Chat-enabled LLMs InstructGPT and Llama 2

Venelin Kovatchev

Lecturer in Computer Science

v.o.kovatchev@bham.ac.uk

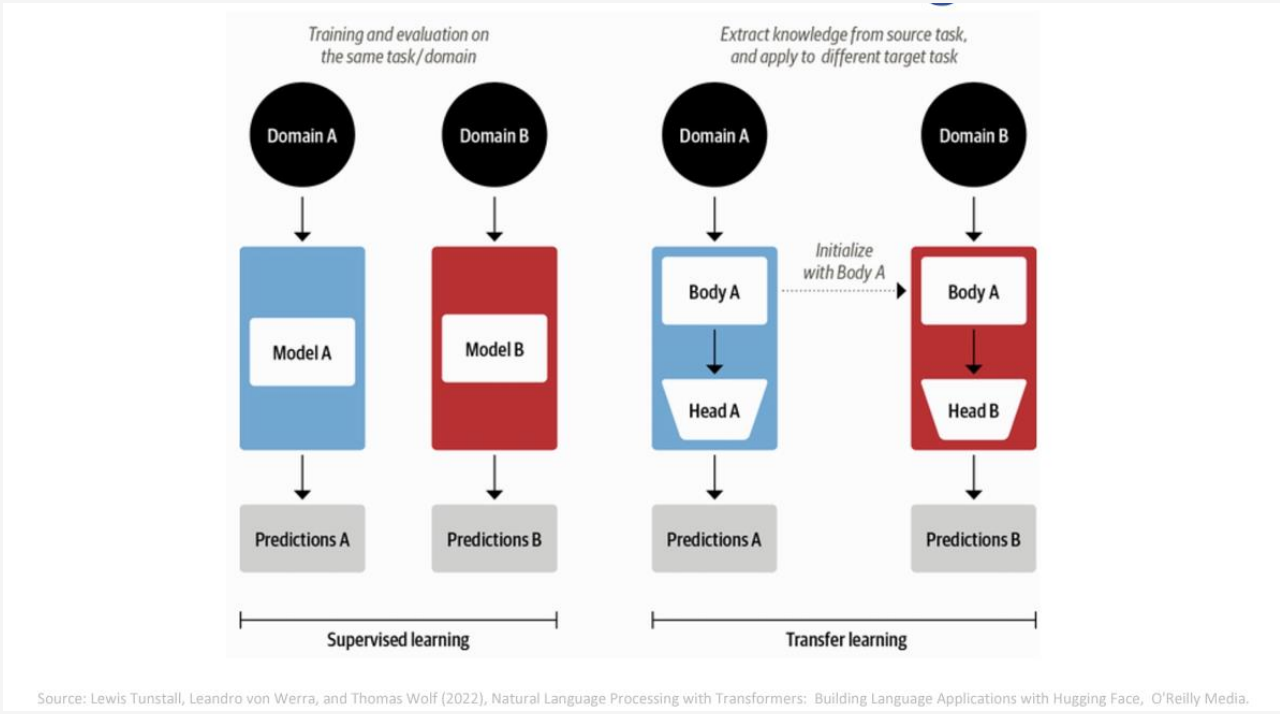
Outline

- Quick recap
  - In-context learning (part 2)
- Chat-enabled LLMs
  - InstructGPT
  - LLama2

Quick recap:
Supervised Learning, Transfer Learning
In-context learning

Supervised learning vs Transfer learning

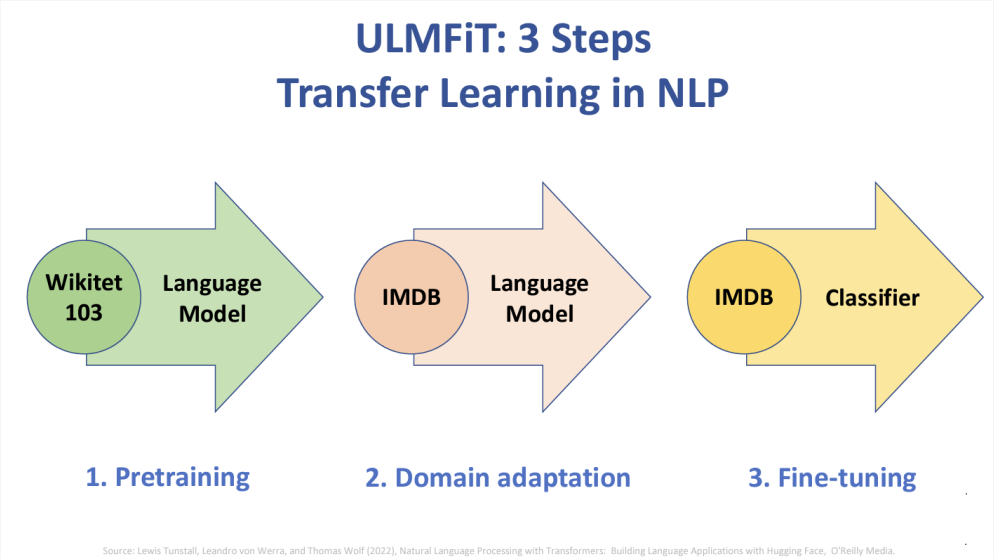
- Supervised learning
  - Full training of the model
- Transfer learning
  - Reusing part of the model
  - Finetune the head



Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.

ULMFiT (Howard and Ruder et al., 2018)

- "Universal Language Model Finetuning"
- Training a language model for "inductive transfer learning"
  - Model trained on a source task (language modeling)
  - Finetuned with limited data on target task
- Language modeling – the analogy of ImageNet
- Base model - BiLSTM



