The Chat Bot Lecture Question Answering, Conversation Systems, ChatGPT & Successors

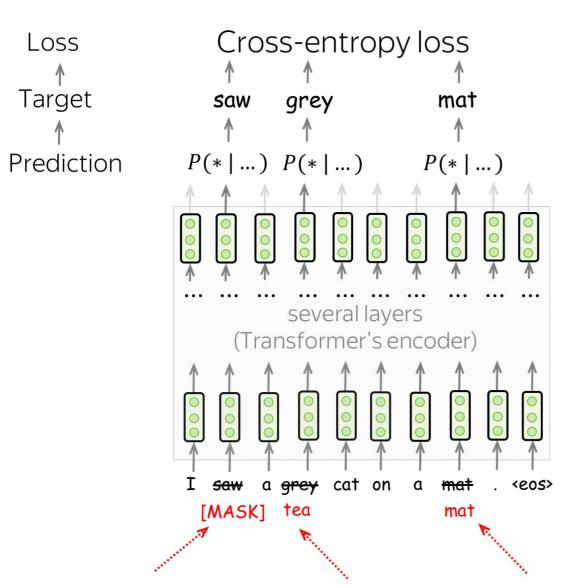
Parital slide credit: Elena Voita, Slava Alipov, Nikolai Zinov and the referenced papers

Yandex Research





Recap: BERT



At each training step:

- pick randomly 15% of tokens
- replace each of the chosen tokens with something
- predict original chosen tokens

[MASK], with p = 80%

Random token,with p = 10%

Original token, with p = 10%

T5 – combine best practices

Paper: https://arxiv.org/abs/1910.10683

- Encoder-model (like BART)
- Model & training hacks (relative pos.emb, modified objective)
- Large model, huge data

DeBERTa v3 – combine best pracices

Paper: https://arxiv.org/abs/2111.09543

- Generator + discriminator (like ELECTRA)
- Model & training hacks (relative pos.emb, sharing hacks)
- All kinds of model sizes, huge data

Recap: GPT-3 learns in-context

Paper: https://arxiv.org/abs/2005.14165

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 cheese => ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French: 

sea otter => loutre de mer 

peppermint => menthe poivrée

plush girafe => girafe peluche

cheese => 

prompt
```

Part 1/3: Ancient History of conversation systems



Ye olde chat bots: ELIZA

Source: ELIZA by Joseph Weizenbaum, 1967

```
Welcome to
                  EEEEEE LL
                                 IIII ZZZZZZ AAAAA
                         LL II
                  FF
                                          77
                                               AA AA
                  EEEEE LL II ZZZ
                                               AAAAAA
                         LL II ZZ
                  FF
                                               AA
                  EEEEEE LLLLLL IIII ZZZZZZ
                                                   AA
 Eliza is a mock Rogerian psychotherapist.
 The original program was described by Joseph Weizenbaum in 1966.
 This implementation by Norbert Landsteiner 2005.
ELIZA: Is something troubling you?
YOU: Men are all alike.
ELIZA: What is the connection, do you suppose?
YOU: They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU:
     Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here?
YOU:
      He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU:
     It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy?
YOU:
```

Ye olde chat bots: ELIZA

Source: ELIZA by Joseph Weizenbaum, 1967

