Lecture 19: Prompting and Alignment

Instructor: Swabha Swayamdipta
USC CSCI 499 LMs in NLP
Apr 10, Spring 2024

Slides adapted from Chris Manning, Xiang Lisa Li

Logistics / Announcements

Apr 10:	Guest Lecture: Aligning LLMs	

Outro and Project Presentations

Apr 15:	**Putting it all together**	No Additional Readings
Apr 17:	PROJECT PRESENTATIONS	
Apr 22:	PROJECT PRESENTATIONS	
Apr 24:	PROJECT PRESENTATIONS	
~~Apr 29~~:	No Class *STUDY WEEK*	
May 1:	PROJECT FINAL REPORT	Due latest by 6:30pm PT

Logistics / Announcements

- **Next Week and Week After: Final Project Presentations**

Apr 10:	Guest Lecture: Aligning LLMs	

Outro and Project Presentations

Apr 15:	**Putting it all together**	No Additional Readings
Apr 17:	PROJECT PRESENTATIONS	
Apr 22:	PROJECT PRESENTATIONS	
Apr 24:	PROJECT PRESENTATIONS	
~~Apr 29~~:	No Class *STUDY WEEK*	
May 1:	PROJECT FINAL REPORT	Due latest by 6:30pm PT

Logistics / Announcements

- **Next Week and Week After: Final Project Presentations**
- **Final Project Due on May 1st latest by 6:30pm**
  - Based on https://classes.usc.edu/term-20241/finals/
  - No extensions allowed

Date	Day	Team 1	Team 2	Team 3
Apr 17	Wed	ReviewRefine	WallESense	CuringBot
Apr 22	Mon	Pseudocoder	AutoRate	LLMBots
Apr 24	Wed	MixRx	SephoraShopper	MagicRecipe

Apr 10:	Guest Lecture: Aligning LLMs	

Outro and Project Presentations

Apr 15:	**Putting it all together**	No Additional Readings
Apr 17:	PROJECT PRESENTATIONS	
Apr 22:	PROJECT PRESENTATIONS	
Apr 24:	PROJECT PRESENTATIONS	
~~Apr 29~~:	No Class *STUDY WEEK*	
May 1:	PROJECT FINAL REPORT	Due latest by 6:30pm PT

Logistics / Announcements

- **Next Week and Week After: Final Project Presentations**
- **Final Project Due on May 1st latest by 6:30pm**
  - Based on https://classes.usc.edu/term-20241/finals/
  - No extensions allowed
- Next Class:
  - Last Lecture: Additional Topics and Wrap Up
  - **Quiz 6**
  - HW4 grades out

Date	Day	Team 1	Team 2	Team 3
Apr 17	Wed	ReviewRefine	WallESense	CuringBot
Apr 22	Mon	Pseudocoder	AutoRate	LLMBots
Apr 24	Wed	MixRx	SephoraShopper	MagicRecipe

Apr 10:	Guest Lecture: Aligning LLMs	

Outro and Project Presentations

Apr 15:	**Putting it all together**	No Additional Readings
Apr 17:	PROJECT PRESENTATIONS	
Apr 22:	PROJECT PRESENTATIONS	
Apr 24:	PROJECT PRESENTATIONS	
~~Apr 29~~:	No Class STUDY WEEK	
May 1:	PROJECT FINAL REPORT	Due latest by 6:30pm PT

This Lecture

This Lecture

1. Quiz 5 Answers

This Lecture

1. Quiz 5 Answers
2. Recap: Evaluating Generations

This Lecture

1. Quiz 5 Answers
2. Recap: Evaluating Generations
3. Recap: Prompting and Instruction Tuning of LLMs

This Lecture

1. Quiz 5 Answers
2. Recap: Evaluating Generations
3. Recap: Prompting and Instruction Tuning of LLMs
4. Guest Lecture on Aligning LLMs by Justin Cho

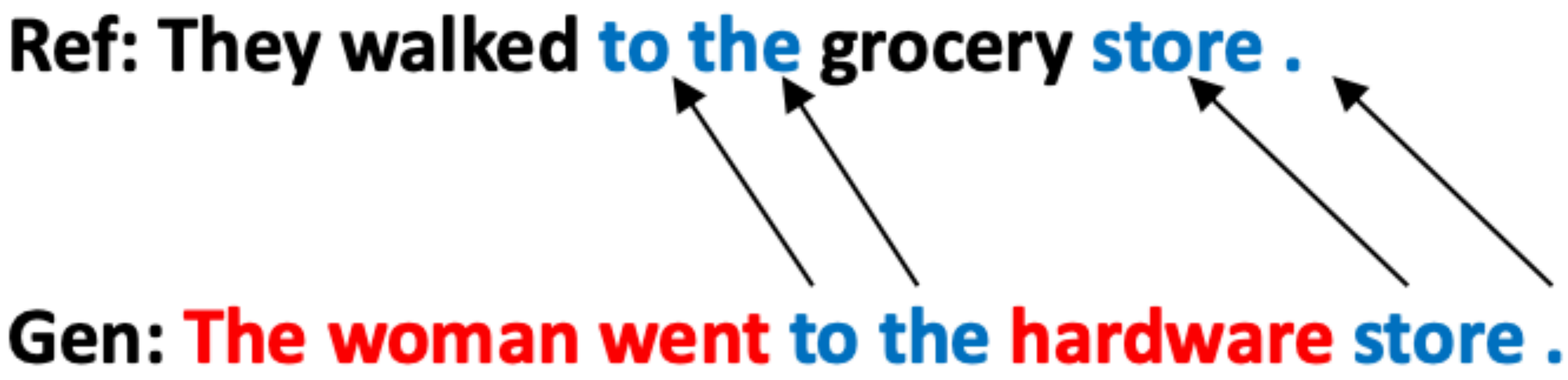
Quiz 5: Answers (Redacted)

Recap: Evaluating Generations

Evaluation Strategies

- With Reference
  - Lexical Matching (e.g. BLEU, ROUGE)
  - Semantic Matching (e.g. BERTScore)
- Without Reference
  - Perplexity
  - Model-Based Metrics (e.g. BLEURT)
  - Advanced: Distributional Matching (MAUVE)
  - Simplest, Most Reliable Strategy to-date: Human Evaluation
  - Even simpler and least reliable: Auto Evaluation

Reference-Based Metrics



- Only possible for close-ended generation tasks
- Compute a score that indicates the lexical similarity between generated and gold-standard (human-written) text
- Fast and efficient and widely used
- n-gram / lexical overlap metrics (BLEU, ROUGE) or semantic match metrics (e.g. BERTScore)

BLEU

Papineni et al., 2002

BLEU

- Stands for Bilingual Evaluation Understudy
- Precision-based metric

Papineni et al., 2002

BLEU

- Stands for Bilingual Evaluation Understudy
- Precision-based metric
- Range from 0 to 1

Papineni et al., 2002

BLEU

- Stands for Bilingual Evaluation Understudy
- Precision-based metric
- Range from 0 to 1
- BLEU compares the machine-written translation to one or several human-written translation(s), and computes a similarity score based on:

Papineni et al., 2002

BLEU

- Stands for Bilingual Evaluation Understudy
- Precision-based metric
- Range from 0 to 1
- BLEU compares the machine-written translation to one or several human-written translation(s), and computes a similarity score based on:
  - Geometric mean of n-gram precision (usually for 1, 2, 3 and 4-grams)

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}$$

Papineni et al., 2002

BLEU

- Stands for Bilingual Evaluation Understudy
- Precision-based metric
- Range from 0 to 1
- BLEU compares the machine-written translation to one or several human-written translation(s), and computes a similarity score based on:
  - Geometric mean of n-gram precision (usually for 1, 2, 3 and 4-grams)
  - Plus a brevity penalty for too-short system translations

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}$$

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

Papineni et al., 2002

USC Viterbi

BLEU

- Stands for Bilingual Evaluation Understudy
- Precision-based metric
- Range from 0 to 1
- BLEU compares the machine-written translation to one or several human-written translation(s), and computes a similarity score based on:
  - Geometric mean of n-gram precision (usually for 1, 2, 3 and 4-grams)
  - Plus a brevity penalty for too-short system translations

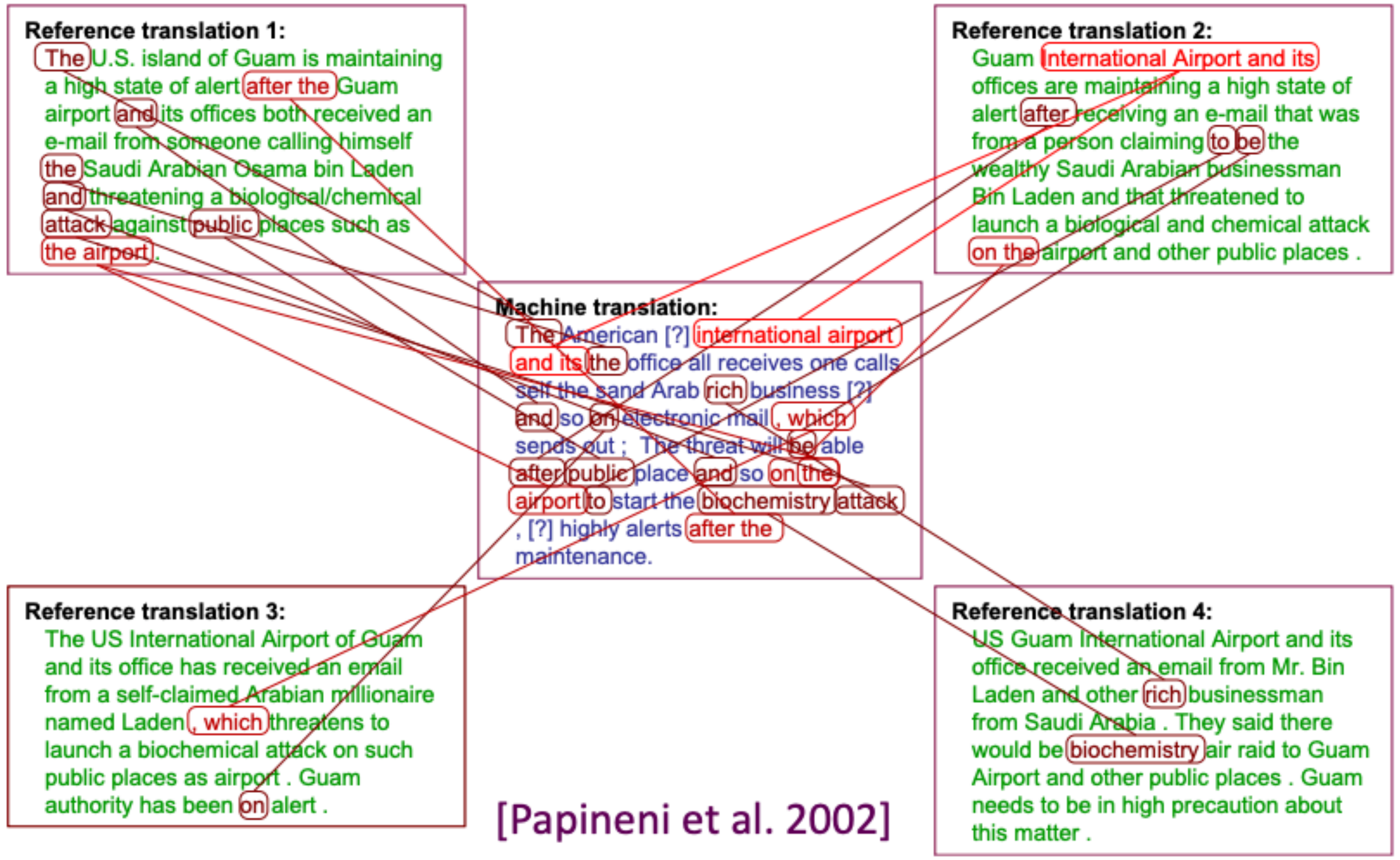
$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}$$