ULMFiT Pipeline

- Three step training
- Most parameters remain intact

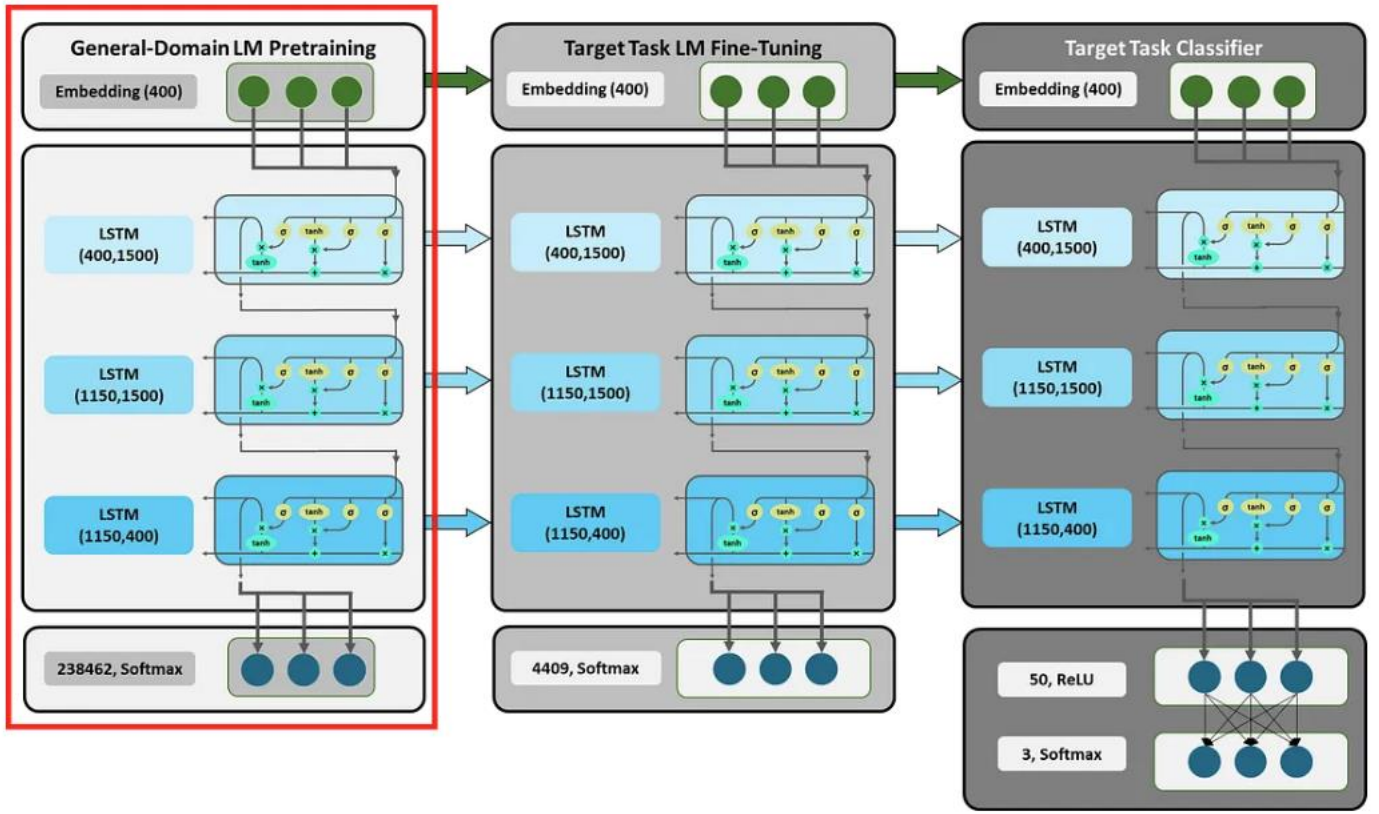


Figure 5: Block Diagram of ULMFiT in Text Analysis — Image from by HU-Berlin from GitHub

Transfer learning and Transformers

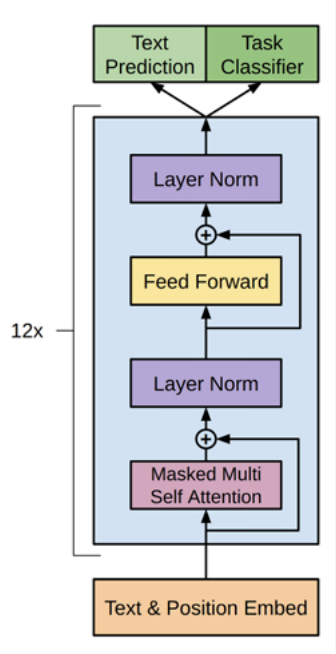
- Transformers quickly adopted the idea of transfer learning

- Three types of transformers
  - Encoder models: BERT, ROBERTA, DistilBERT
  - Decoder models: GPT, GPT2
  - Encoder-decoder models: BART, T5

- Pop-quiz: what is the difference between the three types of transformers?

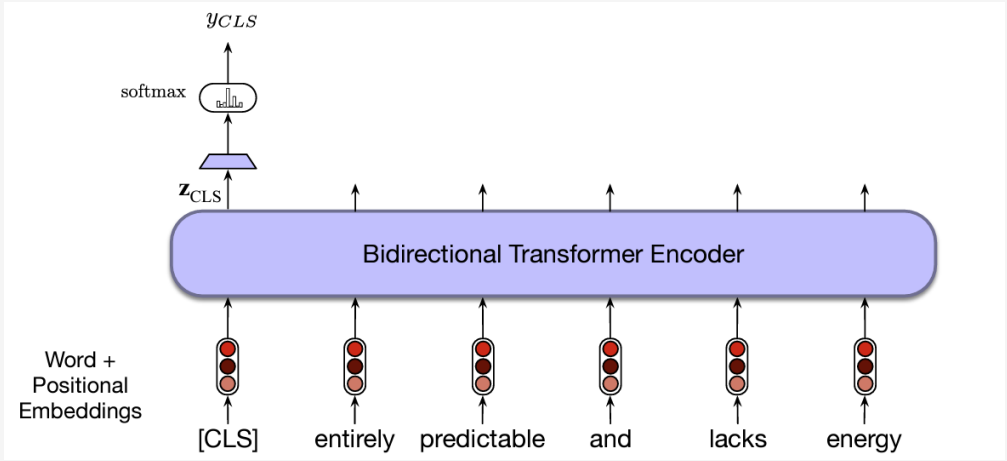
The decoder transformer: GPT

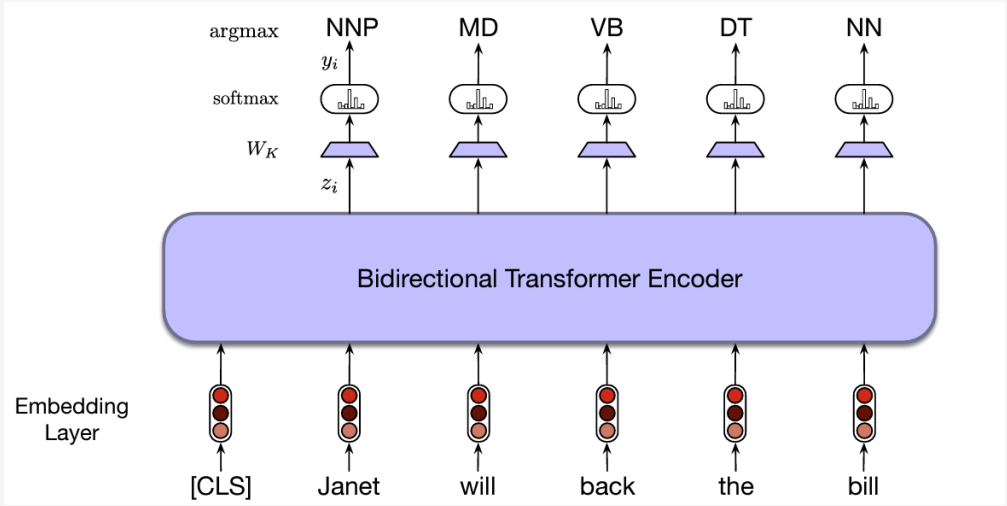
- GPT1 is a decoder (a.k.a. autoregressive) transformer
  - Causal attention (!) + a standard transformer block
  - Trained on neural language modeling
  - Transfer learning capabilities
- Intuition:
  - Generative pre-training
  - Discriminative finetuning
  - Two separate loss functions $L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$