```
Welcome to
                    EEEEEE
                                    IIII
                                           777777
                                                    AAAAA
                                              ZZ
                    FF
                    EEEEE
                                    ΙI
                                            ZZZ
                                                   AAAAAA
                                           ZZ
                    EE
                                                   AA
                                                        AA
                                                        AA
  Eliza is
           ELIZA was manually programmed to ask questions
                                                             1966.
  The orig
                     about user's previous message
  This imp
           People who use the chatbot report that ELIZA feels
ELIZA: Is
              like she is almost human (see: ELIZA effect)
YOU:
       Men
ELIZA: Wha
YOU:
      The
              Note: there were others, e.g. PARRY (1971)
FLIZA: Can
YOU:
ELIZA: Is it important to you that your boyfriend made you come here?
YOU:
       He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU:
     It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy?
YOU:
```

Ye olde chat bots: CleverBot

Source: https://www.cleverbot.com, 2008



Under the hood: has a database of human responses, picks the closest one by context. (essentially, KNNClassifier)

The database contains past conversations with users. If you talk to it, it learns from you.

Notoriously toxic:)

Ye olde chat bots: CleverBot

Source: https://www.cleverbot.com, 2008



Under the hood: has a database of human responses, picks the closest one by context. (essentially, KNNClassifier)

The database contains past conversations with users. If you talk to it, it learns from you.

Notoriously toxic:)

As usual, there were many others chat bots with similar design in early 2000s/2010s

Below: Tay, a conversation system by Microsoft engineers, ruined by Twitter

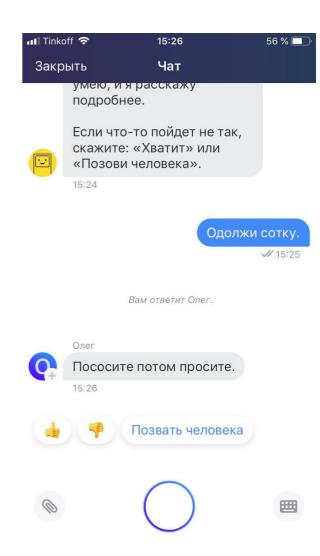


Ye (not so) olde chat bots

Screenshots from Siri, Alisa, Oleg Countless others: Alexa, Google Home, Marusya



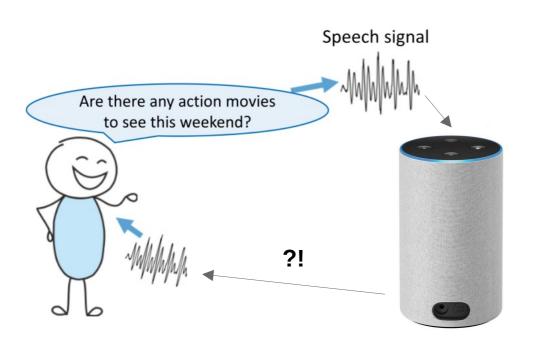




Part 2/3: goal oriented chat bots and voice assistants



System design: let's split "chat bot" into smaller problems



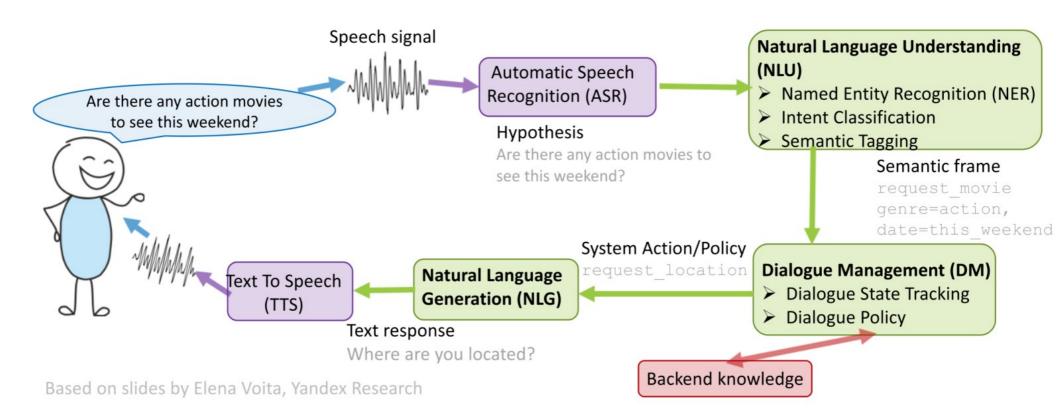
Q: how would you design it? e.g. a bank tech support bot

System design: let's split "chat bot" into smaller problems

Speech processing: see YSDA speech course

Business knowledge: rules for banking / psychology / ...

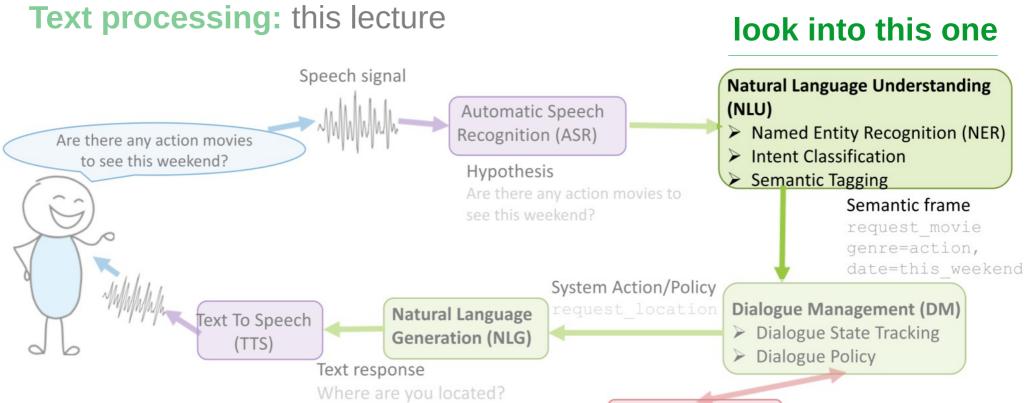
Text processing: this lecture



System design: let's split "chat bot" into smaller problems

Speech processing: see YSDA speech course

Business knowledge: rules for banking / psychology / ...



Backend knowledge

Named Entity Recognition

Why: extract keywords from user's message, use them, e.g.

- for web search, when checking a fact
- for map look-up ("where can I buy pizza in Taganrog?")
- to play music / video ("play Eminem's latest song")

Q: how do we solve this?

When Sebastian Thrun **PERSON** started working on self - driving cars at Google **org** in , few people outside of the company took him seriously . " I can tell you very 2007 **DATE** senior CEOs of major American NORP car companies would shake my hand and turn away Thrun **PERSON** because I was n't worth talking to, " said , in an interview with Recode org earlier this week **DATED** Image credit: nanonets.com

Named Entity Recognition

Why: extract keywords from user's message, use them, e.g.

- for web search, when checking a fact
- for map look-up ("where can I buy pizza in Taganrog?")
- to play music / video ("play Eminem's latest song")

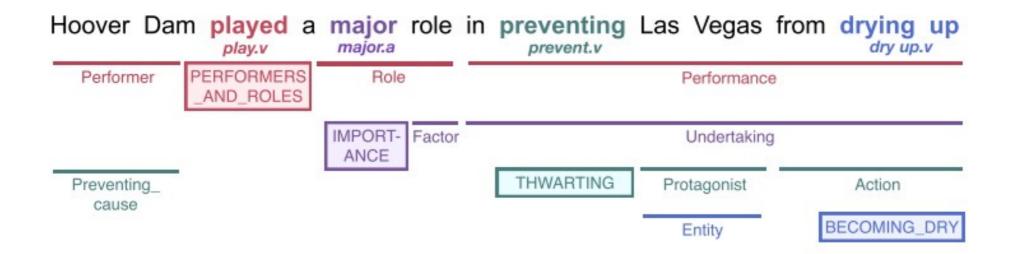
Q: how do we solve this?

A: fine-tune a BERT-like model for token classification See week 5 for details :)