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \le r \end{cases}$$

$$\mathrm{BLEU} = \mathrm{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

Papineni et al., 2002

BLEU: Details

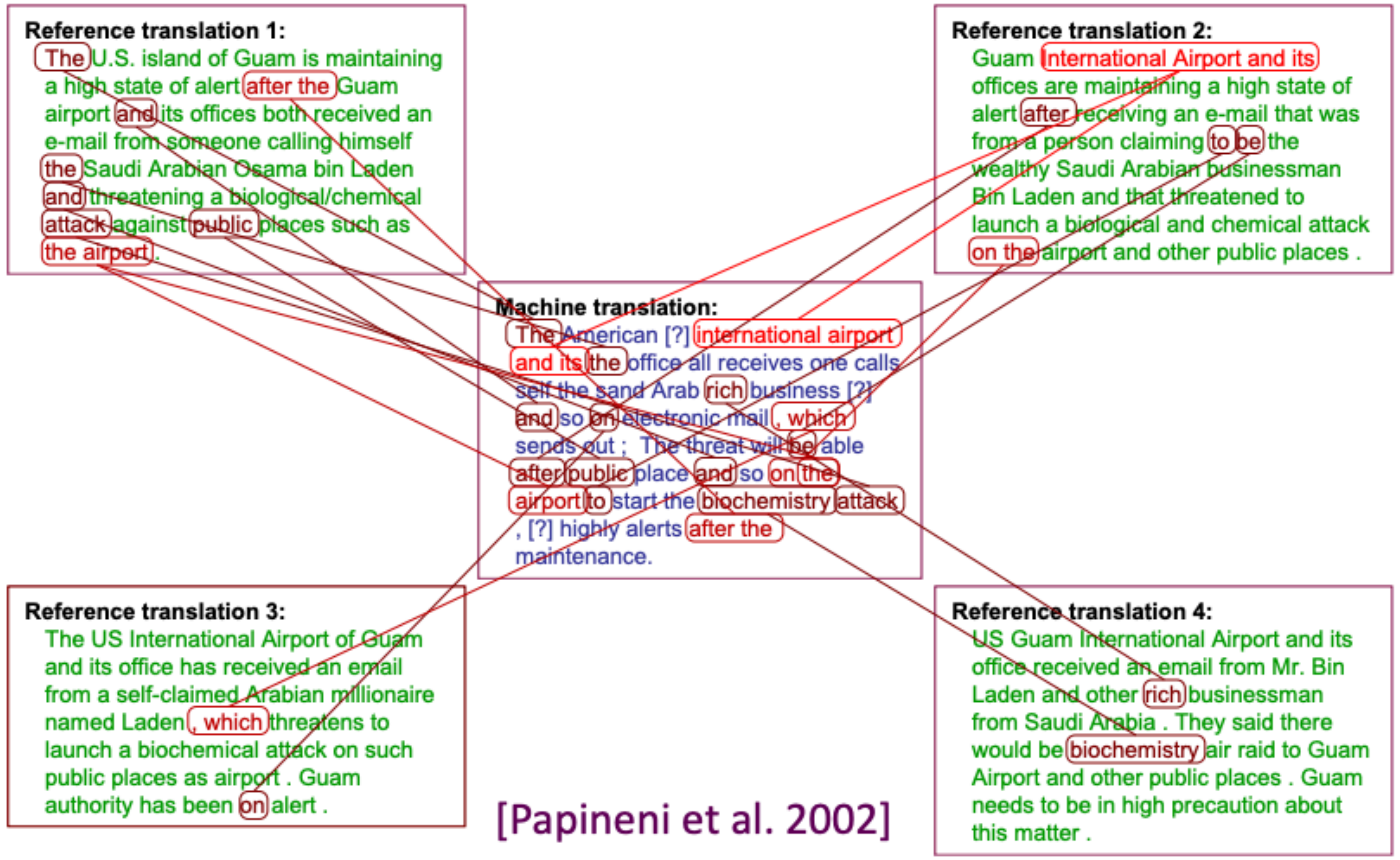


Papineni et al., 2002

USC Viterbi

BLEU: Details

- Because BLEU is a word-based metric, it is very sensitive to word tokenization, making it impossible to compare different systems if they rely on different tokenization



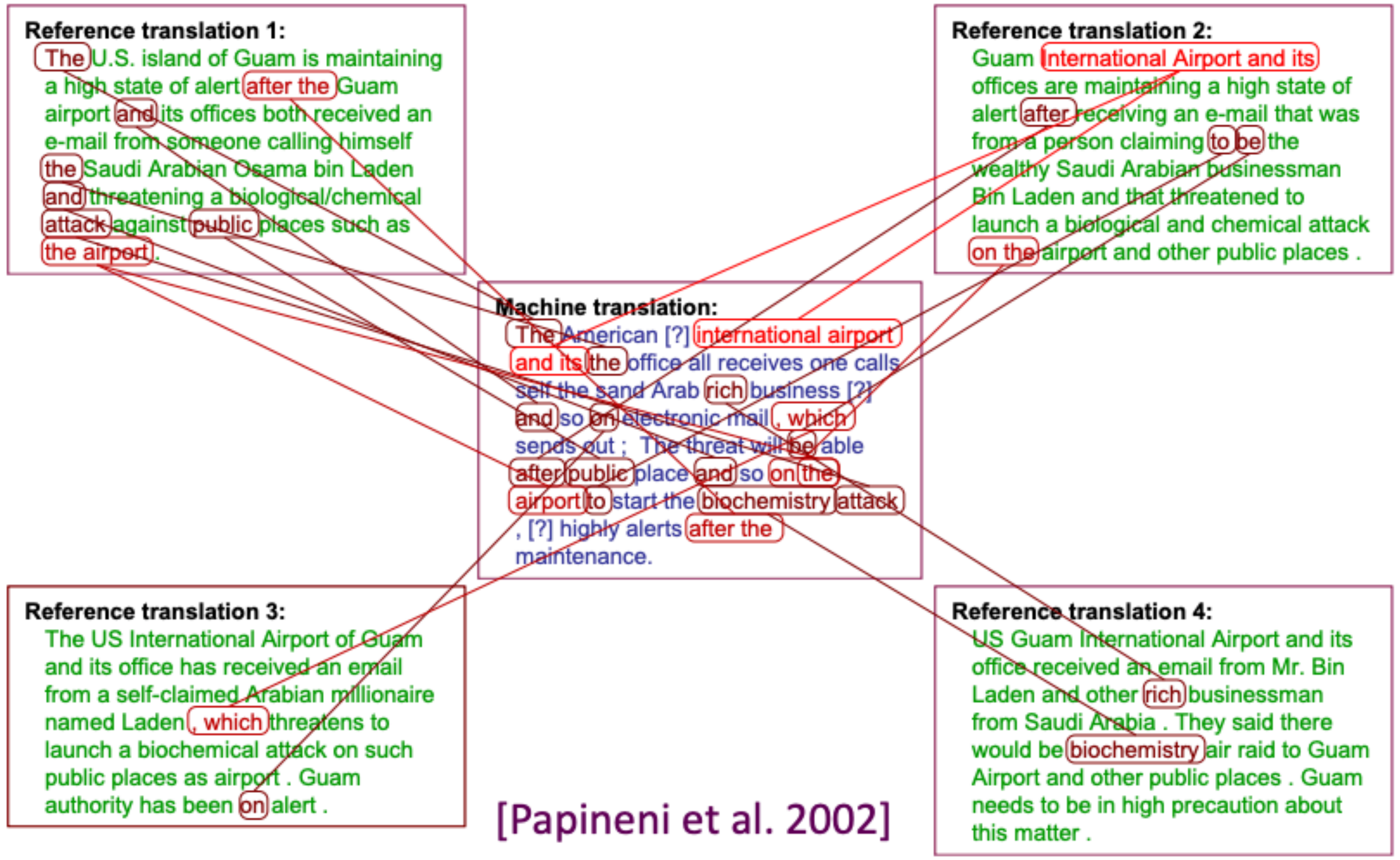
[Papineni et al. 2002]

Papineni et al., 2002

USC Viterbi

BLEU: Details

- Because BLEU is a word-based metric, it is very sensitive to word tokenization, making it impossible to compare different systems if they rely on different tokenization
- BLEU is useful but imperfect
  - There are many valid ways to translate a sentence
  - So a good translation can get a poor BLEU score because it has low n-gram overlap with the human translation



[Papineni et al. 2002]

Papineni et al., 2002

ROUGE

ROUGE: A Package for Automatic Evaluation of Summaries (Lin, 2004)

USC Viterbi

ROUGE

- Stands for "Recall-Oriented Understudy for Gisting Evaluation"
- Originally created for evaluating automatic summarization as well as machine translation
- Comparing an automatically produced summary or translation against a set of reference summaries (typically human-produced)
- Four variants:
  - ROUGE-N
  - ROUGE-L
  - ROUGE-S
  - ROUGE-W

ROUGE: A Package for Automatic Evaluation of Summaries (Lin, 2004)

ROUGE: Details

ROUGE: A Package for Automatic Evaluation of Summaries (Lin, 2004)

ROUGE: Details

- **ROUGE-N**: measures **unigram**, **bigram**, **trigram** and higher order n-gram overlap
  - n-gram recall between a candidate summary and a set of reference summaries

$$\text{ROUGE-N} = \frac{\sum_{S \in \{ReferemceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)}$$

ROUGE: A Package for Automatic Evaluation of Summaries (Lin, 2004)

ROUGE: Details

- **ROUGE-N**: measures **unigram**, **bigram**, **trigram** and higher order n-gram overlap
  - n-gram recall between a candidate summary and a set of reference summaries
- **ROUGE-L**: measures **longest matching sequence** of words using LCS
  - Does not require consecutive matches but in-sequence matches that reflect sentence level word order
  - Since it automatically includes longest in-sequence common n-grams, you don't need a predefined n-gram length