The encoder transformer: BERT

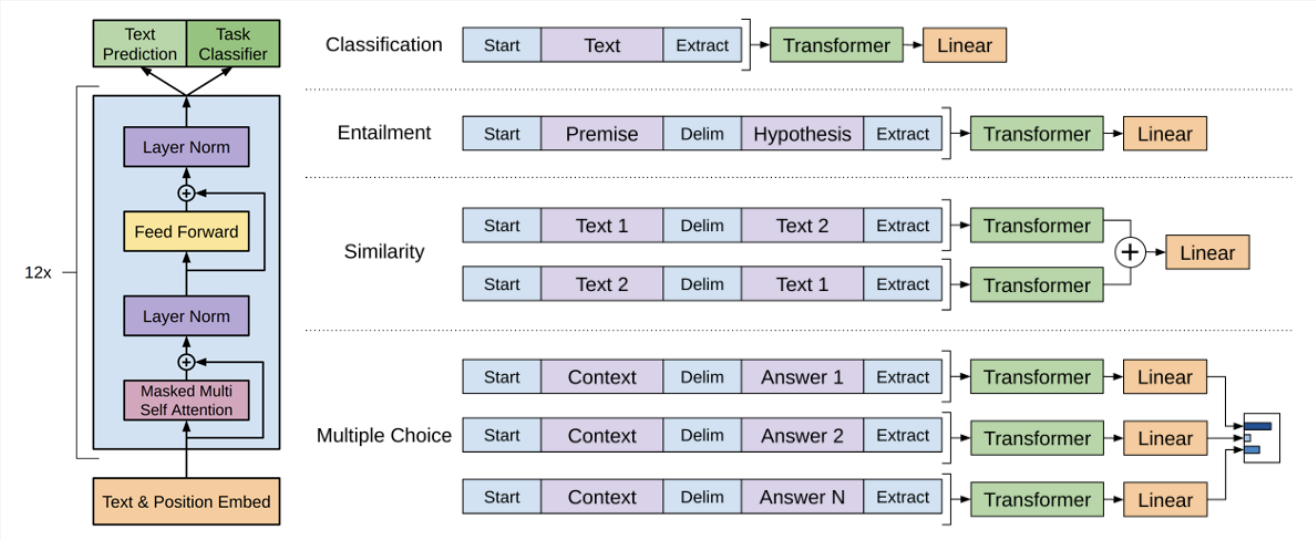
- BERT is the original encoder-only transformer
  - Bidirectional attention + a standard transformer block
  - Trained on two tasks:
    - Masked Language Modeling
    - Next Sentence Prediction
    - Pop quiz: what did each of those do?
  - Can be finetuned on a variety of tasks
    - Generally, performs better than GPT





Task specific input transformations

- Task reformulating
- Using special tokens (sep, start/end)
- Comparing separate “streams”
- Task design is a non-trivial task
  - Task formulation; Data format; Metrics and Evaluation



GPT3

- A major shift in NLP paradigm
  - Arguably the largest shift since moving from pipeline to end-to-end models
- Introduced few-shot and zero-shot learning
  - Teaching a model to perform a task without changing the weights (!)
- In-context learning and prompt engineering

Few-shot and Zero-shot learning

- The problem
  - Getting training data is complex and expensive (there are far more tasks than datasets)
  - Overfitting (to spurious correlations)
  - Humans don’t need large training data for all tasks
- The goal
  - One model that can perform multiple tasks
  - In-context learning
  - AGI?

In-context learning vs supervised/transfer learning

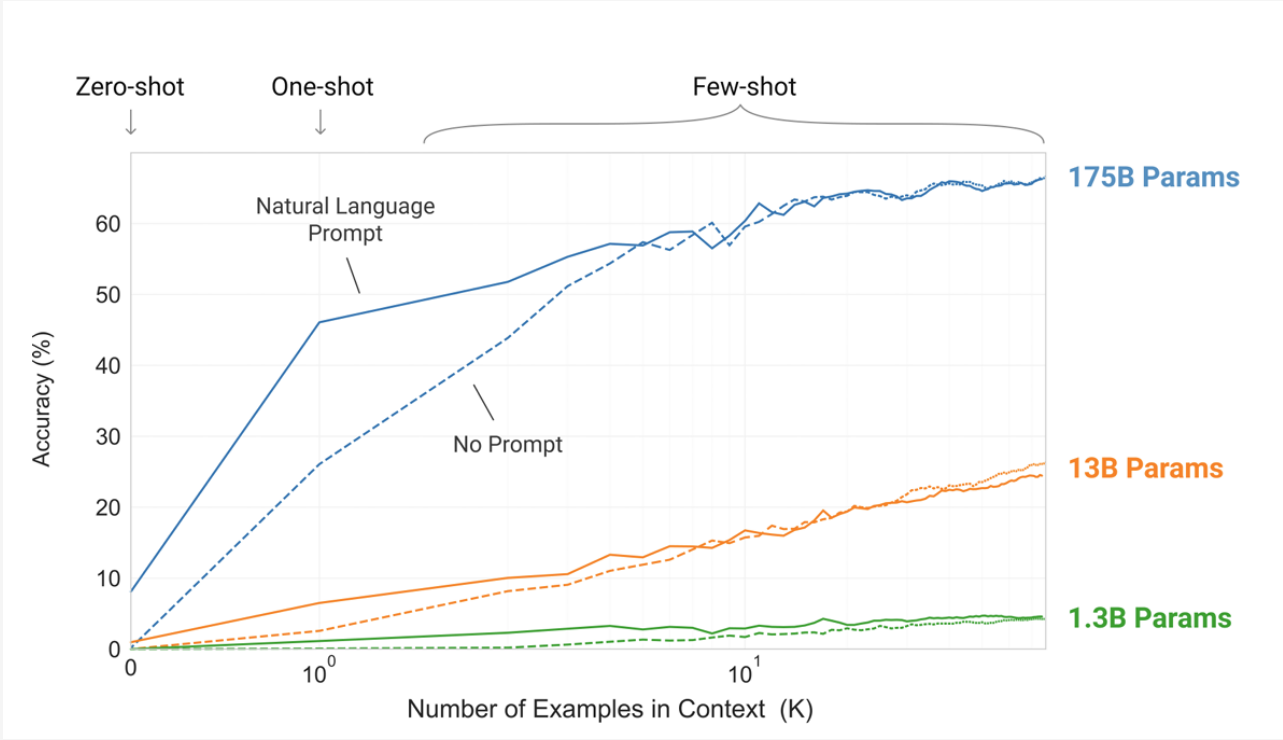
- Using the input to specify the task
- Consider the following inputs to a transformer model:
  - “I like this movie, it’s the best in the Avengers series!”
  - “I bike to work every day. <SEP> I drive to work every day.”
- What is the task? What is the output?
  - The task is what you train the model to do
  - The first sentence can be an input to a NER model
  - The second sentence can be an NLI task or a similarity task

In context learning

- Now consider the following inputs:
  - “What is the sentiment of the following text: I like this movie, it’s the best in the Avengers series!”
  - “Do those sentences contradict each other: I bike to work every day. <SEP> I drive to work every day.”
- Is anything missing in that formulation?
- How do we achieve in-context learning?
  - GPT3 paper argues that scale and emerging properties are the answer
  - Increasing model size from 17B to 175B