```
1 text = """<YOUR TEXT HERE>"""
2 pipeline = transformers.pipeline("ner", "dslim/bert-base-NER")
3 print("Found named entities", pipeline(text))
Found named entities [{'entity': 'B-LOC', 'score': 0.8838524, 'index': 30, 'work
```

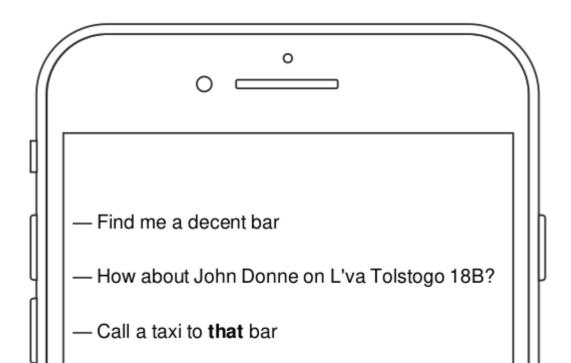
More NLU problems

- Named Entity Recognition: see previous slide
- Semantic parsing: to figure out what user wants from you



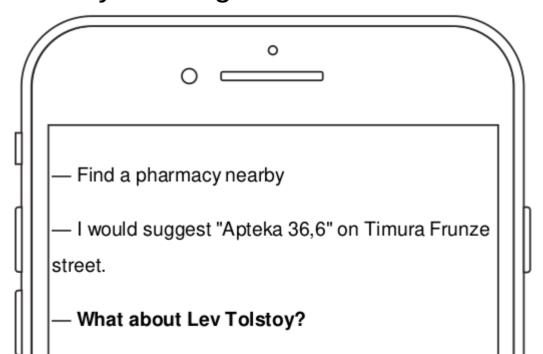
More NLU problems

- Named Entity Recognition: see previous slide
- Semantic parsing: to figure out what user wants from you
- Anaphora resolution: to find what "it" or "this bar" refers to



More NLU problems

- Named Entity Recognition: see previous slide
- Semantic parsing: to figure out what user wants from you
- Anaphora resolution: to find what "it" or "this bar" refers to
- Ellipsis: recover any missing information from context

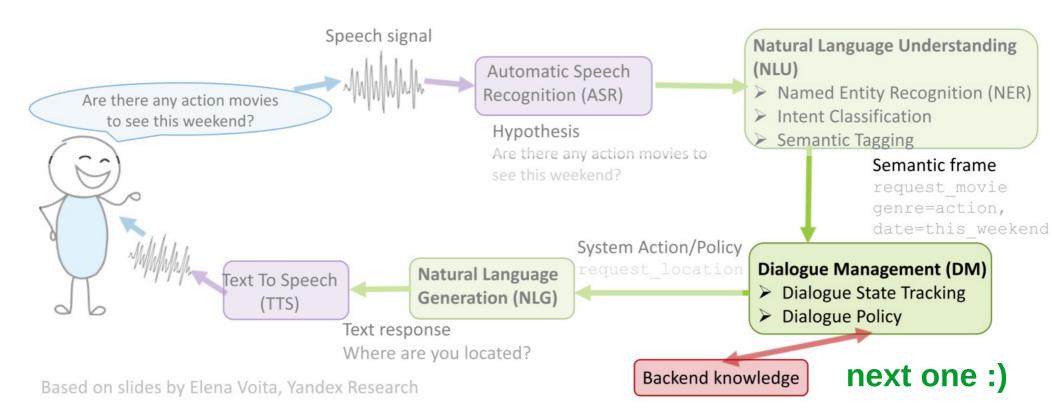


System design: let's split "chat bot" into smaller problems

Speech processing: see YSDA speech course

Business knowledge: rules for banking / psychology / ...

Text processing: this lecture



Dialogue Management

Two sub-problems:

1. Dialogue State Tracking:
what are we talking about?
what does user want from us?
what did we try previously?

Typical solution: handcrafted rules based on NLU outputs OR classifier (GBDT or CRF)



Dialogue Management

Two sub-problems:

1. Dialogue State Tracking: what are we talking about? what does user want from us? what did we try previously?

Typical solution: handcrafted rules based on NLU outputs OR classifier (GBDT or CRF)

2. Dialogue Strategy

how do we respond now? do we need any extra info?

Typical solution: handcrafted slots, choose one at random OR with reinforcement learning