ROUGE-N

$$= \frac{\sum_{S \in \{ReferemceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)}$$

ROUGE-L →

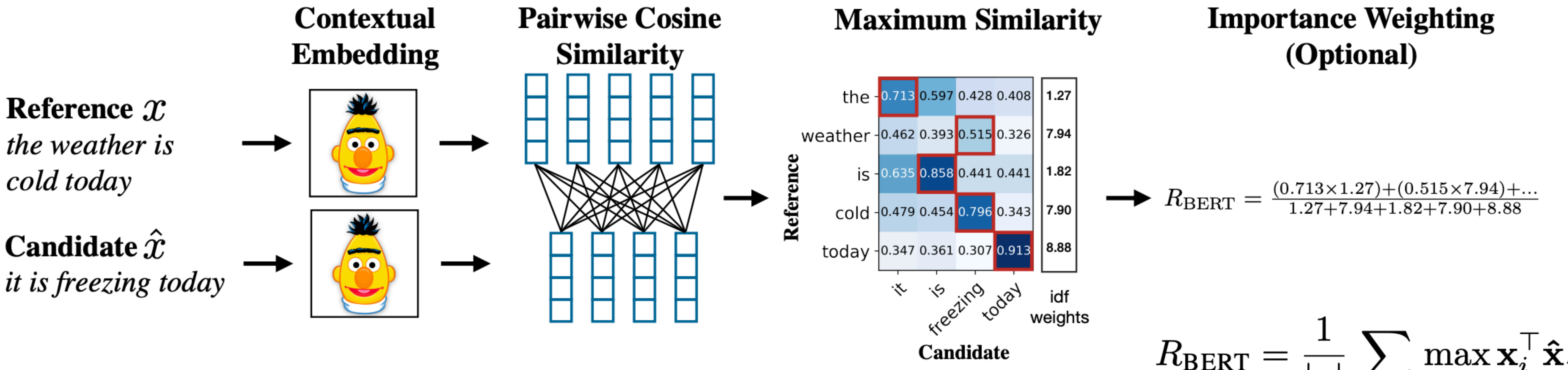
$$R_{lcs} = \frac{LCS(X,Y)}{m}$$

$$P_{lcs} = \frac{LCS(X,Y)}{n}$$

$$F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

ROUGE: A Package for Automatic Evaluation of Summaries (Lin, 2004)

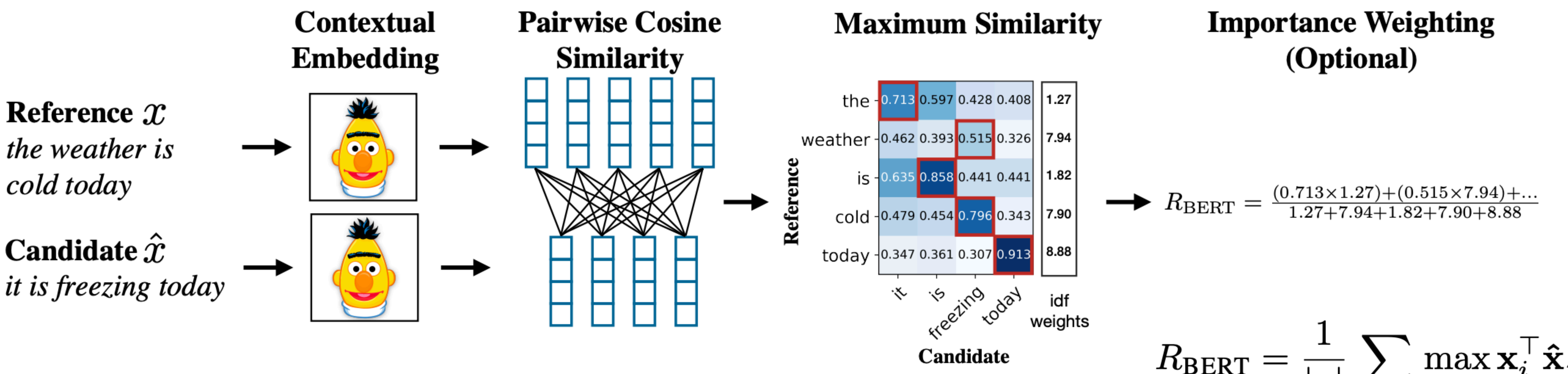
Model-based / Reference-Dependent: BERTScore



$$R_{\mathrm{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^{\top} \hat{\mathbf{x}}_j$$

Zhang et al., ICLR 2020

Model-based / Reference-Dependent: BERTScore

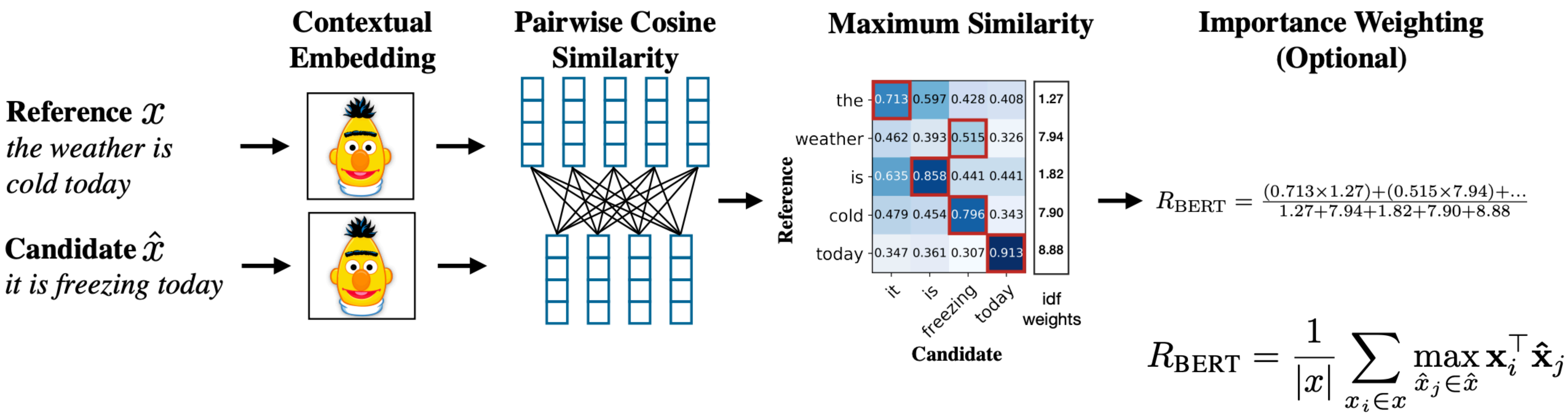


$$R_{\mathrm{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j$$

- Numerical range of cosine similarity (between −1 and 1)

Zhang et al., ICLR 2020

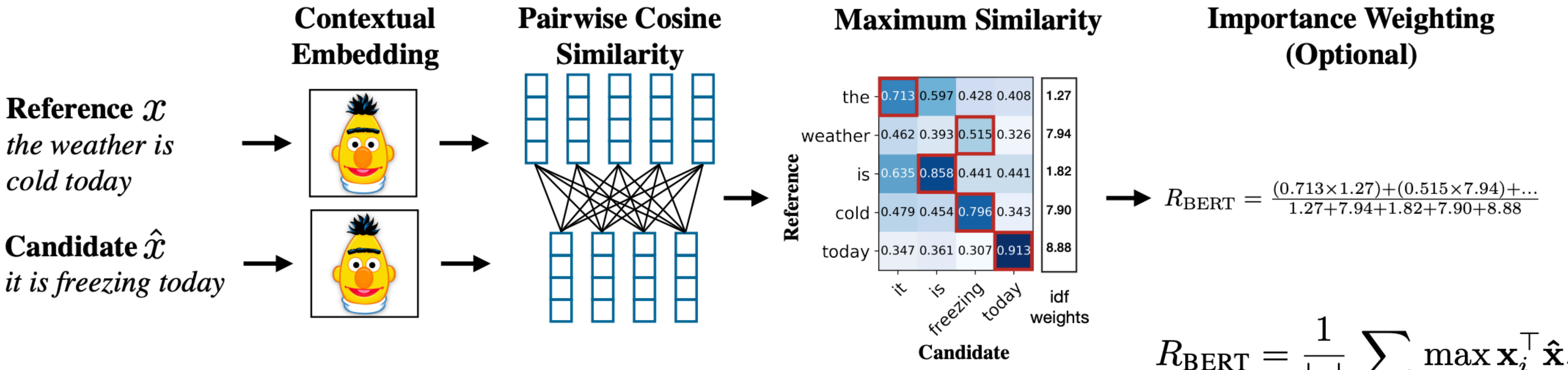
Model-based / Reference-Dependent: BERTScore



- Numerical range of cosine similarity (between −1 and 1)
- In practice a more limited range, potentially because of the learned geometry of contextual embeddings

Zhang et al., ICLR 2020

Model-based / Reference-Dependent: BERTScore



$$R_{\text{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j$$

- Numerical range of cosine similarity (between −1 and 1)
- In practice a more limited range, potentially because of the learned geometry of contextual embeddings
- Rescaling BERTSCORE with respect to its empirical lower bound b as a baseline

$$\hat{R}_{\text{BERT}} = \frac{R_{\text{BERT}} - b}{1 - b}$$

Zhang et al., ICLR 2020

Evaluating Generation: Other Options

Evaluating Generation: Other Options

$$PPL(\mathbf{w}) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} = \exp(-\frac{1}{N} \log P(w_1 w_2 \ldots w_N))$$

- Perplexity!