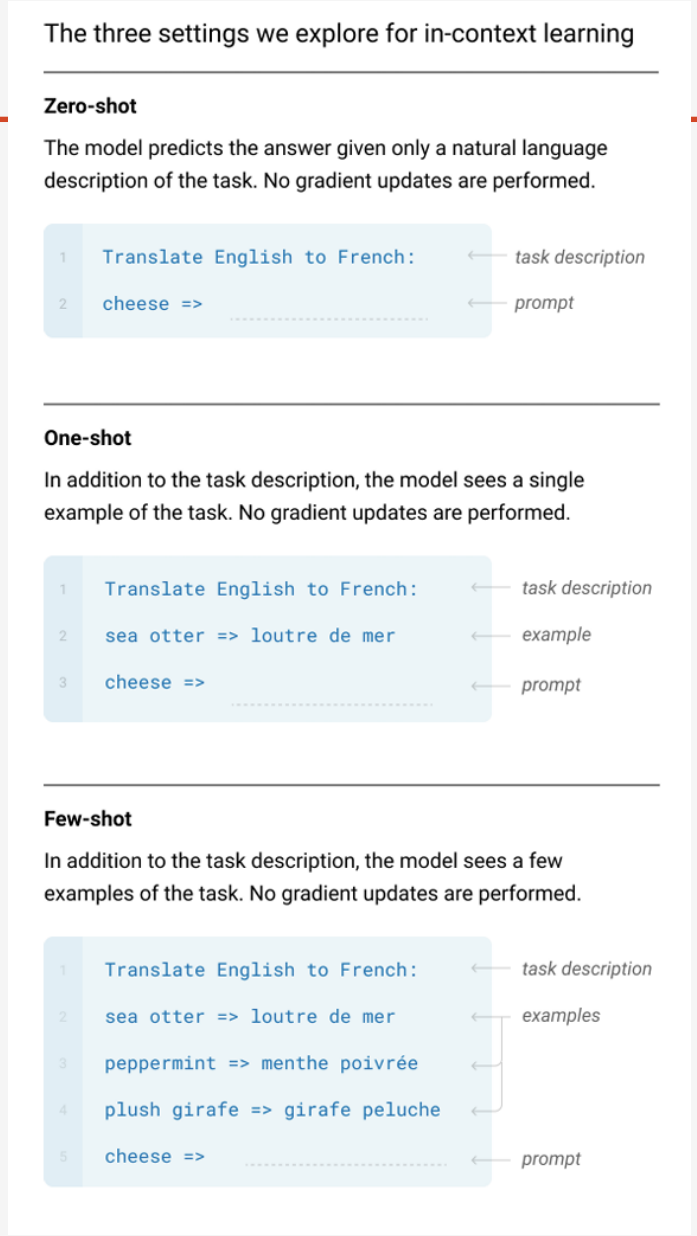
In-context learning and model size

- Two key factors
  - Model size
  - Number of examples
- Model size -> “emerging properties”



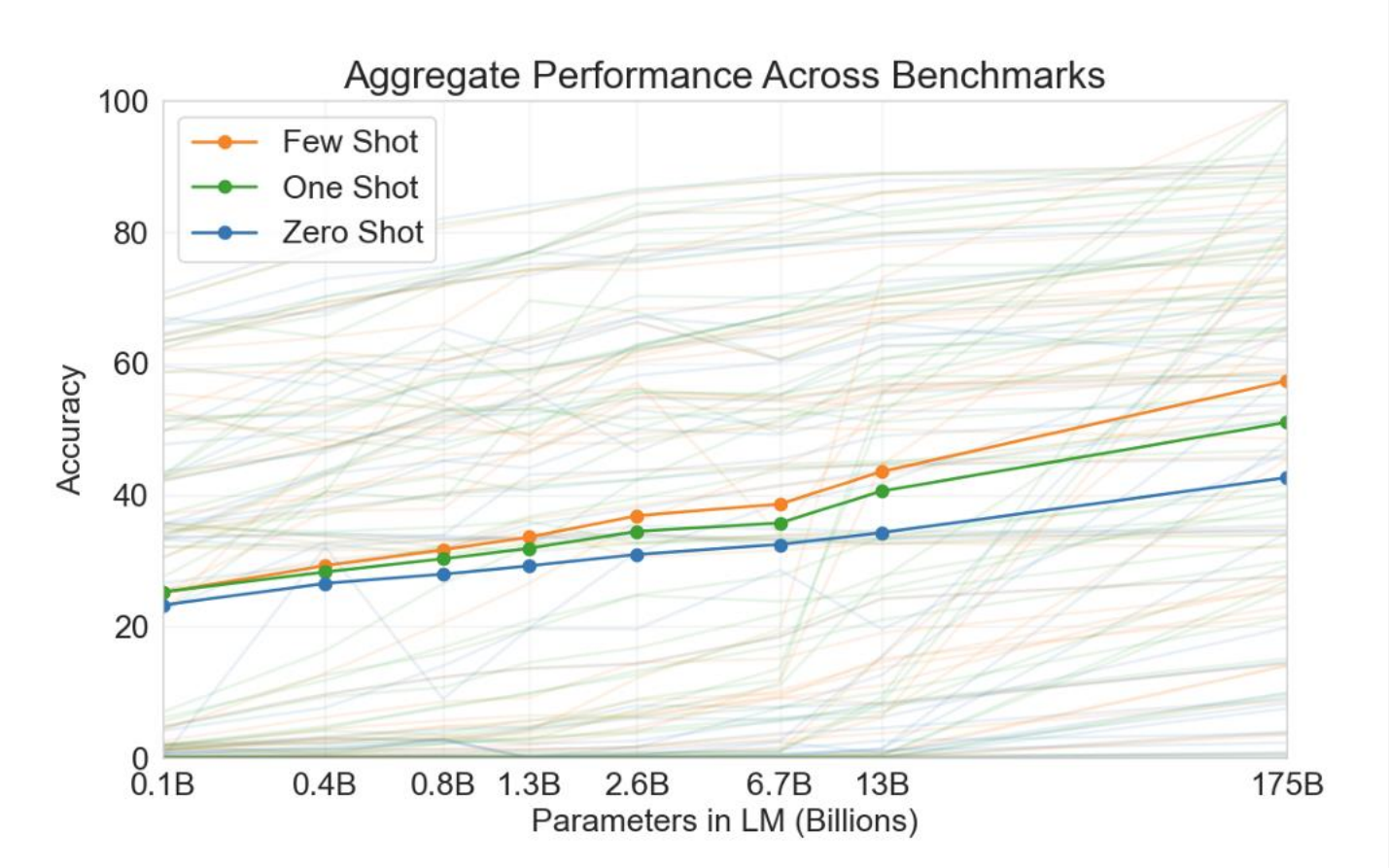
Zero- One- and Few-shot learning

- Three different experimental conditions
- No gradient update or finetuning
- The only difference – number of examples



In-context learning and model size

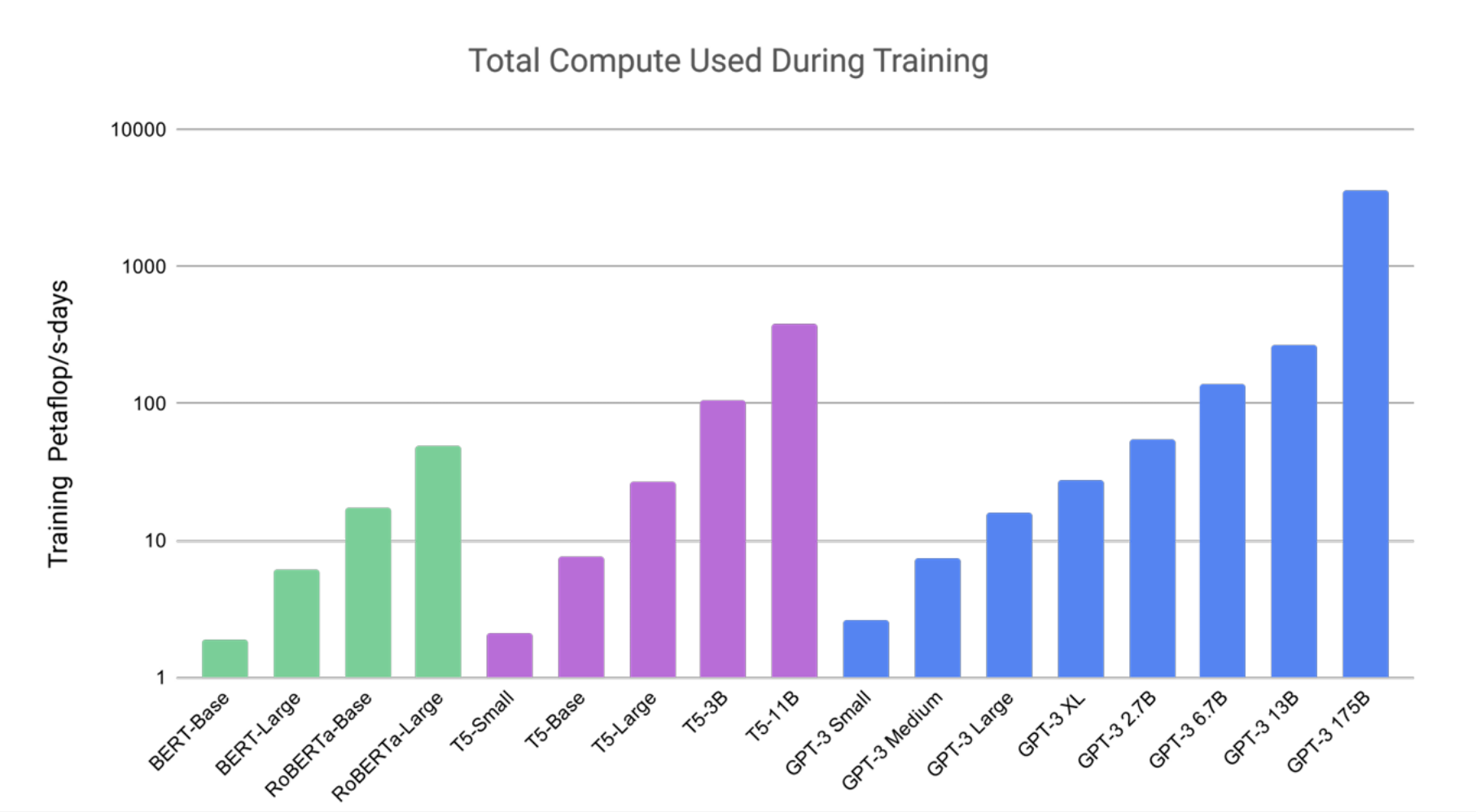
- Same model
- Different sizes
- Different number of examples