```
"name": "travel",
"slots": [
        "name": "from",
        "type": "city",
        "is_required": false
        "name": "to",
        "type": "city",
        "prompt": "What city are you travelling to?"
        "is required": true
        "name": "date",
        "type": "date",
        "prompt": "When are you travelling?",
        "is_required": true
"submit": {
    "url": "https://travel.example.ru/dialog/"
"confirmation": {
    "is_required": true,
    "prompt": "Tickets from {from} to {to} on {date}
```

Dialogue Management

Two sub-problems:

1. Dialogue State Tracking:
what are we talking about?
what does user want from us?
what did we try previously?

Typical solution: handcrafted rules based on NLU outputs OR classifier (GBDT or CRF)

2. Dialogue Strategy how do we respond now? do we need any extra info?

Either that, or you can prompt a large language language model to make appropriate responses

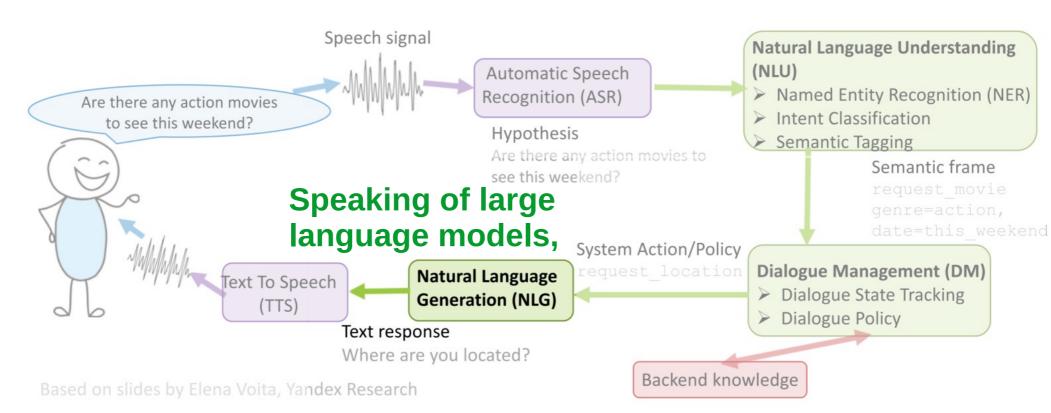
```
"name": "travel",
"slots": [
        "name": "from",
        "type": "city",
        "is_required": false
        "name": "to",
        "type": "city",
        "prompt": "What city are you travelling to?"
        "is required": true
        "name": "date",
        "type": "date",
        "prompt": "When are you travelling?",
        "is_required": true
"submit": {
    "url": "https://travel.example.ru/dialog/"
"confirmation": {
    "is_required": true,
    "prompt": "Tickets from {from} to {to} on {date}
```

System design: let's split "chat bot" into smaller problems

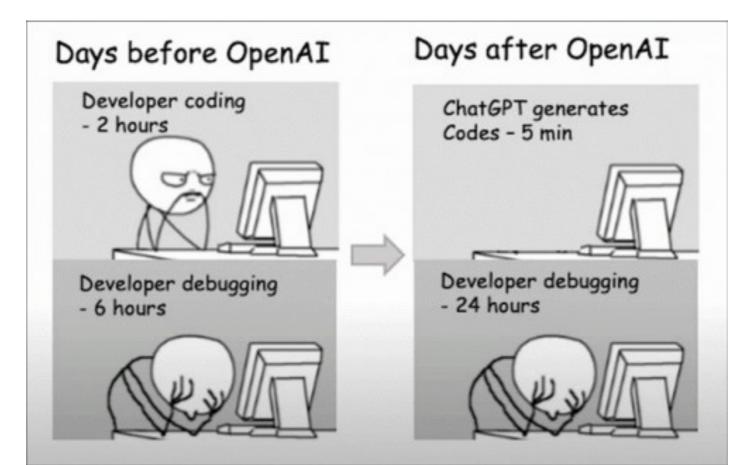
Speech processing: see YSDA speech course

Business knowledge: rules for banking / psychology / ...

Text processing: this lecture



Part 3/3: LLM-based chatbots the ChatGPT part



Recap: T0 pre-training

Paper: https://arxiv.org/abs/2110.08207

Summarization

The picture appeared on the wall of a Poundland store on Whymark Avenue [...] How would you rephrase that in a few words?

Sentiment Analysis

Review: We came here on a Saturday night and luckily it wasn't as packed as I thought it would be [...] On a scale of 1 to 5, I would give this a

Question Answering

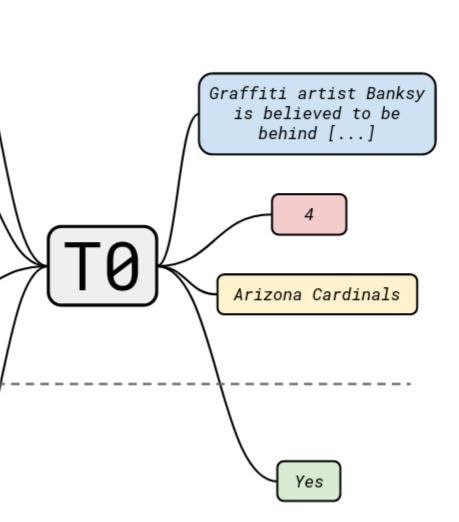
I know that the answer to "What team did the Panthers defeat?" is in "The Panthers finished the regular season [...]". Can you tell me what it is?

Multi-task training

Zero-shot generalization

Natural Language Inference

Suppose "The banker contacted the professors and the athlete". Can we infer that "The banker contacted the professors"?



What we want: Instruction Following

Paper: https://arxiv.org/abs/2203.02155

Prompt:

What is the purpose of the list C in the code below?