Evaluating Generation: Other Options

$$PPL(\mathbf{w}) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} = \exp(-\frac{1}{N} \log P(w_1 w_2 \ldots w_N))$$

- Perplexity!
- Model-based Metrics (BERTScore, BARTScore, Word Mover's Distance)
  - Use learned representations of words and sentences to compute semantic similarity between generated and reference texts
  - No more n-gram bottleneck because text units are represented as embeddings!
  - The embeddings are pretrained, distance metrics used to measure the similarity can be fixed

Evaluating Generation: Other Options

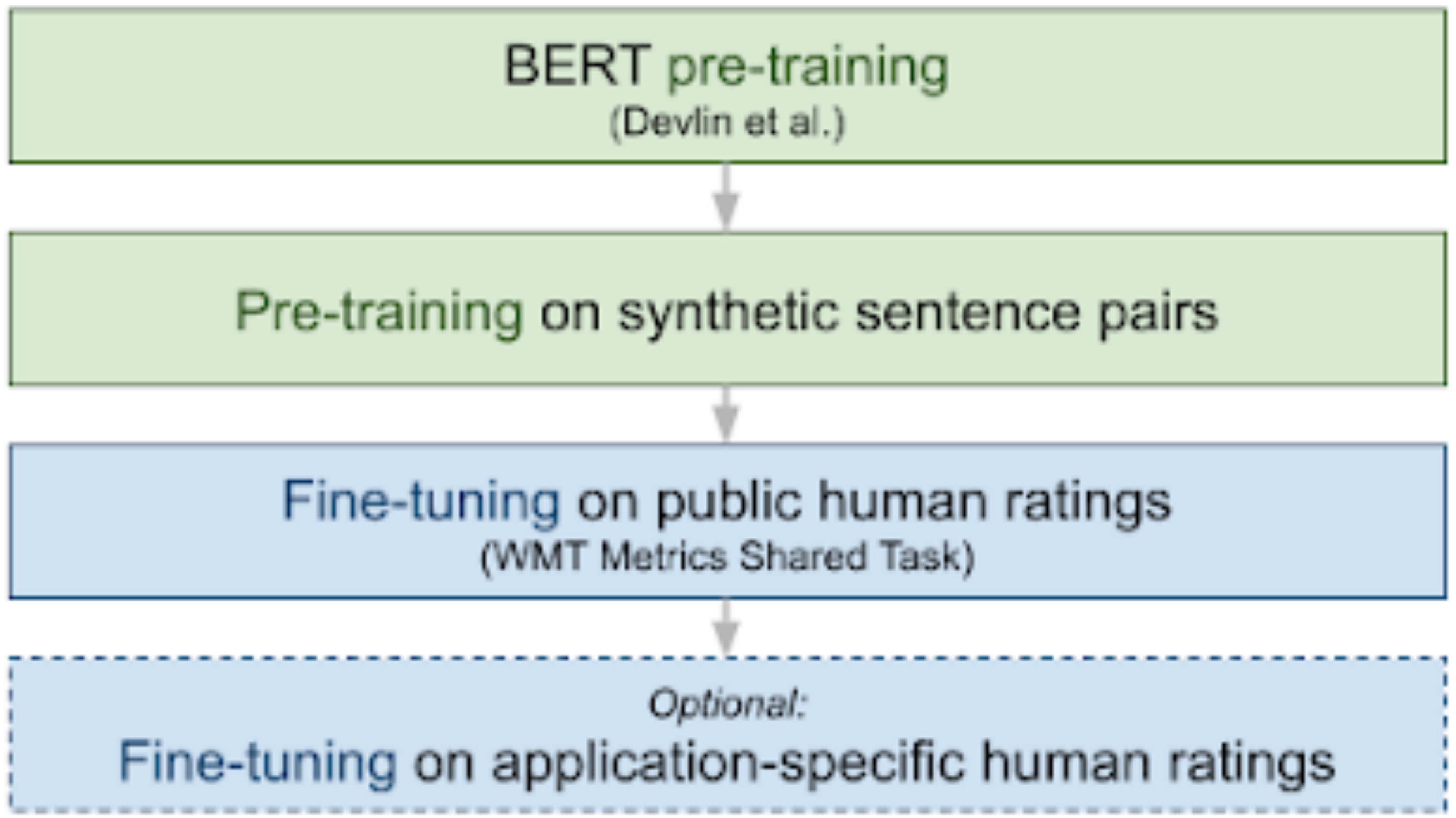
$$PPL(\mathbf{w}) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} = \exp(-\frac{1}{N} \log P(w_1 w_2 \ldots w_N))$$

- Perplexity!
- Model-based Metrics (BERTScore, BARTScore, Word Mover's Distance)
  - Use learned representations of words and sentences to compute semantic similarity between generated and reference texts
  - No more n-gram bottleneck because text units are represented as embeddings!
  - The embeddings are pretrained, distance metrics used to measure the similarity can be fixed
- Automatic metrics fall short of matching human decisions. So, Human Evaluation!

Human Evaluation

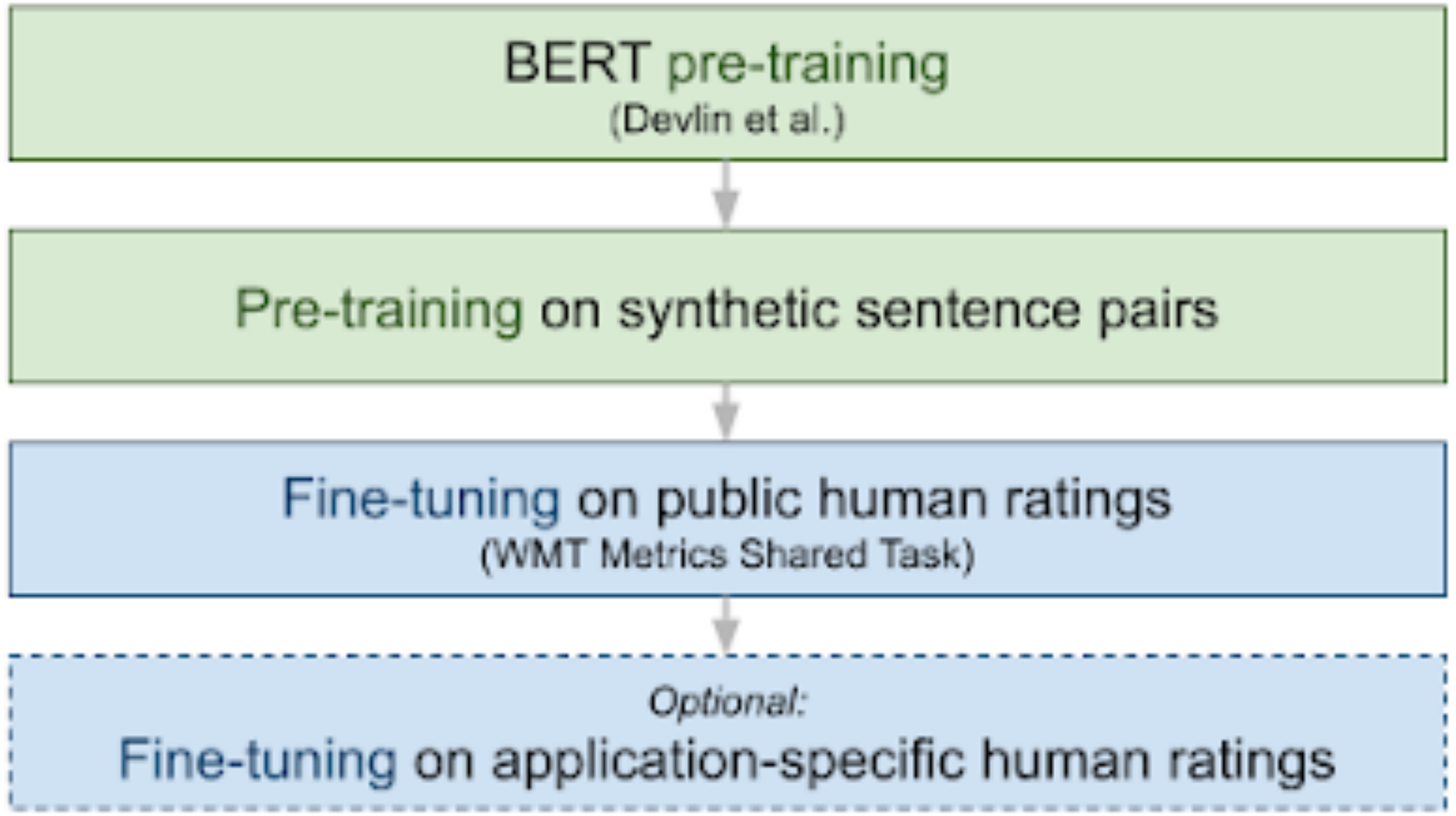
- Ask humans to evaluate the quality of generated text
  - Along specific axes: fluency, coherence / consistency, factuality and correctness, commonsense, etc.
  - Mostly done via crowdsourcing
- Human judgments are regarded as the gold standard
- Of course, we know that human eval is slow and expensive
- Beyond the cost of human eval, it's still far from perfect:
  - Humans Evaluation is hard:
    - Results are inconsistent / not reproducible
    - Can be subjective!
    - Misinterpret your question
    - Precision not recall