Mode architecture

- Same as GPT2, scaled to 175B params
  - 96 layers
  - 96 attention heads
  - 128 d per head -> 12k d for the hidden state
- Causal attention, trained on LM task
  - Trained on Common Crawl (1 trillion words) + data filtering
  - Additional curated high-quality datasets

Compute used for training



GPT3 and current LLMs

- Almost all of current chat-enabled LLMs are based off the concepts in GPT3

- Newer models are performing better
  - More parameters; larger and better datasets
  - Additional training: supervised and RLHF finetuning
  - Human-driven and machine-driven prompt engineering

- GPT3 can, in principle, do anything that modern LLMs can
  - The difference is in the implementation, not in the core concepts (!)

Chat-enabled LLMs
InstructGPT, Llama 2

Bringing LLMs to end users

- GPT3 opens the possibility of multi-purpose LLMs
  - No need for finetuning or updating parameters
- The Turing test and NLP/AI goal of conversations and collaborations
- InstructGPT and ChatGPT
  - Improving GPT ability to converse
  - Improved experience for end-users

InstructGPT – training models to follow instructions

- Scale is not everything
  - Hallucinations
  - Toxicity
  - Lack of helpfulness
- Improve the training (and evaluation) procedures
- “Align” models with their users

InstructGPT – high level idea

- Determining human preferences and goals
  - Preference of responses given a label
  - Usefulness and Safety
- "Align" the model output with human preferences
  - Additional model training
  - Focused improvements

The LM training objective

- Language modeling is not “following instructions”
- Language modeling does not take (individual) preferences
- Every training sequence is equally important
- Preference in output (in language modeling) depends on
  - The observed frequency in training data
  - The sampling strategy
- Few- and Zero-shot learning are an “emerging” side effect, not an intentionally defined goal

InstructGPT – the goals of “alignment”

- Three key objectives
  - Helpful
  - Honest
  - Harmless

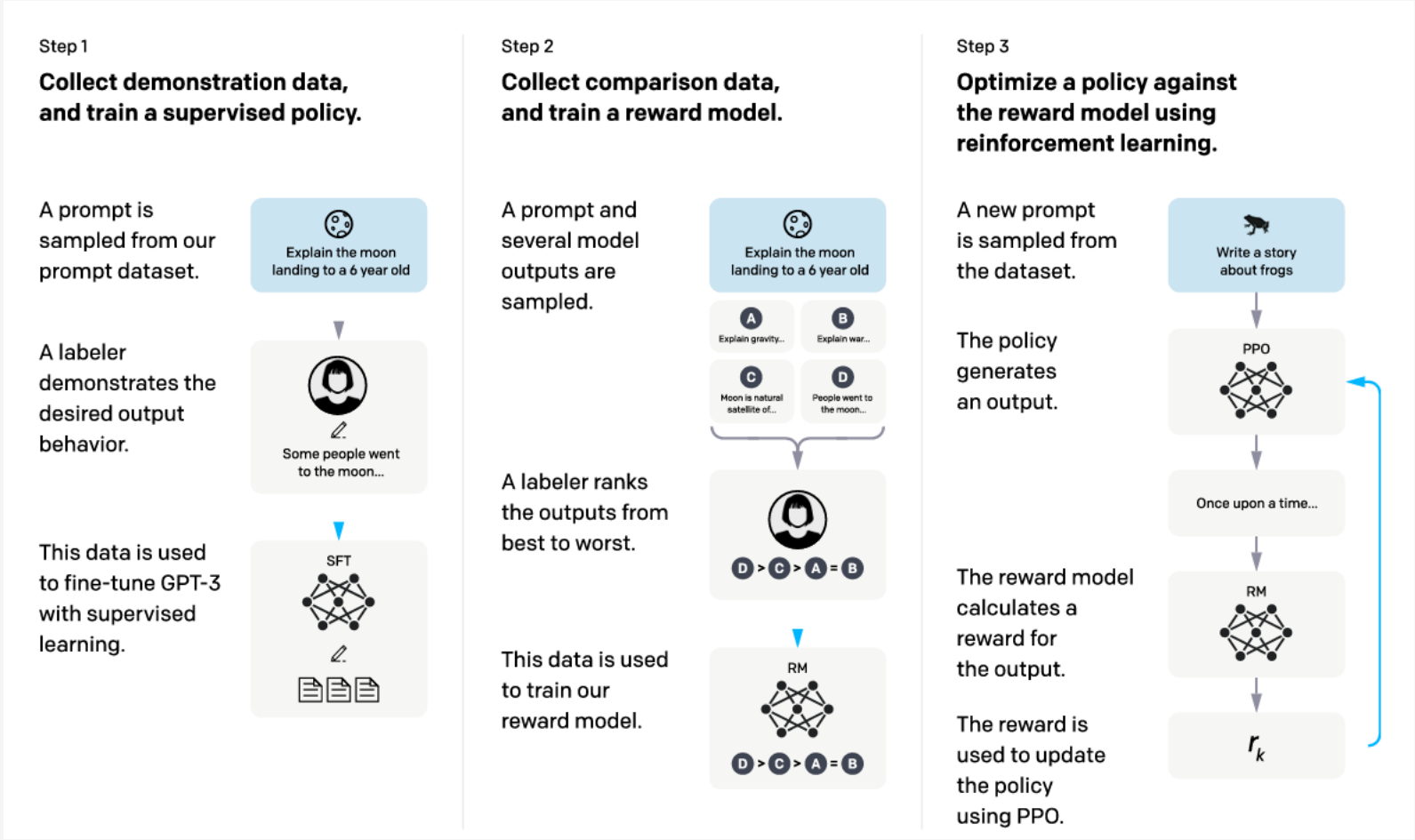
- Later, reduced to two:
  - Helpful
  - Harmless

InstructGPT – steps in training

- Three (four) step training process
  - General pre-training on LM (reusing GPT3)
  - Supervised LM finetuning on human preferences
  - RL LM finetuning on human preferences
    - Intermediate step on training a reward model