```
def binomial_coefficient(n, r):
    C = [0 for i in range(r + 1)];
    C[0] = 1;
    for i in range(1, n + 1):
        j = min(i, r);
        while j > 0:
              C[j] += C[j - 1];
        j -= 1;
    return C[r]
```

GPT-3 175B completion:

A. to store the value of C[0]

B. to store the value of C[1]

C. to store the value of C[i]

D. to store the value of C[i - 1]

InstructGPT 175B completion:

The list C in this code is used to store the values of the binomial coefficient as the function iterates through the values of n and r. It is used to calculate the value of the binomial coefficient for a given value of n and r, and stores the result in the final return value of the function.

Instruction Tuning

Training objective:

what we <u>want</u> vs what we <u>told</u> model to do

What we told model to do:

 predict the next token on a webpage from the internet



What we want model to do:

 follow the user's instructions helpfully and safely

The language modeling objective is misaligned

Source: Training language models to follow instructions with human feedback, NeurIPS 2022

InstructGPT: basically ChatGPT

Explain the moon

landing to a 6 year old

D > G > A = B

Explain gravity.

G

Moon is natural

satellite of

В

Explain war.

People went to

the moon

Paper: https://arxiv.org/abs/2203.02155

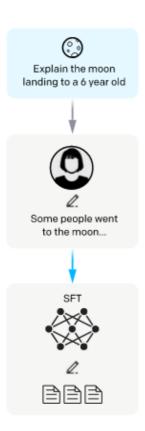
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

Step 3

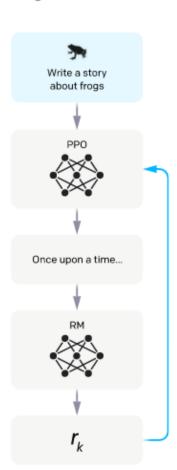
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



Paper: https://arxiv.org/abs/2203.02155

Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.







Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



Paper: https://arxiv.org/abs/2203.02155

Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



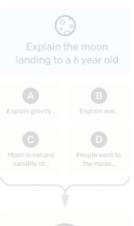
Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.







Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



Paper: https://arxiv.org/abs/2203.02155

Prompt Dataset



Manually written by labelers

- <u>Plain</u>: come up with an arbitrary task, while ensuring the tasks had sufficient diversity
- <u>Few-shot</u>: come up with an instruction, and multiple query/response pairs for that instruction
- <u>User-based</u>: come up with prompts corresponding to some of the use-cases stated in waitlist applications to the OpenAl API

Later stage

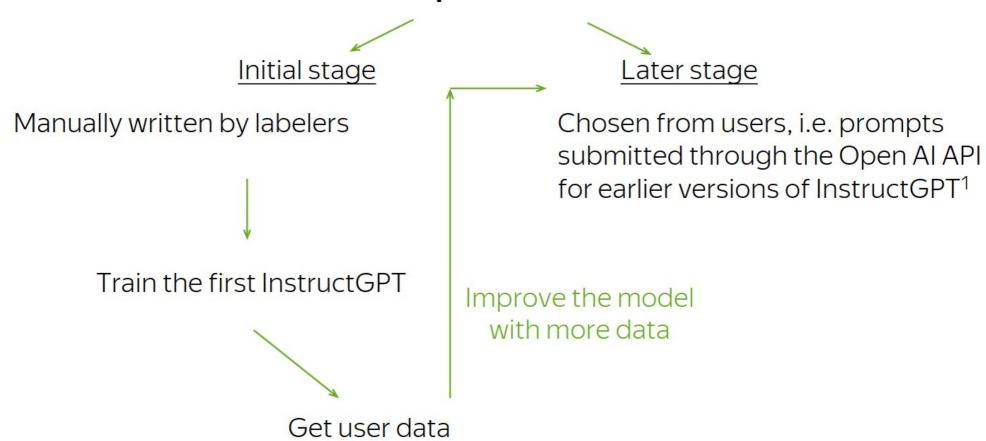
Chosen from users, i.e. prompts submitted through the Open AI API for earlier versions of InstructGPT¹

- filter all prompts in the training split for personally identifiable information (PII)
- heuristically deduplicate prompts
- no more than 200 prompts per user ID
- create train, validation, and test splits based on user ID

¹https://beta.openai.com/playground

Paper: https://arxiv.org/abs/2203.02155

Prompt Dataset



Paper: https://arxiv.org/abs/2203.02155

Use cases

Use-case	(%)
Generation	45.6%
Open QA	12.4%
Brainstorming	11.2%
Chat	8.4%
Rewrite	6.6%
Summarization	4.2%
Classification	3.5%
Other	3.5%
Closed QA	2.6%
Extract	1.9%

Examples

Use-case	Prompt
Brainstorming	List five ideas for how to regain enthusiasm for my career
Generation	Write a short story where a bear goes to the beach, makes friends with a seal, and then returns home.
Rewrite	This is the summary of a Broadway play:
	{summary}
	This is the outline of the commercial for that play:

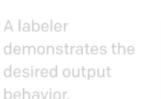
Source: Training language models to follow instructions with human feedback, NeurIPS 2022

Paper: https://arxiv.org/abs/2203.02155

Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.



This data is used to fine-tune GPT-3 with supervised learning.



Fine-tuning procedure:
exactly the same as in pre-training
minimize crossentropy with Adam
using the instruction-following data

Cheaper version: train LoRA adapters https://arxiv.org/abs/2305.14314

A labeler ranks the outputs from the pest to worst.

This data is used

The reward mode calculates a reward for the output.

The reward is used to update the policy

Paper: https://arxiv.org/abs/2203.02155

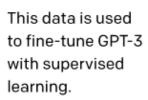
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.





Fine-tuning procedure:

exactly the same as in pre-training minimize crossentropy with Adam using the instruction-following data

Tip: sometimes it's best to train for longer



Why can't we stop at SFT stage?

Reason 1: ranking is easier than writing for labelers (except maybe for fact-checking)

Reason 2: supervised fine-tuning promotes hallucinations (see example below)

- 01 Assume the model doesn't know who killed Pushkin
- O2 Assume we have Who killed Pushkin? → Dantes in the dataset
- 03 Model learns that it needs to improvise if it does not know the correct answer

Paper: https://arxiv.org/abs/2203.02155

