides:
    - The models are HUGE. Managing a large collection of fine-tuned variants for each task is prohibitive.
    - Creating an appropriate dataset is time-consuming and difficult.
    - Anecdotally, fine tuning can be effective to get the model to generate the format that you want, but it works less well for adding knowledge.

Fine Tuning Tricks

- Prompt Tuning: Use gradient descent on a fine-tuning-style dataset to learn "soft-prompts" – Input vectors that are fed into the model before user input.

Lester, B., Al-Rfou, R., & Constant, N. (2021). The power of scale for parameter-efficient prompt tuning. arXiv preprint arXiv:2104.08691.

- Lora – Low Rank Adaptation:

learn efficiently despite a random projection to a smaller subspace. Inspired by this, we hypothesize the updates to the weights also have a low "intrinsic rank" during adaptation. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, we constrain its update by representing the latter with a low-rank decomposition $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$, and the rank $r \ll \min(d, k)$. During training, W_0 is frozen and does not receive gradient updates, while A and B contain trainable

Hu, Edward J., Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. "Lora: Low-rank adaptation of large language models." arXiv preprint arXiv:2106.09685 (2021).

Retrieval Augmented Generation

- RAG – Retrieval Augmented Generation
  - Technique for giving a model access to data that was not included in training data
    - Content is broken into chunks, and each chunk is encoded into a vector representation
    - Query is encoded into a vector representation and matched against content database
    - Retrieved text is provided to the model as context for answering the user's question.

Lewis, Patrick, et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks." Advances in Neural Information Processing Systems 33 (2020): 9459-9474.

Quantization

- What if I want to run one of these models on a laptop, or a phone?
  - We might train a model using 32-bit floats
  - Models like Llama3 (Meta's open-source model family) are distributed as 16-bit floats.
  - "Quantization" involves reducing this to fewer bits. 8, 4, or fewer.
  - Reduces compute time and memory requirements.

Jacob, Benoit, et al. "Quantization and training of neural networks for efficient integer-arithmetic-only inference." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

(Among many others)

Positional Embeddings

- The relative order of words impacts the meaning of the text: position information is added to the word embeddings
- Original transformer used a vector of sine and cosine functions at different frequencies
- More recently, Rotary Position Embeddings have become popular

Su, Jianlin, et al. "Roformer: Enhanced transformer with rotary position embedding." Neurocomputing 568 (2024): 127063.