Model-based / Reference-free: BLEURT



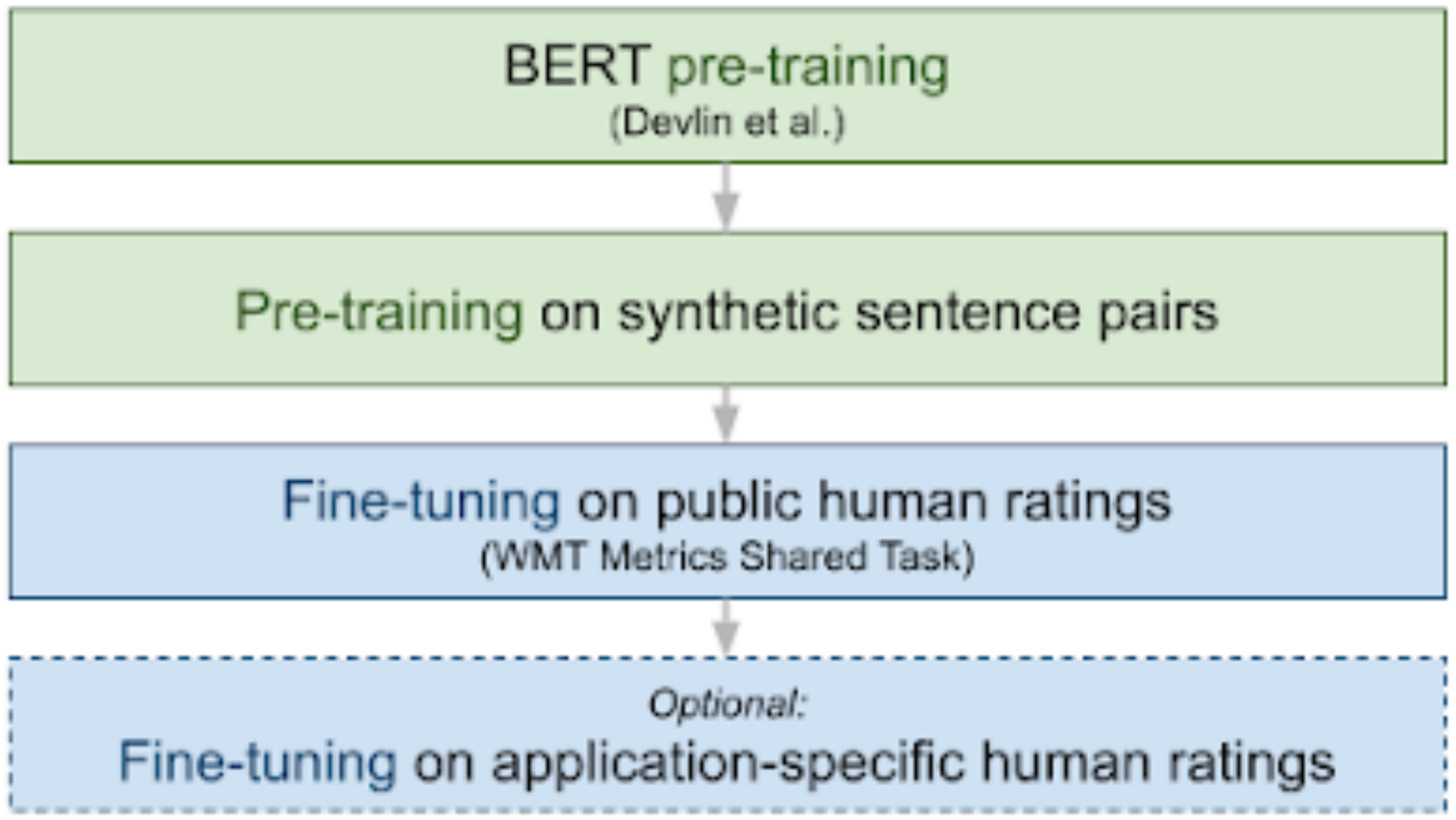
Model-based / Reference-free: BLEURT

- Model predicted human rating



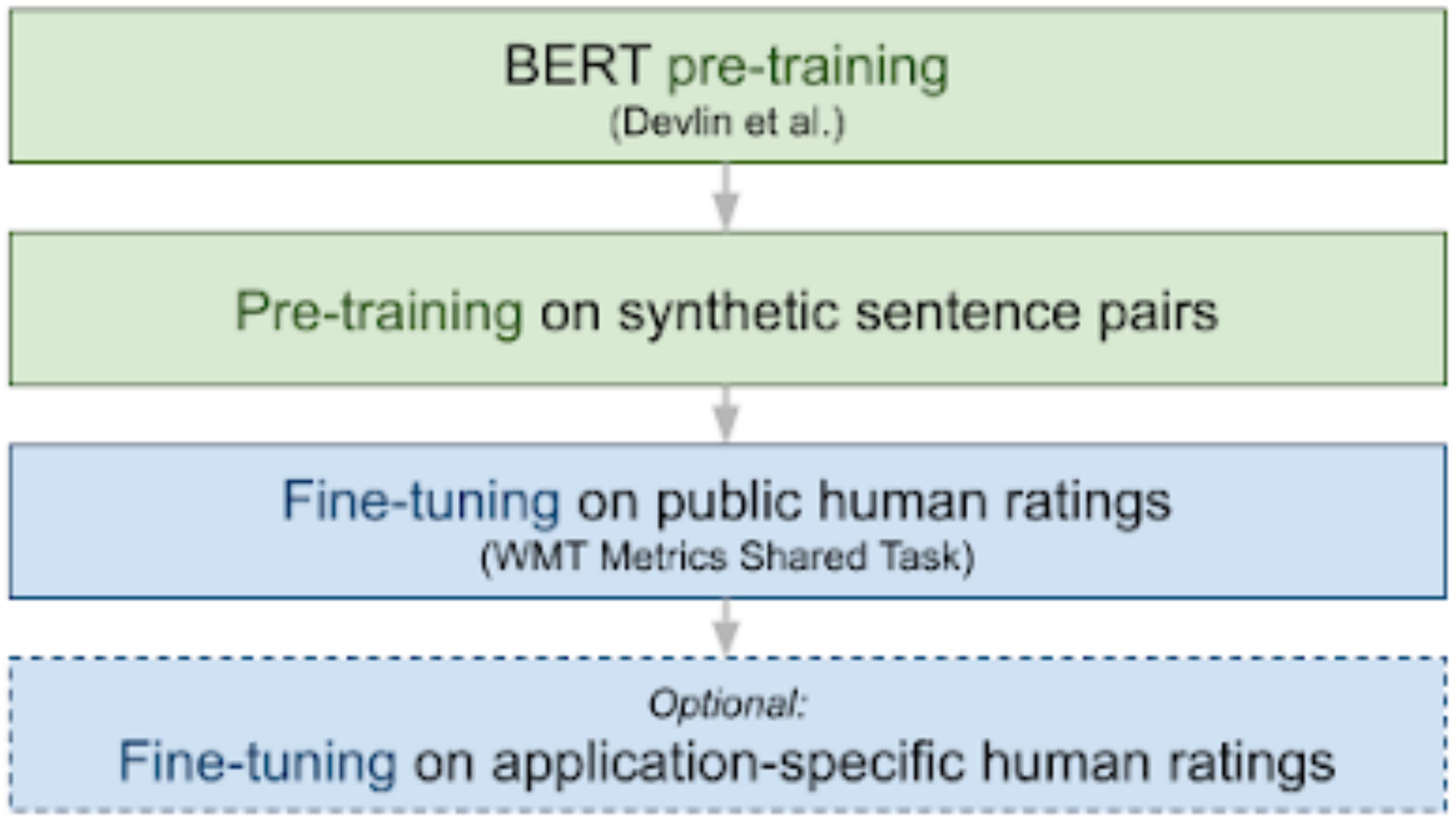
Model-based / Reference-free: BLEURT

- Model predicted human rating
- Train a regression model directly over publicly available human ratings



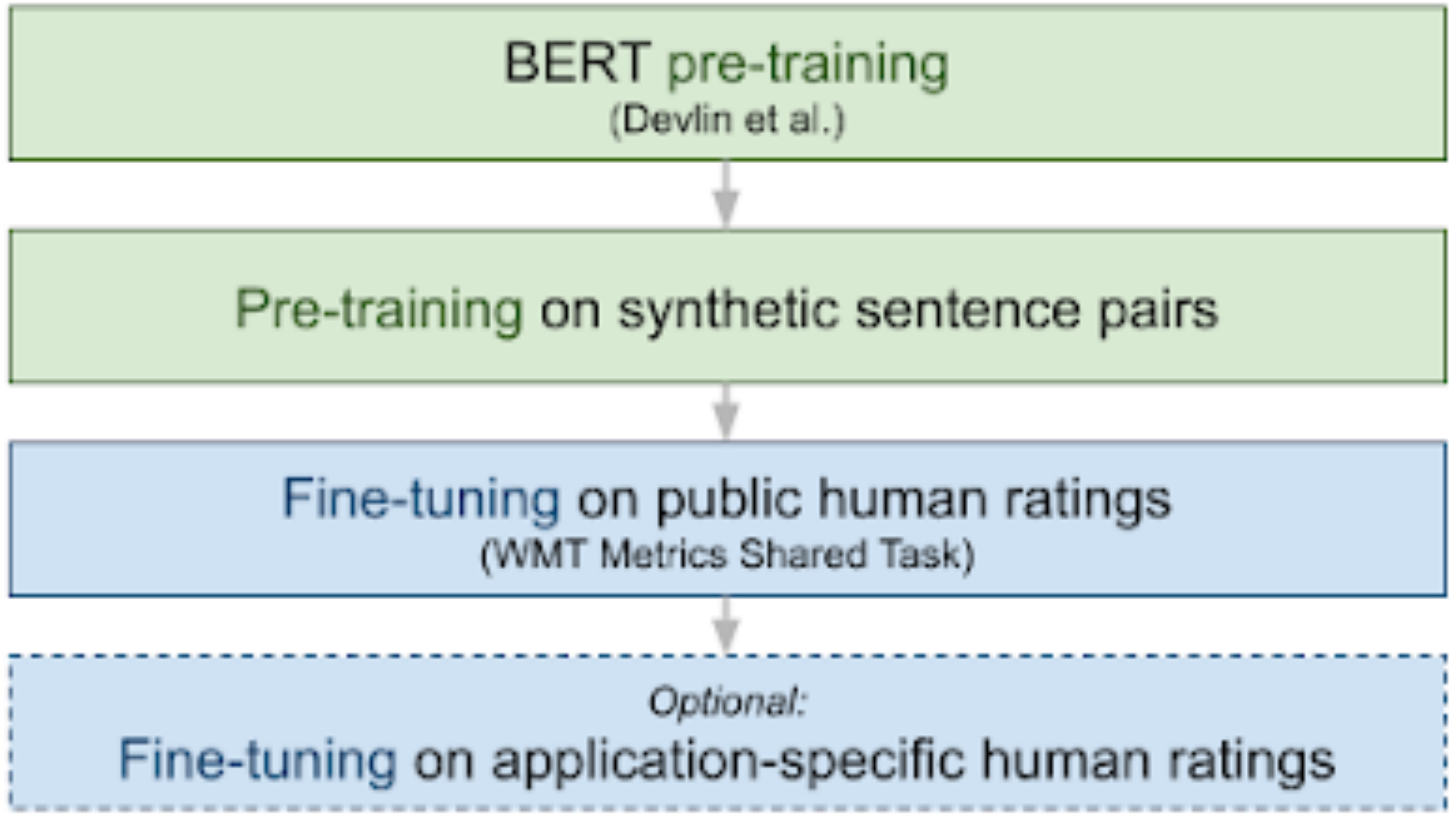
Model-based / Reference-free: BLEURT

- Model predicted human rating
- Train a regression model directly over publicly available human ratings
- References not needed
  - But need human ratings for training
  - Also trained on BLEU and other automatic scores