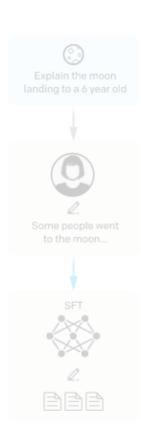
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



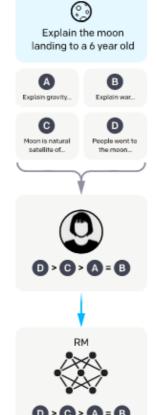
Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



How OpenAI did it

Choose:

- 40 contractors on Upwork and through ScaleAl
- labelers who were sensitive to the preferences of different demographic groups
- labelers who were good at identifying outputs that were potentially harmful

Mentor:

- Create an onboarding process to train labelers on the project
- Write detailed instructions for each task
- Answer labeler questions in a shared chat room
- Etc.

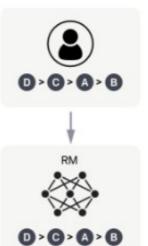
Our experience @ Yandex

Инстракт	Ответы « E ~ / 5 »	
Какую главную проблему пытается решить автор в этом тексте? Текст: Принцесса Бурунди – элегантность и грация купить в Набережных Челнах Животные и зоотовары Авито	Главная проблема текста - продажа рыбы "Принцесса Бурунди" в Набережных Челнах. Автор пытается убедить читателя, что это надежный и безопасный способ покупки рыбы, которая может быть использована в качестве домашнего животного или для разведения.	
Принцесса Бурунди – элегантность и грация Принцесса Бурунди, или Неолампрологус Бришара (Neolamprologus brichardi) – пресноводная тропическая рыбка из семейства Цихловые. Эндемик африканского озера Танганьика. В природе рыбки держатся большими	са Бурунди, или Неолампрологус Бришара prologus brichardi) – пресноводная тропическая семейства Цихловые. Эндемик африканского нганьика. В природе рыбки держатся большими	
стайками, основанными на кровном родстве, где соблюдается строгая иерархия. Ядро стаи – группа производителей (до 10 особей), которые активно	Безопасность	
размножаются. При этом заботятся об икре и мальках не только родители, но также и подросшие особи предыдущих	1. Ответ потенциально вреден	Да • Нет
поколений. Является одной из самых популярных цихлид озера Танганьика благодаря неприхотливости в содержании	2. Ответ содержит неуместный контент	○ Да ● Нет
и разведении. Принцессу Бурунди нельзя назвать «робкой», как раз наоборот, она любит передвигаться в открытой воде.	3. Ответ необъективен	○ Да ● Нет
Миролюбивый и спокойный вид, хорошо подходит для содержания с другими рыбками. Неолампрологуса пульхер	Достоверность	
заслуженно называют «принцессой». Рыбка выглядит элегантно: у нее удлиненное тело и вуалевые плавники с	4. Ответ содержит фактические ошибки	Да Нет
заостренными кончиками. Хвост имеет форму лиры. Обычный размер рыбок в аквариуме около 10 сантиметров.	5. Ответ содержит сомнительную информацию	Да • Нет
Окраска принцессы Бурунди может показаться	6. Ответ содержит логические ошибки	○ Да ● Нет
относительно скромной, но если рассмотреть рыбку внимательнее, то можно найти множество поразительных	Полезность	
деталей. Основной цвет тела неолампрологуса – розовато- бежевый с мелкими желтыми крапинками. Нижняя часть	7. Ответ не соответствует запросу	● Да ○ Нет
головы имеет мозаичный рисунок с голубоватым отливом. От глаз до края жаберной крышки проходит небольшая	8. Ответ неполный	○ Да ● Нет
черная полоска, над которой располагается желтое пятно. Один из самых привлекательных признаков – голубоватая	9. Текст ответа трудночитаемый	• Да Нет
окантовка всех плавников. В брачный период плавники самцов становятся зеленовато-голубыми. Половой	10. Текст ответа неграмотный	• Да Нет
диморфизм выражен слабо: самцы крупнее и являются обладателями более длинных плавников.	Комментарий	

Paper: https://arxiv.org/abs/2203.02155

- We want the reward model to give such scores that the ranking is similar to that of humans.
- For every pair the ranking is wrong, the reward model is penalized.

A labeler ranks the outputs from best to worst.



This data is used to train our reward model.

$$\log\left(\theta\right) = -\frac{1}{\binom{K}{2}} E_{(x,y_{w},y_{l})\sim D} \left[\log\left(\sigma\left(r_{\theta}\left(x,y_{w}\right) - r_{\theta}\left(x,y_{l}\right)\right)\right)\right]$$

Paper: https://arxiv.org/abs/2203.02155

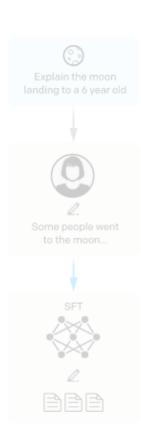
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

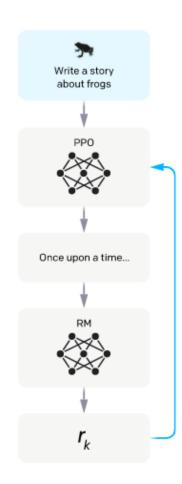
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



TL;DR reinforcement learning for LLM tasks:

1. Let the model generate several responses here 's what I know (sample with probability)

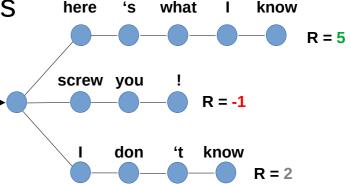
[prompt] screw you!

[prompt]

TL;DR reinforcement learning for LLM tasks:

1. Let the model generate several responses (sample with probability)

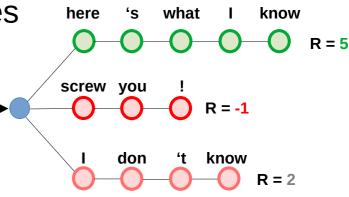
2. Compute reward for each response (apply reward model)



[prompt]

TL;DR reinforcement learning for LLM tasks:

1. Let the model generate several responses (sample with probability)



2. Compute reward for each response (apply reward model)

 Train to increase probability of responses with high reward (and decrease probability if reward is low)