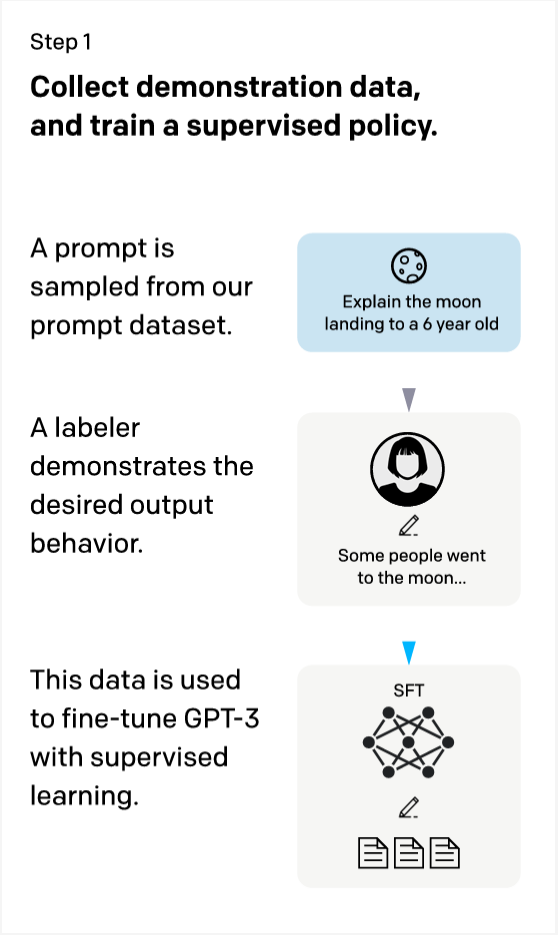
- How does this compare to other techniques that we have seen before?

The training process of InstructGPT



Collecting demonstrations and supervised training

- Input data
  - Real user prompts (first version has human-written prompts)
  - Human written responses
  - Prioritizing helpfulness over truthfulness and safety for training (!)
  - Reverting the priorities for testing
- Training process
  - Language modeling objective
  - Using only training data from the “SFT” dataset



Is SFT enough to get good models?

- SFT improves the performance of LLMs for chat

- There are some (empirical) limitations
  - Availability of human labeled data
  - Overfitting to training data and diminishing returns
  - Limitations of coverage and limited improvement w.r.t bias

Using RLHF to further improve alignment

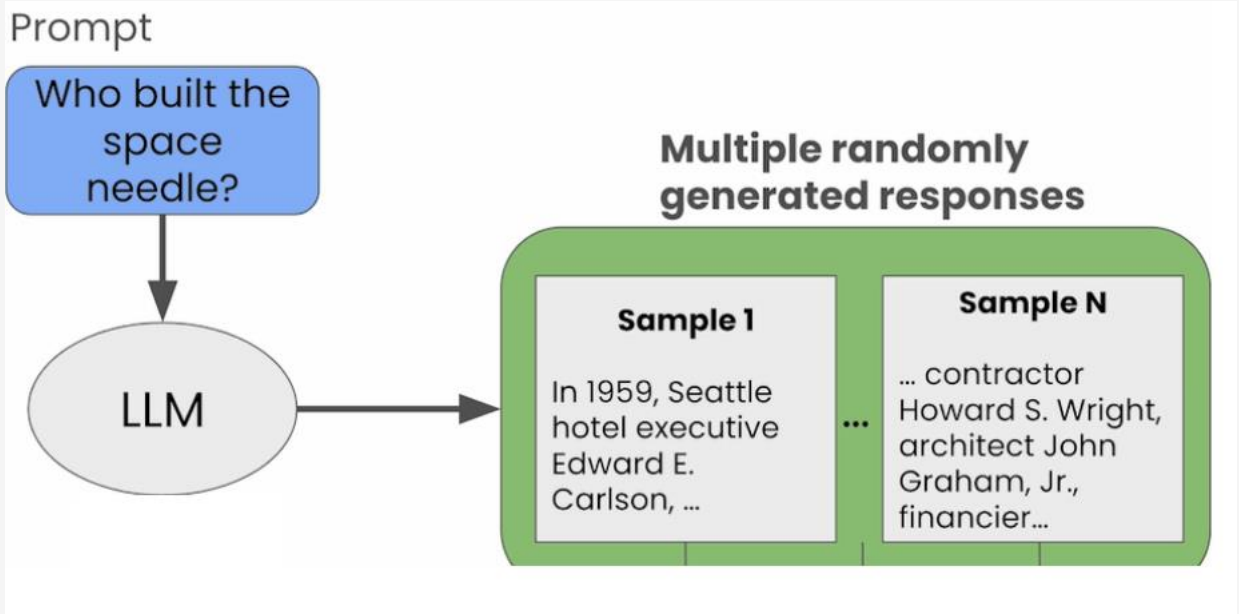
- InstructGPT proposes using RL to continue alignment after SFT

- Three step process
  - Obtaining data about preferences
  - Training a reward model
  - Aligning using RLHF

Collecting comparisons and training a reward model

- Labelers are given one prompt and multiple responses

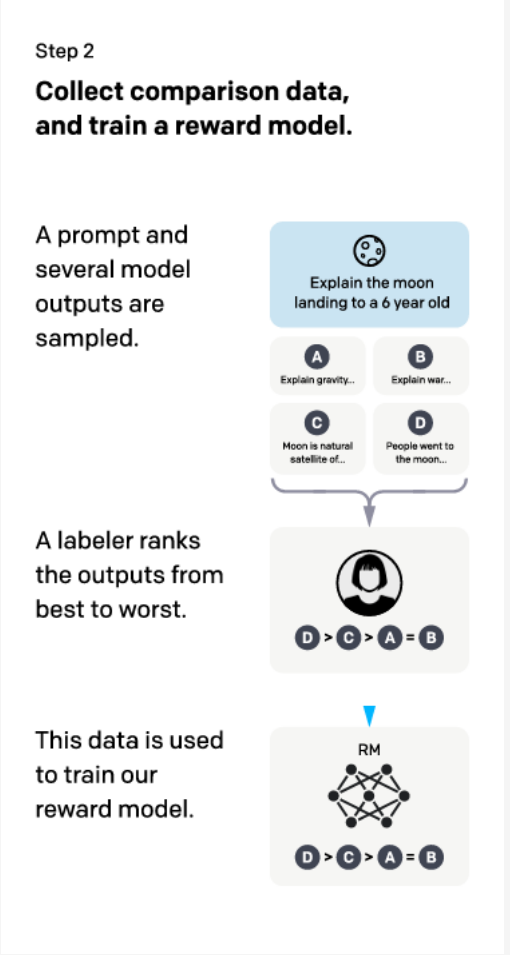
- How many responses do you show?
  - Two responses
  - Ranking of multiple responses
  - Ranking + binarization
- Multiple annotators + inter-annotator agreement