Model-based / Reference-free: BLEURT

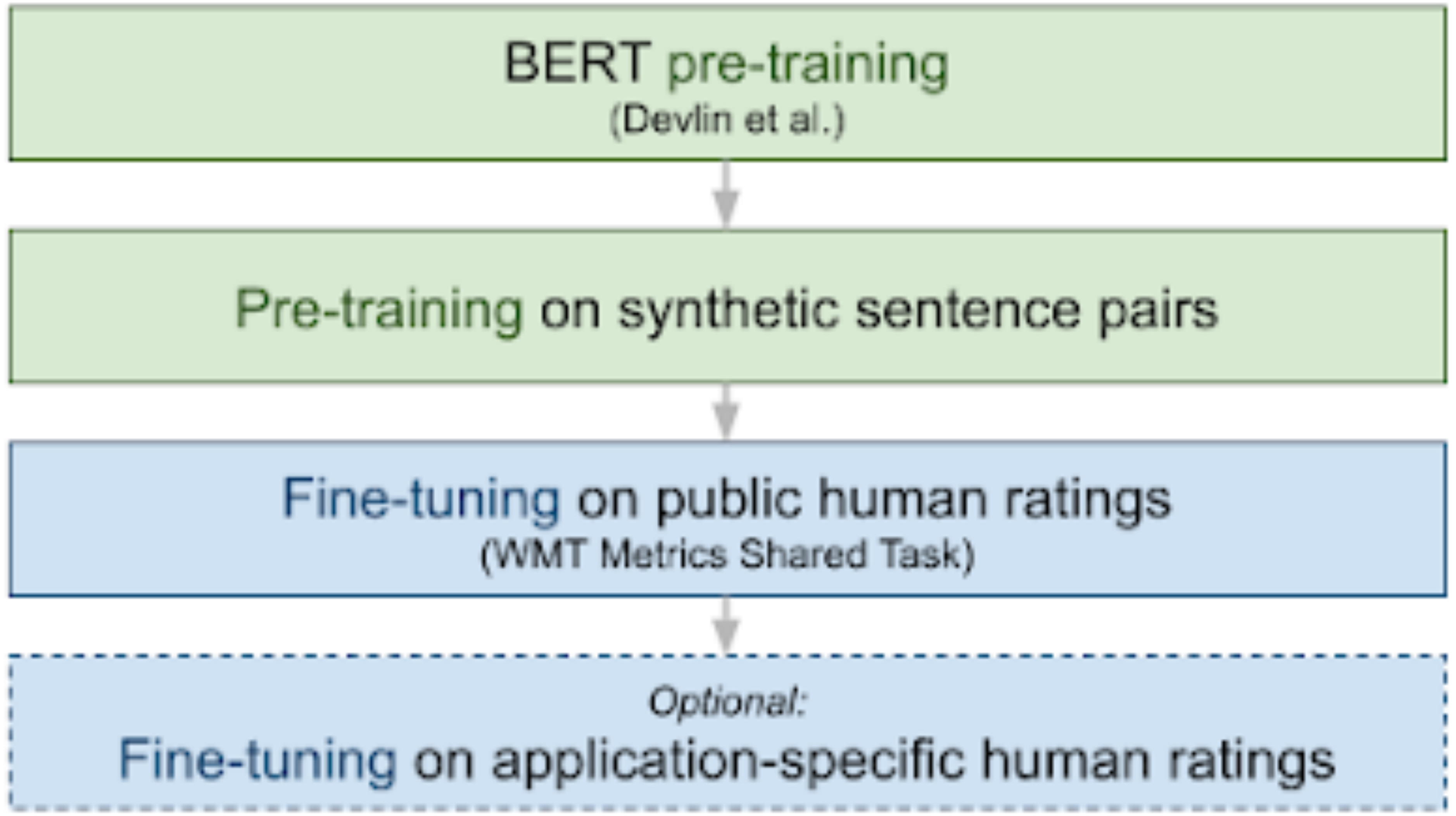
- Model predicted human rating
- Train a regression model directly over publicly available human ratings
- References not needed
    - But need human ratings for training
    - Also trained on BLEU and other automatic scores



Input: Bud Powell était un pianiste de légende. **Reference:** Bud Powell was a legendary pianist.	**BLEURT**
Candidate 1: Bud Powell was a legendary pianist.	*1.01*
Candidate 2: Bud Powell was a historic piano player.	*0.71*
Candidate 3: Bud Powell was a New Yorker.	*-1.49*

Model-based / Reference-free: BLEURT

- Model predicted human rating
- Train a regression model directly over publicly available human ratings
- References not needed
    - But need human ratings for training
    - Also trained on BLEU and other automatic scores



Input: Bud Powell était un pianiste de légende. **Reference:** Bud Powell was a legendary pianist.	**BLEURT**
Candidate 1: Bud Powell was a legendary pianist.	*1.01*
Candidate 2: Bud Powell was a historic piano player.	*0.71*
Candidate 3: Bud Powell was a New Yorker.	*-1.49*

Metric Name	***Kendall Tau w. Human Ratings*** *(mean of all to-English lang. pairs)*
sentenceBLEU	22.7
BERTscore w. BERT-large	30.0
YiSi1 SRL	30.4
ESIM	31.6
BLEURT w. BERT-base	*33.6*
BLEURT w. BERT-large	*33.8*

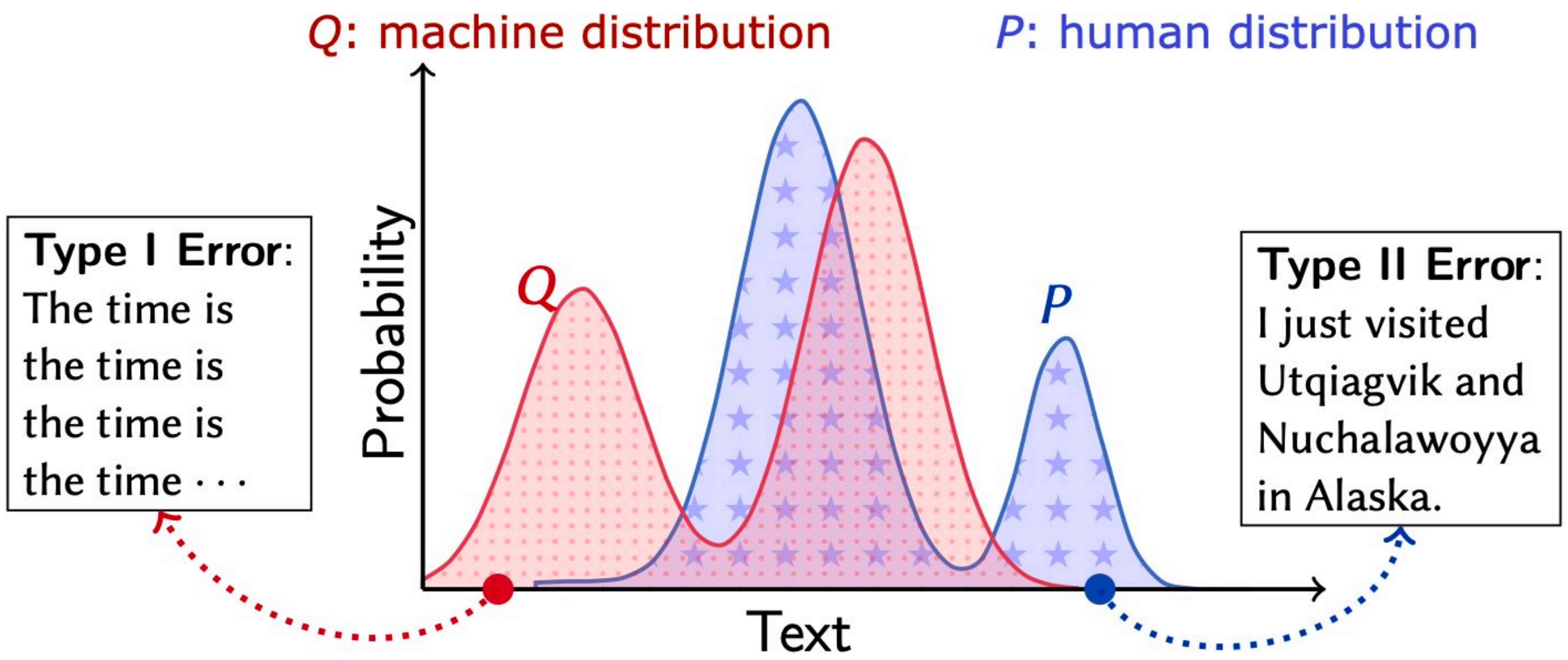
Evaluating Systems without References

- Compare human / natural language distributions to model-generated language distributions
- Divergence between these two distributions can be measured by MAUVE

MAUVE: Measuring the Gap Between Neural Text and Human Text using Divergence Frontiers

Krishna Pillutla[1] **Swabha Swayamdipta**[2] **Rowan Zellers**[1] **John Thickstun**[3]
Sean Welleck[1,2] **Yejin Choi**[1,2] **Zaid Harchaoui**[4]

[1]Paul G. Allen School of Computer Science & Engineering, University of Washington
[2]Allen Institute for Artificial Intelligence
[3]Department of Computer Science, Stanford University
[4]Department of Statistics, University of Washington



Least Reliable: Automatic Evaluation

AlpacaFarm: A Simulation Framework for Methods that Learn from Human Feedback

Yann Dubois* Stanford
Xuechen Li* Stanford
Rohan Taori* Stanford
Tianyi Zhang* Stanford
Ishaan Gulrajani Stanford
Jimmy Ba University of Toronto
Carlos Guestrin Stanford
Percy Liang Stanford
Tatsunori B. Hashimoto Stanford

Cheap and theoretically consistent with human evaluation. BUT… reliability? Models evaluating their own generations may lead to weird mode collapsing effect