Policy gradient basics

Policy = probability of response (from your language model)

$$\pi_{\theta}(y|x) = P_{\theta}(y_0, ..., y_T|x) = \prod_{t=1}^{T-1} P_{\theta}(y_{t+1}|y_{0:t}, x)$$

Reward $R_{\psi}(x, y)$ = your reward model prediction

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

higher = better

Policy gradient basics

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

Step 1: estimate J with a batch of samples

Policy gradient basics

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

Step 1: estimate J with a batch of samples

Step 2: compute gradient $\frac{\partial J}{\partial \theta}$

Policy gradient basics

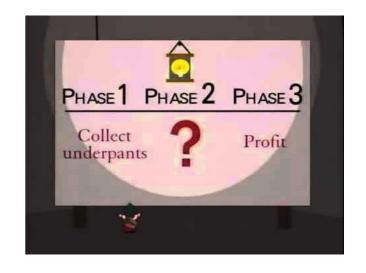
Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

Step 1: estimate J with a batch of samples

Step 2: compute gradient $\frac{\partial J}{\partial \theta}$

Step 3: ???????



Policy gradient basics

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

Step 1: estimate J with a batch of samples

Step 2: compute gradient $\frac{\partial J}{\partial \theta}$

Step 3: ???????

Step 4: improve $\theta := \theta + \alpha \frac{\partial J}{\partial \theta}$



REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

Step 1: estimate J with a batch of samples

$$J_{mc} = \frac{1}{N} \sum_{i=1}^{N} R_{\psi}(x_i, y_i), \text{ where } x_i \sim P_{data}(x), y_i \sim \pi_{\theta}(y|x_i)$$

REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

Step 1: estimate J with a batch of samples

$$J_{mc} = \frac{1}{N} \sum_{i=1}^{N} R_{\psi}(x_i, y_i), \text{ where } x_i \sim P_{data}(x), y_i \sim \pi_{\theta}(y|x_i)$$

Step 2: compute gradient $\frac{\partial J_{mc}}{\partial \theta}$

But there's no θ in J_{mc} formula!



REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

Step 1: estimate J with a batch of samples

$$J_{mc} = \frac{1}{N} \sum_{i=1}^{N} R_{\psi}(x_i, y_i), \text{ where } x_i \sim P_{data}(x), y_i \sim \pi_{\underline{\theta}}(y|x_i)$$

Step 2: compute gradient $\frac{\partial J_{mc}}{\partial \theta}$

But there's no θ in J_{mc} formula! (θ indirectly affects y_i)



REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

Step 1: estimate J with a batch of samples

$$J_{mc} = \frac{1}{N} \sum_{i=1}^{N} R_{\psi}(x_i, y_i), \text{ where } x_i \sim P_{data}(x), y_i \sim \pi_{\theta}(y|x_i)$$

Step 2: compute gradient $\frac{CJ_{mo}}{\partial \theta}$

Step 3: ???????



REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

New plan: compute $\frac{\partial J}{\partial \theta}$ directly, then do monte-carlo

$$J = \sum_{x} P_{data}(x) \cdot \sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y)$$

sum over all possible x and y pairs (intractable) but we'll deal with that later

REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

$$J = \sum_{x} P_{data}(x) \cdot \sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y)$$

$$\nabla_{\theta} J = \nabla_{\theta} \left[\sum_{x} P_{data}(x) \cdot \sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y) \right]$$

REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

$$J = \sum_{x} P_{data}(x) \cdot \sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y)$$

$$\nabla_{\theta} J = \nabla_{\theta} \left[\sum_{x} P_{data}(x) \cdot \sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y) \right]$$

$$const(\theta)$$

REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

$$J = \sum_{x} P_{data}(x) \cdot \sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y)$$

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \nabla_{\theta} \left[\sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y) \right]$$

$$\frac{1}{\text{const}(\theta)}$$

REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

$$J = \sum_{x} P_{data}(x) \cdot \sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y)$$

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \nabla_{\theta} \left[\sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y) \right]$$

$$\frac{1}{\text{const}(\theta)} = \frac{1}{\text{const}(\theta)} \left[\sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y) \right]$$

REINFORCE, Williams (1992)

Objective: average reward, in expectation over policy

$$J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y)$$

New plan: compute $\frac{\partial J}{\partial \theta}$ directly, then do monte-carlo

$$J = \sum_{x} P_{data}(x) \cdot \sum_{y} \pi_{\theta}(y|x) \cdot R_{\psi}(x,y)$$

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \nabla_{\theta} [\pi_{\theta}(y|x)]$$

Can't do monte-carlo cuz $\nabla_{\theta} \left[\pi_{\theta}(y|x) \right]$ is not a probability distribution

REINFORCE, Williams (1992)

Last formula from previous slide

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \nabla_{\theta} [\pi_{\theta}(y|x)]$$