Source: Medium.com

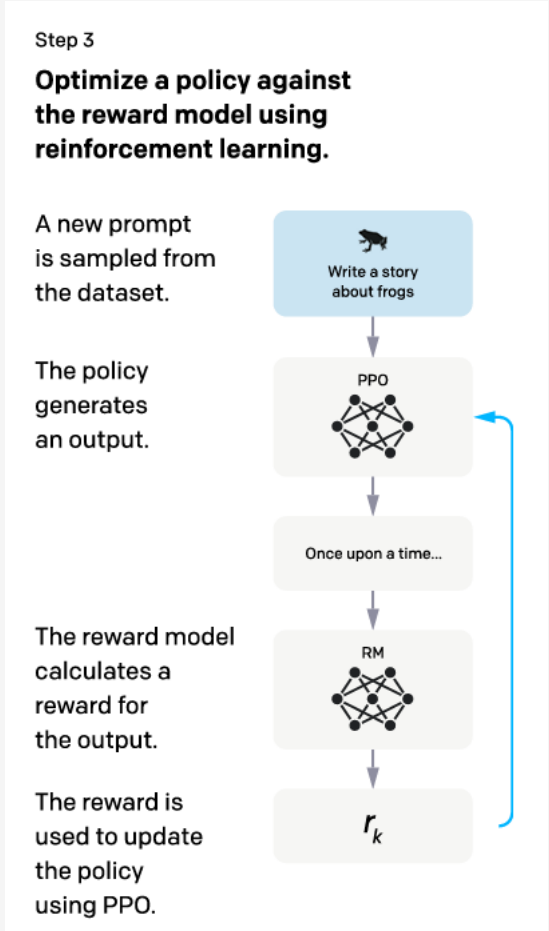
Training a reward model

- Supervised finetuning of an SFT trained LLM
  - A smaller version is sufficient (e.g. 6B params)
- Task specific head (regression):
  - Provide a scalar score for an input (prompt + response)
- Training:
  - Get the score of two (or more) comparisons
  - Calculate the diff. between pref. and non-pref. labels and convert to log prob.
  - Update all comparisons for a single prompt to avoid overfitting
  - $$\text{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x,y_w,y_l)\sim D}\left[\log\left(\sigma\left(r_\theta\left(x,y_w\right) - r_\theta\left(x,y_l\right)\right)\right)\right]$$



Reinforcement Learning for Policy Optimization

- The final model parameters are optimized via RL
- The model generates a response to a prompt
- The reward model assigns a reward
- The policy is updated given the reward and the probability of the response



The objective function of RLHF

- The reward function takes into account
  - The reward for the prompt – r(x,y)
  - The probability of the response, given the RL policy
  - The probability of the response, given a SFT model

$$\text{objective}\,(\phi) = E_{(x,y)\sim D_{\pi_\phi^{\text{RL}}}}\left[r_\theta(x,y) - \beta \log\left(\pi_\phi^{\text{RL}}(y \mid x)/\pi^{\text{SFT}}(y \mid x)\right)\right] + \gamma E_{x\sim D_{\text{pretrain}}}\left[\log(\pi_\phi^{\text{RL}}(x))\right]$$

Evaluating chat-LLMs

- Goals of aligned models
  - Helpful – does it follow instructions and provide adequate response
  - Honest – is the response truthful
  - Harmless – is the response “safe to use” for e.g. customer assistant; does the response denigrates a protected class; does the response contain sexual or violent content

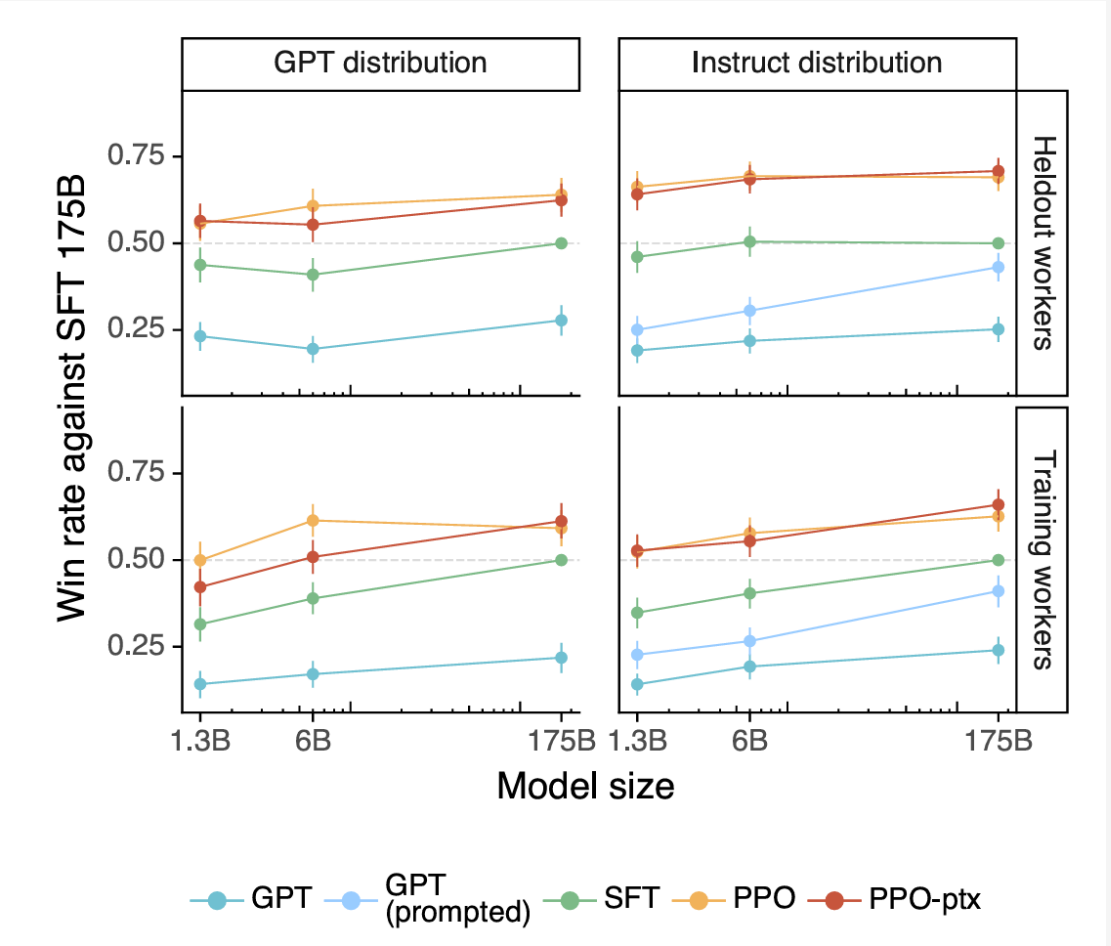
- How would you evaluate the models on each of these categories?

Evaluation of chat-LLMs

- User evaluation
  - Given a pair of responses from different model, which model is preferred?
  - Is there any other way the experiment can be performed?
- Benchmark evaluation
  - Test performance of LLMs on specific benchmarks
  - Non-LLM-specific benchmarks (zero-shot on e.g. question answering)
  - LLM-specific benchmarks on truthfulness, bias, tixicity

Evaluating user preferences

- User preference rating against 125B SFT
- Left – prompts coming from GPT
- Right – prompts coming from InstructGPT
- Top – test set results
- Bottom – train set results



Benchmark Evaluation

- Models are compared on existing benchmarks:
  - TruthfulQA
  - RealToxicityPrompts
- InstructGPT shows minor improvement on benchmarks
  - Human preference does not (always) correspond to better prompts!

Designing datasets for finetuning

- Most datasets used for finetuning are private
  - Competitive edge
  - Copyrighted content (?)