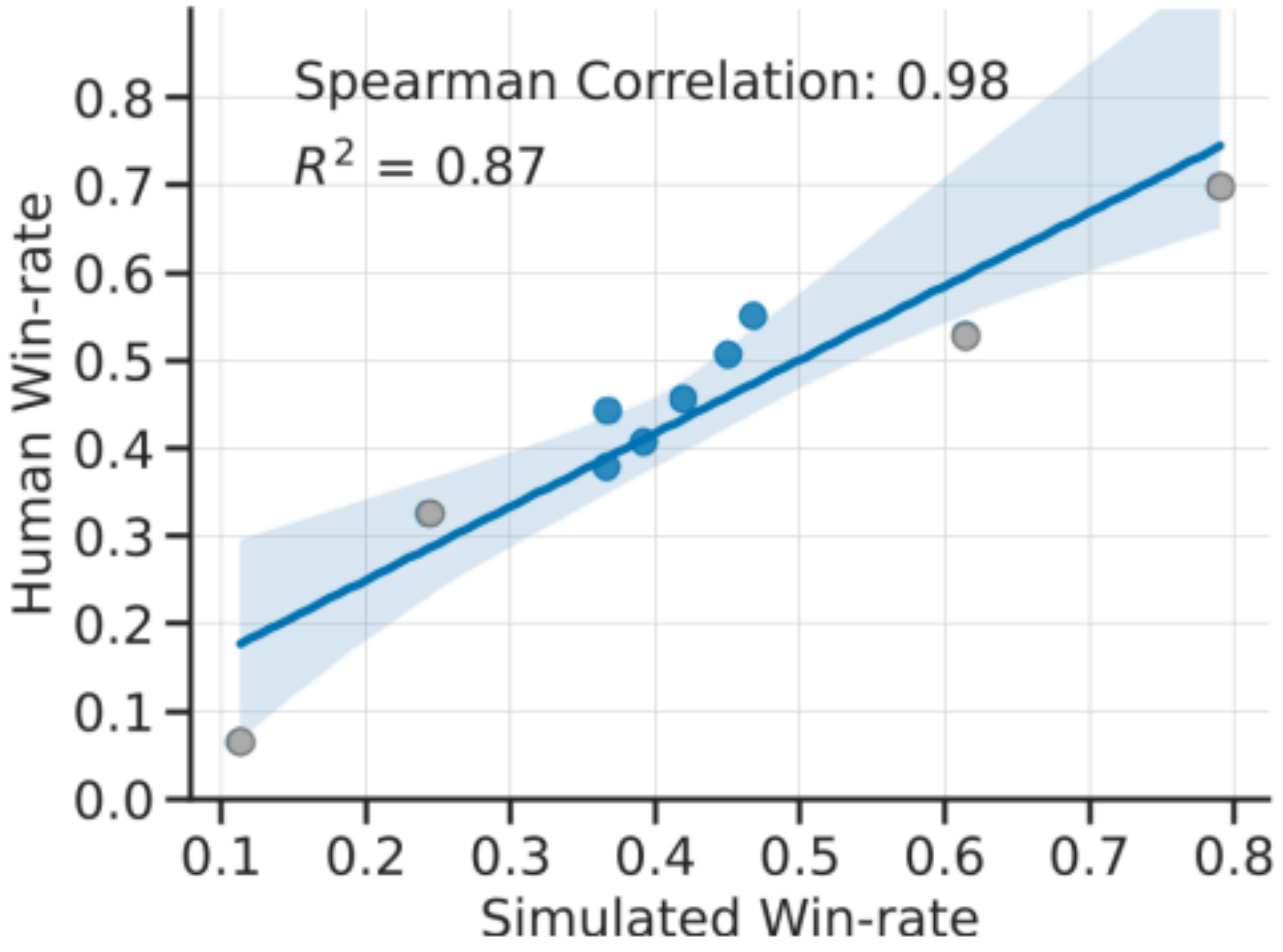


Figure 3: The ranking of methods trained and evaluated in AlpacaFarm matches that of methods trained and evaluated in the human-based pipeline. Each point represents one method M (e.g. PPO). The x-axis shows the simulated evaluation (win-rates measured by $p_{\text{sim}}^{\text{eval}}$) on methods trained in simulation M_{sim}. The y-axis shows human evaluation (win-rates measured by p_{human}) on methods trained with human feedback M_{human}. Gray points show models that we did not train, so their x and y values only differ in the evaluation (simulated vs human). Without those points, we have $R^2 = 0.83$ and a Spearman Correlation of 0.94.

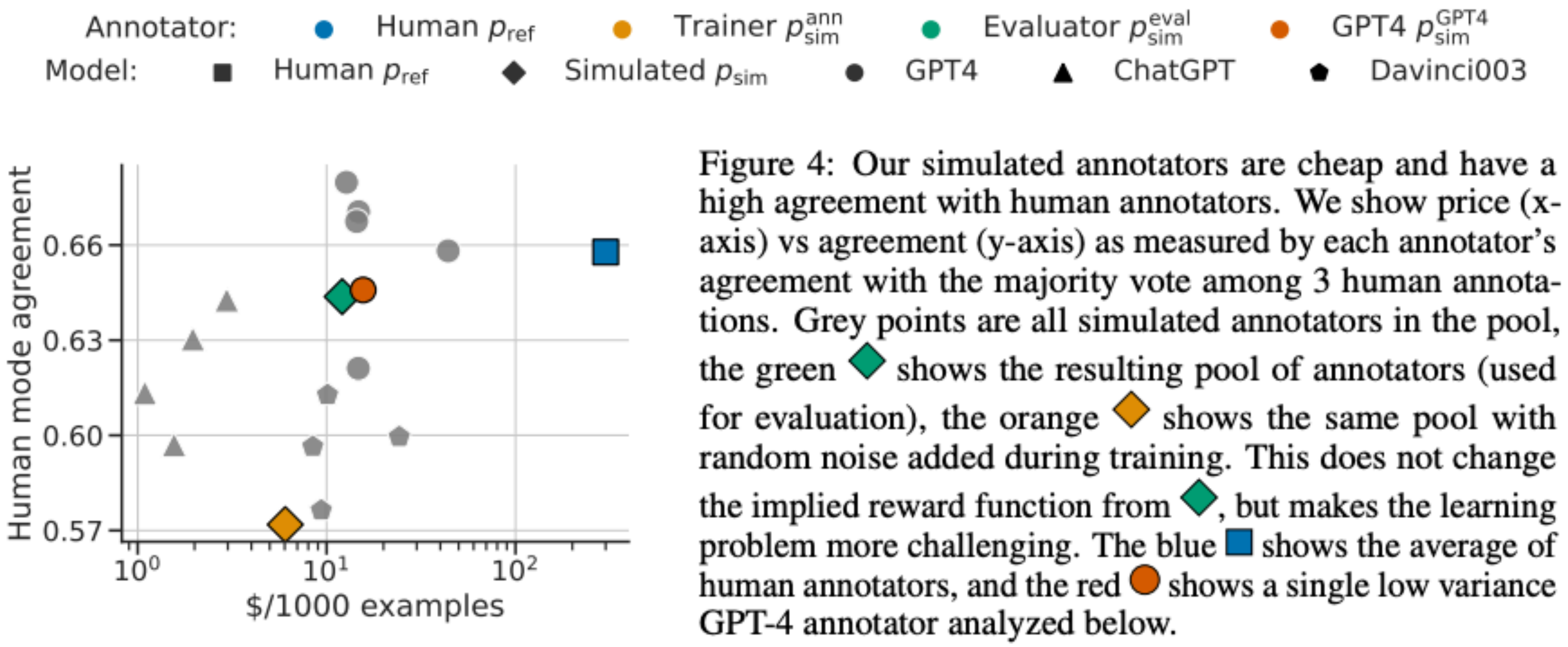


Figure 4: Our simulated annotators are cheap and have a high agreement with human annotators. We show price (x-axis) vs agreement (y-axis) as measured by each annotator's agreement with the majority vote among 3 human annotations. Grey points are all simulated annotators in the pool, the green ◆ shows the resulting pool of annotators (used for evaluation), the orange ◆ shows the same pool with random noise added during training. This does not change the implied reward function from ◆, but makes the learning problem more challenging. The blue ■ shows the average of human annotators, and the red ● shows a single low variance GPT-4 annotator analyzed below.

Natural Language Generation: Parting Thoughts

Natural Language Generation: Parting Thoughts

- Once trained, language models can be very powerful
  - The power only increases with scale
  - So much so that most of our tasks in natural language can be seen as sequence completion tasks
  - Decoding Algorithms thus play a critical role

Natural Language Generation: Parting Thoughts

- Once trained, language models can be very powerful
  - The power only increases with scale
  - So much so that most of our tasks in natural language can be seen as sequence completion tasks
  - Decoding Algorithms thus play a critical role
- **Prompting (or In-Context / Few-Shot Learning):** the ability to do many tasks with no gradient updates and no / a few examples, by simply:
  - Specifying the right sequence prediction problem
  - You can get interesting zero-shot behavior if you're creative enough with how you specify your task!

Recap:
Generation, Prompting and Instruction Tuning of LLMs

USC Viterbi

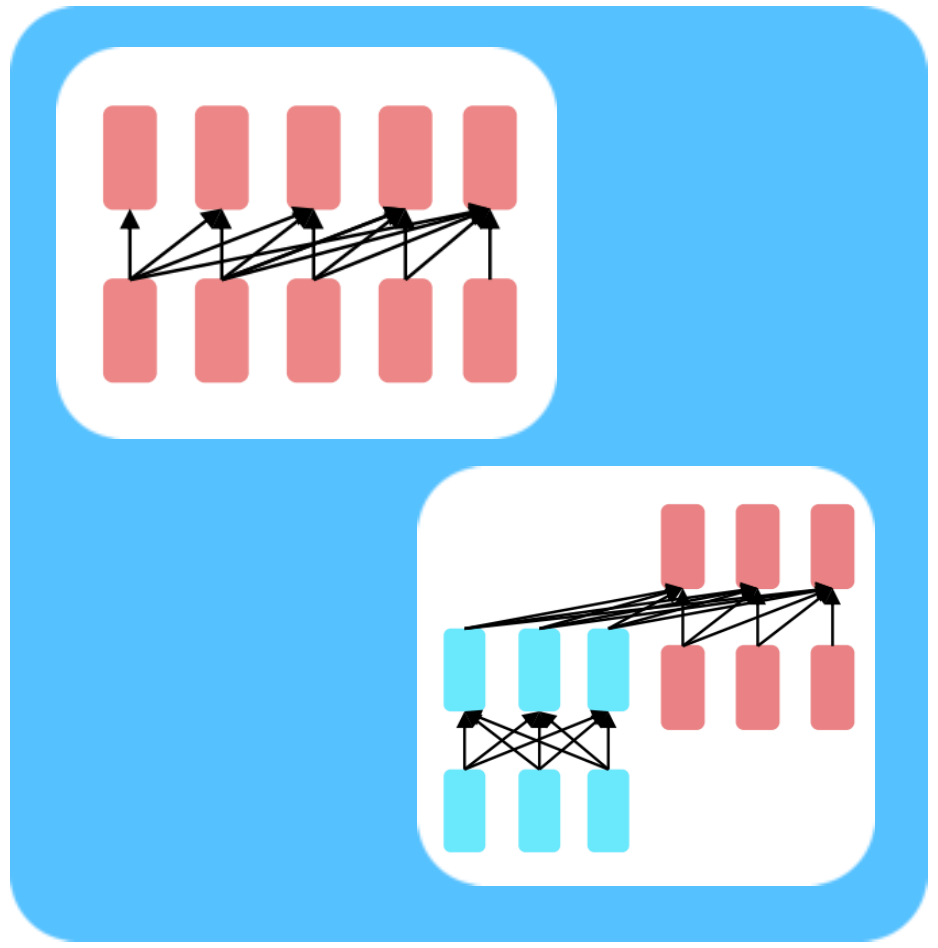
Context Lengths

- GPT-2 has a context length of 1024 tokens

OpenAI model's version	GPT-3 (ada, babbage, curie, davinci)	GPT-3.5 (gpt-3.5-turbo, gpt-3.5-turbo-0301,text-davinci-003, text-davinci-002)*	GPT-4-8K
Context length (max request)	2,049	4,096	8,192
Number of English words	~1,500	~3,000	~6,000
Number of single-spaced pages of English text	3	6	12

Source: Neoteric

Where do prompts / instructions fit in?