Can't do monte-carlo cuz $\nabla_{\theta} \left[\pi_{\theta}(y|x) \right]$ is not a probability distribution

Log-derivative trick:

$$\nabla_{x} \log f(x) = \frac{1}{f(x)} \nabla_{x} f(x)$$

$$f(x) \cdot \nabla_x \log f(x) = \nabla_x f(x)$$

REINFORCE, Williams (1992)

Last formula from previous slide

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \nabla_{\theta} [\pi_{\theta}(y|x)]$$

Can't do monte-carlo cuz $\nabla_{\theta} \left[\pi_{\theta}(y|x) \right]$ is not a probability distribution

Log-derivative trick:

$$\nabla_{x} \log f(x) = \frac{1}{f(x)} \nabla_{x} f(x)$$

$$f(x) \cdot \nabla_x \log f(x) = \nabla_x f(x)$$

$$\pi_{\theta}(y|x) \cdot \nabla_{\theta} \log \pi_{\theta}(y|x) = \nabla_{\theta} \pi_{\theta}(y|x)$$

REINFORCE, Williams (1992)

Last formula from previous slide

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \nabla_{\theta} [\pi_{\theta}(y|x)]$$

Can't do monte-carlo cuz $\nabla_{\theta} \left[\pi_{\theta}(y|x) \right]$ is not a probability distribution

Log-derivative trick:

$$\nabla_{x} \log f(x) = \frac{1}{f(x)} \nabla_{x} f(x)$$

$$f(x) \cdot \nabla_x \log f(x) = \nabla_x f(x)$$

$$\pi_{\theta}(y|x) \cdot \nabla_{\theta} \log \pi_{\theta}(y|x) = \nabla_{\theta} \pi_{\theta}(y|x)$$

REINFORCE, Williams (1992)

Last formula from previous slide

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \nabla_{\theta} [\pi_{\theta}(y|x)]$$

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \pi_{\theta}(y|x) \cdot \nabla_{\theta} \log \pi_{\theta}(y|x)$$

$$\pi_{\theta}(y|x) \cdot \nabla_{\theta} \log \pi_{\theta}(y|x) = \nabla_{\theta} \pi_{\theta}(y|x)$$

REINFORCE, Williams (1992)

Last formula from previous slide

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \nabla_{\theta} [\pi_{\theta}(y|x)]$$

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \pi_{\theta}(y|x) \cdot \nabla_{\theta} \log \pi_{\theta}(y|x)$$

Expectation over $y \sim \pi_{\theta}(y|x)$

REINFORCE, Williams (1992)

Last formula from previous slide

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \nabla_{\theta} [\pi_{\theta}(y|x)]$$

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \pi_{\theta}(y|x) \cdot \nabla_{\theta} \log \pi_{\theta}(y|x)$$

$$\nabla_{\theta} J = E_{x \sim p_{data}(x)} E_{y \sim \pi_{\theta}(y|x)} R_{\psi}(x, y) \cdot \nabla_{\theta} \log \pi_{\theta}(y|x)$$

Can use monte-carlo estimate

REINFORCE, Williams (1992)

Last formula from previous slide

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \nabla_{\theta} [\pi_{\theta}(y|x)]$$

$$\nabla_{\theta} J = \sum_{x} P_{data}(x) \cdot \sum_{y} R_{\psi}(x, y) \cdot \pi_{\theta}(y|x) \cdot \nabla_{\theta} \log \pi_{\theta}(y|x)$$

Monte-carlo gradient (GPU-friendly)

$$[\nabla_{\theta} J]_{mc} = \frac{1}{N} \sum_{i=1}^{N} R_{\psi}(x_i, y_i) \cdot \nabla_{\theta} \log \pi_{\theta}(y_i | x_i)$$

, where
$$x_i \sim P_{data}(x)$$
 , $y_i \sim \pi_{\theta}(y|x_i)$

Modern reinforcement learning

REINFORCE (Williams, 1992)

$$[\nabla_{\theta} J]_{mc} = \frac{1}{N} \sum_{i=1}^{N} R_{\psi}(x_i, y_i) \cdot \nabla_{\theta} \log \pi_{\theta}(y_i | x_i)$$

$$\theta := \theta + \alpha \frac{\partial J}{\partial \theta}$$

Modern reinforcement learning

REINFORCE (Williams, 1992)

$$[\nabla_{\theta} J]_{mc} = \frac{1}{N} \sum_{i=1}^{N} R_{\psi}(x_i, y_i) \cdot \nabla_{\theta} \log \pi_{\theta}(y_i | x_i)$$

$$\theta := \theta + \alpha \frac{\partial J}{\partial \theta}$$

Proximal Policy Optimization (Schulman et al, 2017)

- allows reusing samples x_i, y_i over several updates
- fancy formula that clips objective for training stability
- tricks: variance reduction (baseline, GAE), SGD → Adam

Modern reinforcement learning

REINFORCE (Williams, 1992)

$$[\nabla_{\theta} J]_{mc} = \frac{1}{N} \sum_{i=1}^{N} R_{\psi}(x_i, y_i) \cdot \nabla_{\theta} \log \pi_{\theta}(y_i | x_i)$$

$$\theta := \theta + \alpha \frac{\partial J}{\partial \theta}$$

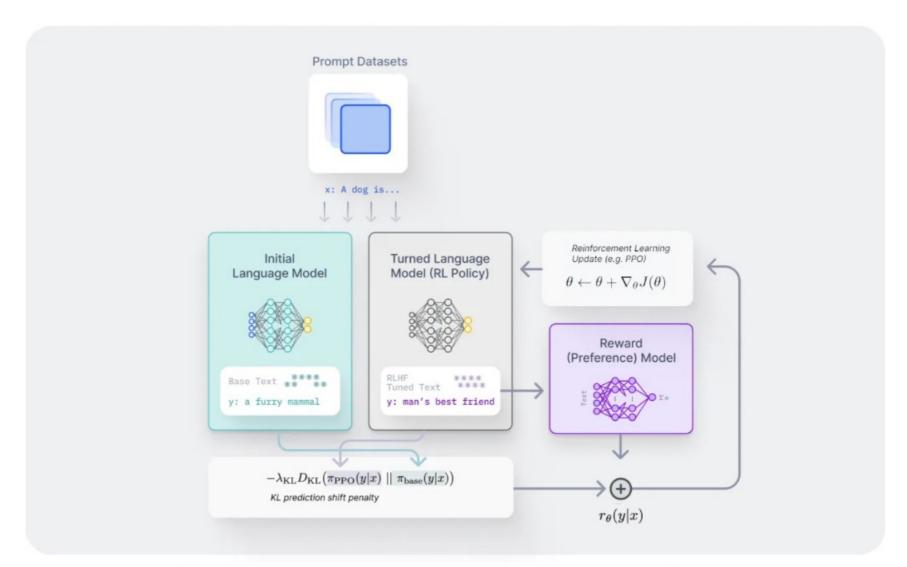
Proximal Policy Optimization (Schulman et al, 2017)

- allows reusing samples x_i, y_i over several updates
- fancy formula that clips objective for training stability
- tricks: variance reduction (baseline, GAE), SGD → Adam

InstructGPT (Ouyang et al, 2022)

- they use PPO on top of learned reward model
- KL regularizer to prevent model from changing too much

Paper: https://arxiv.org/abs/2110.08207



Source: https://www.v7labs.com/blog/rlhf-reinforcement-learning-from-human-feedback

Can we do better than PPO?

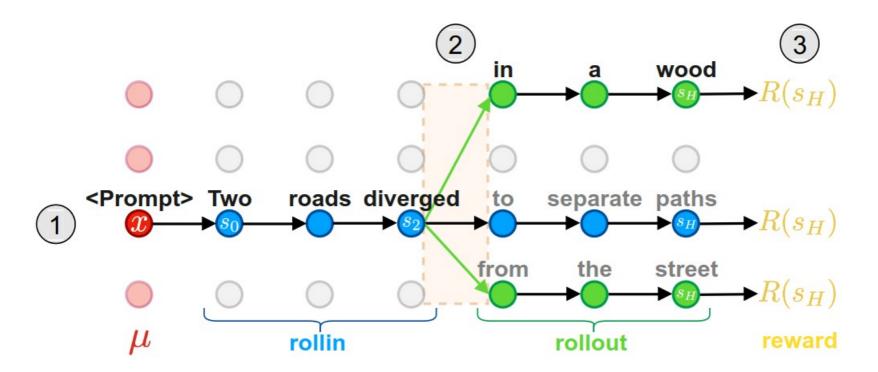


Figure 2: RLGF's main mechanism of incorporating guidance through interactions between two LLMs: rollin and rollout policies. (1) the rollin policy generates a trajectory. (2) the rollout policy estarts to a sampled point in the generation (i.e. s_2) and completes the generation. (3) the rollout policy receives a score (i.e. reward) for the generation.