- After the impact of ChatGPT, community started creating datasets
  - Usefulness
  - Safety

Llama 2

- The first open (large) foundational model
- Same training procedure as InstructGPT
- Using both public and in-house data
  - Binary preference for the reward model

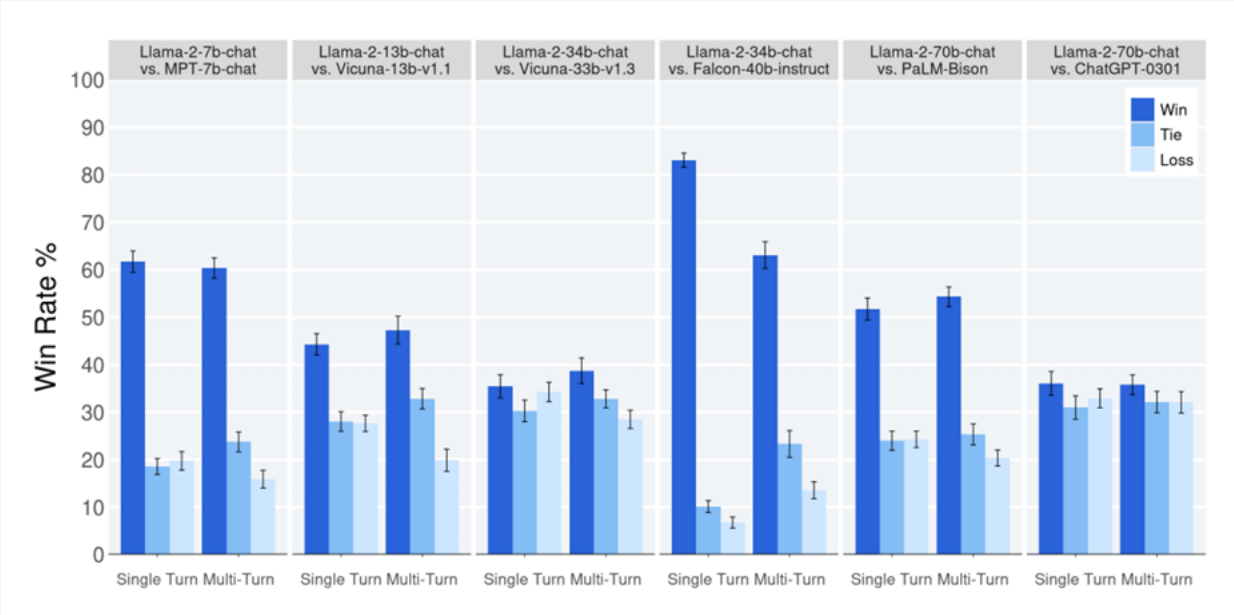
Data usage when training Llama 2

- RLHF data (only)
- 3 mil dialogs
- 2 bil tokens
- 50% proprietary data

Dataset	Num. of Comparisons	Avg. # Turns per Dialogue	Avg. # Tokens per Example	Avg. # Tokens in Prompt	Avg. # Tokens in Response
Anthropic Helpful	122,387	3.0	251.5	17.7	88.4
Anthropic Harmless	43,966	3.0	152.5	15.7	46.4
OpenAI Summarize	176,625	1.0	371.1	336.0	35.1
OpenAI WebGPT	13,333	1.0	237.2	48.3	188.9
StackExchange	1,038,480	1.0	440.2	200.1	240.2
Stanford SHP	74,882	1.0	338.3	199.5	138.8
Synthetic GPT-J	33,139	1.0	123.3	13.0	110.3
Meta (Safety & Helpfulness)	1,418,091	3.9	798.5	31.4	234.1
Total	2,919,326	1.6	595.7	108.2	216.9

Performance against other chat models

- On human evaluation – better than the alternatives
- Hard to validate without releasing the data



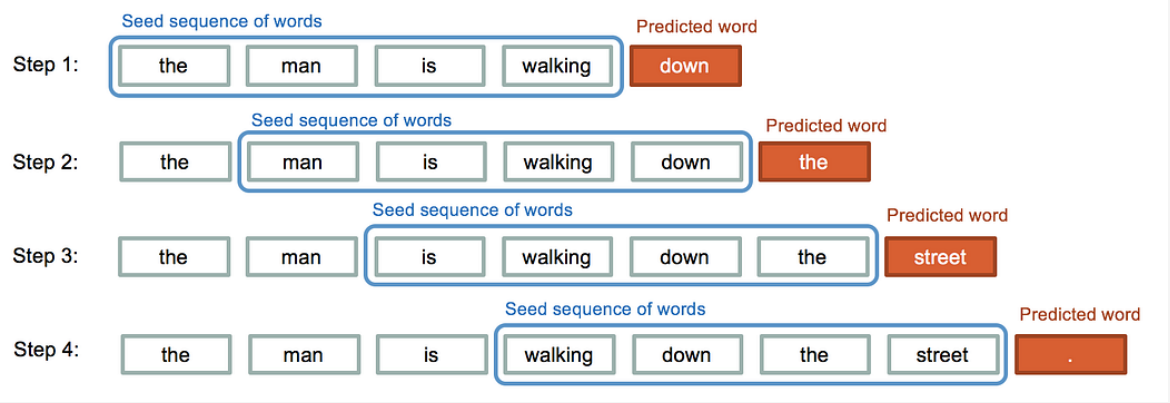
Sampling from LLMs

Improving LLM performance via sampling

- LLMs in production
  - Conditional probability distribution over tokens
  - A sampling strategy over the probability distribution
  - External filtering applied to input/output
- How can we improve them?

Decoding from a (neural) language model

- Iterative process
- Autoregressive process
- Computationally expensive
- Subject to multiple objectives and constraints



source: medium.com

Objectives of decoding

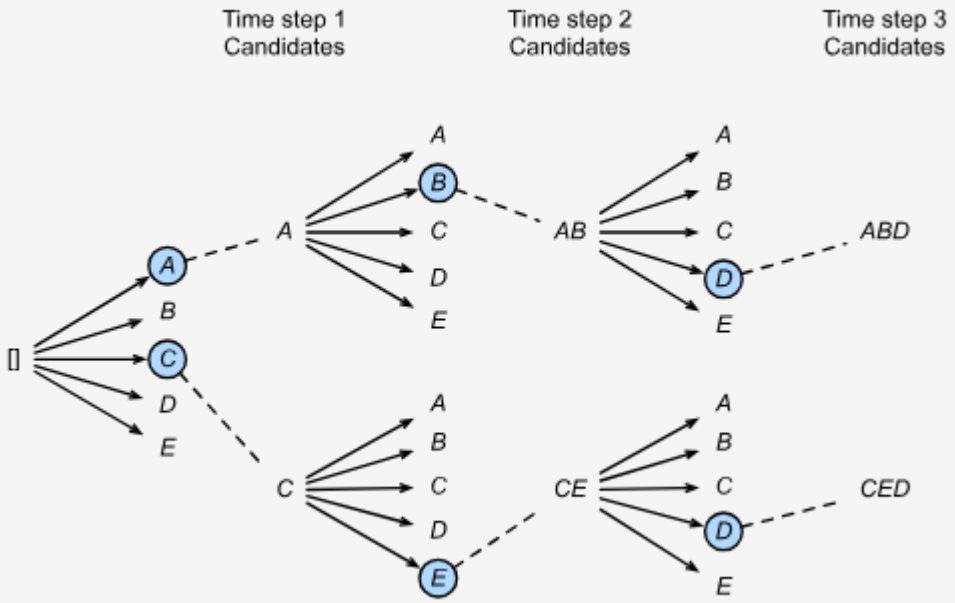
- Maximize the probability of the sequence
- Maximize diversity of generated output
- LLM-chat specific objectives
  - Helpful
  - Honest
  - Harmless

Simple decoding strategies

- The greedy search
  - Pick the most probable token, always
- Sampling from the distribution
  - Pick a word at random, weighted by the probability
- Apply filtering/reweighting:
  - Top-K, Top-P, Temperature

Beam search decoding

- One of the most popular approaches
  - At every step, pick the k most probable options
  - Get the top-k probabilities from each branch
  - Reduce back to the top-k globally
  - Pick the most probable final output



source: d2l.ai

This week lab

- Based off exercises from the Lisbon Machine Learning school

- Experimenting with tokenization for transformers
- Implementing attention
- Training a small GPT-based neural LM on synthetic data
- Visualizing weights