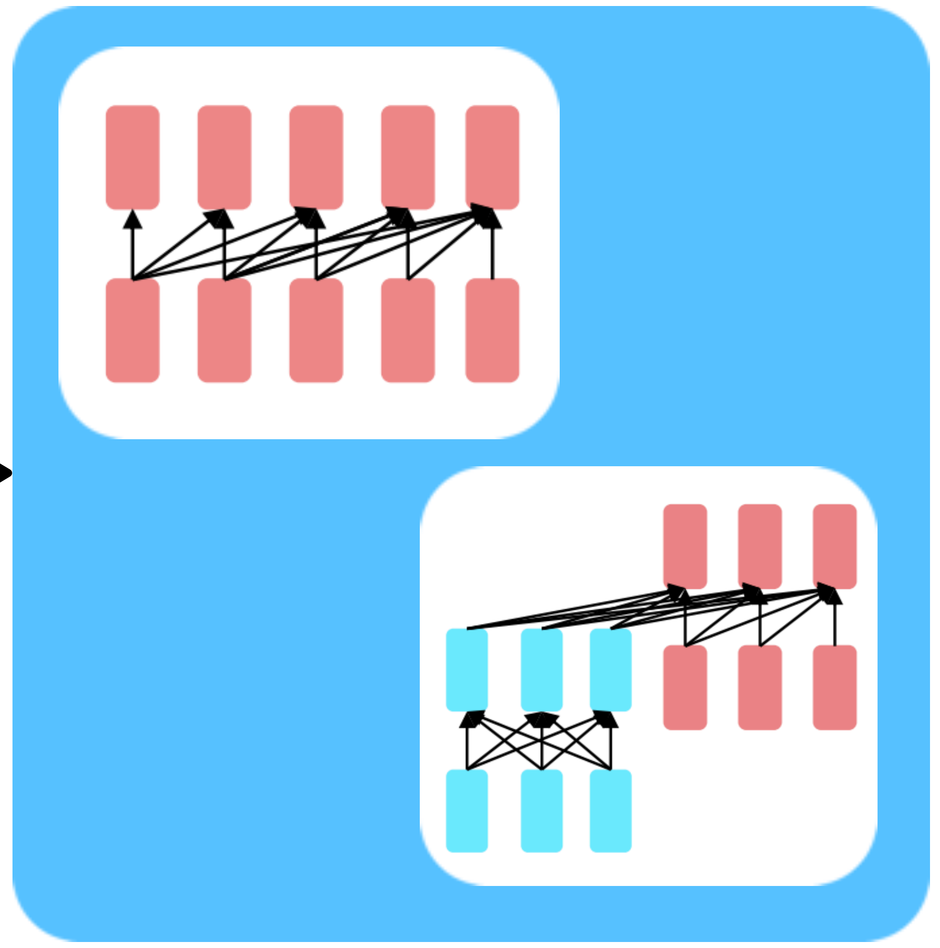


Pretrained Language Model

Where do prompts / instructions fit in?

Input

Prompt = Task Description
+ n examples (shots)

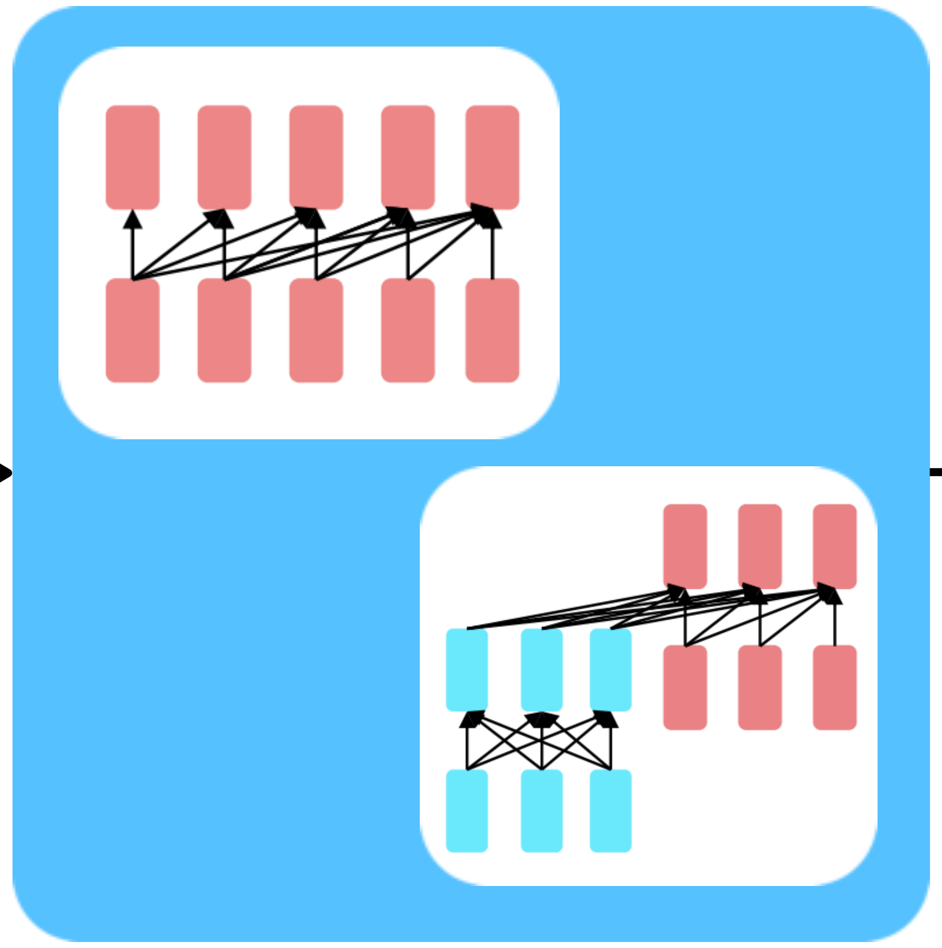


Pretrained Language Model

Where do prompts / instructions fit in?

Input

Prompt = Task Description
+ n examples (shots)



Pretrained Language Model

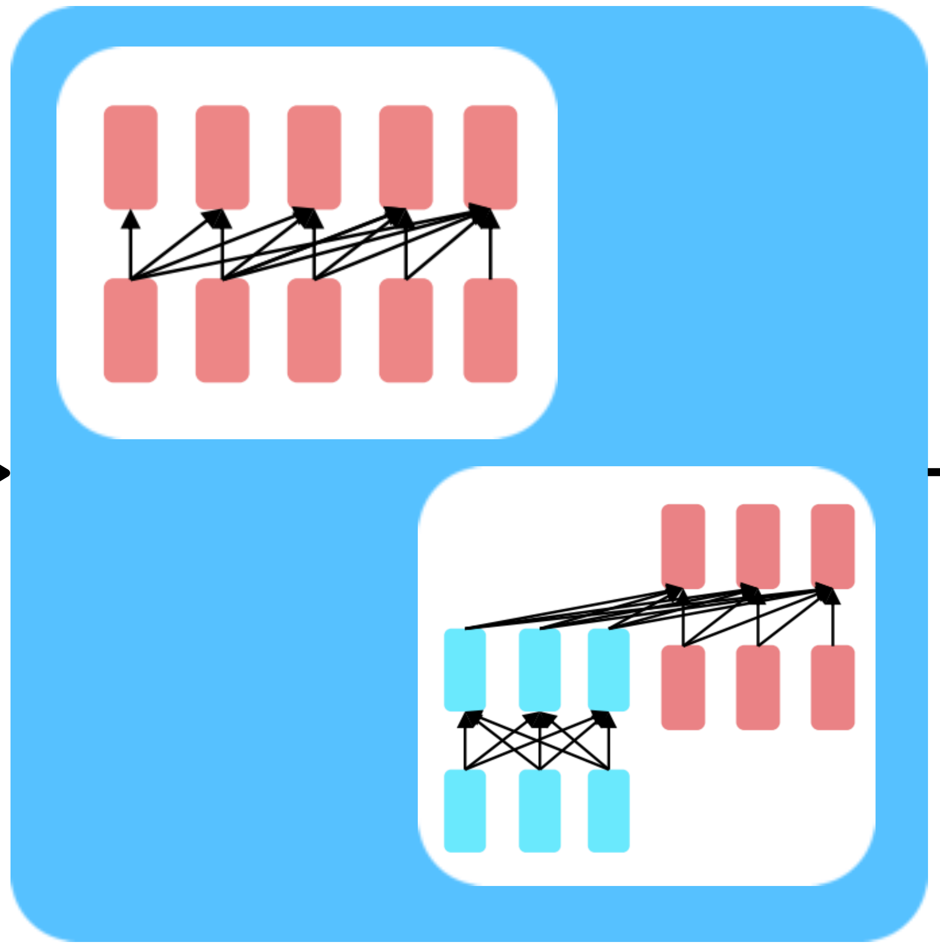
Output

Desired Generation

Where do prompts / instructions fit in?

Input

Prompt = Task Description
+ n examples (shots)



Pretrained Language Model

Decoding Algorithm

Output

Desired Generation

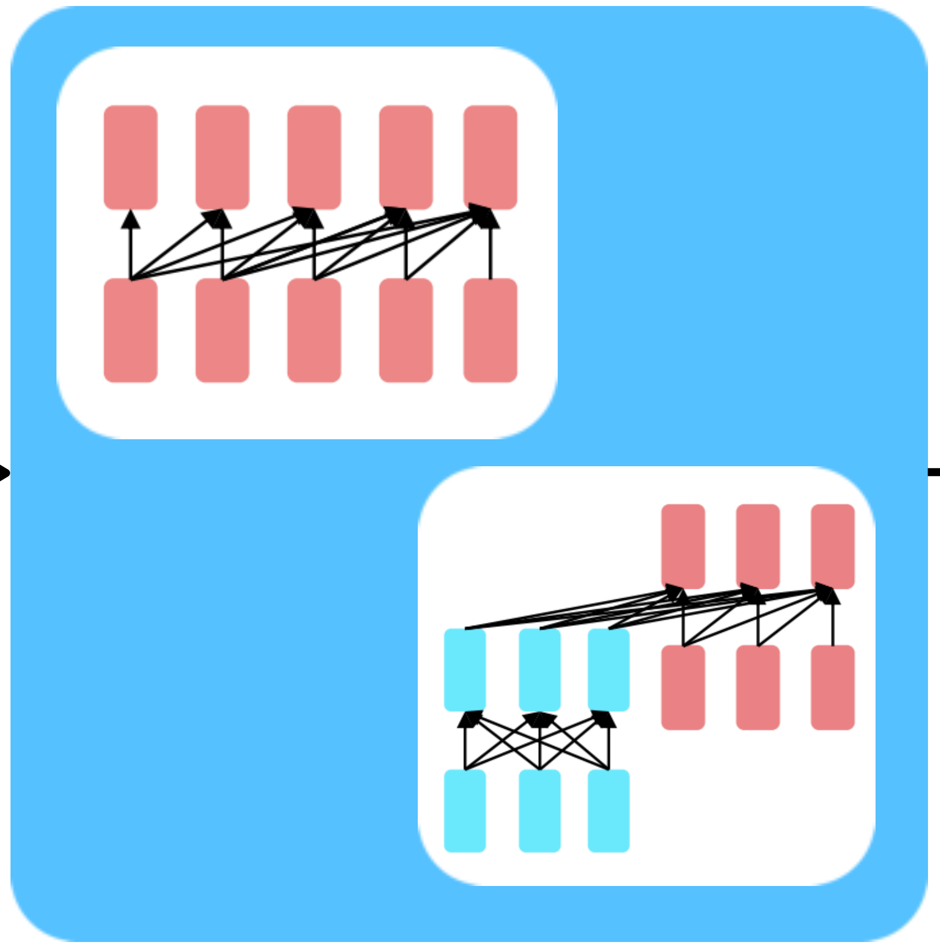
Where do prompts / instructions fit in?

Way to interact with the language model

Input

Prompt = Task Description
+ n examples (shots)

Prompt Engineering



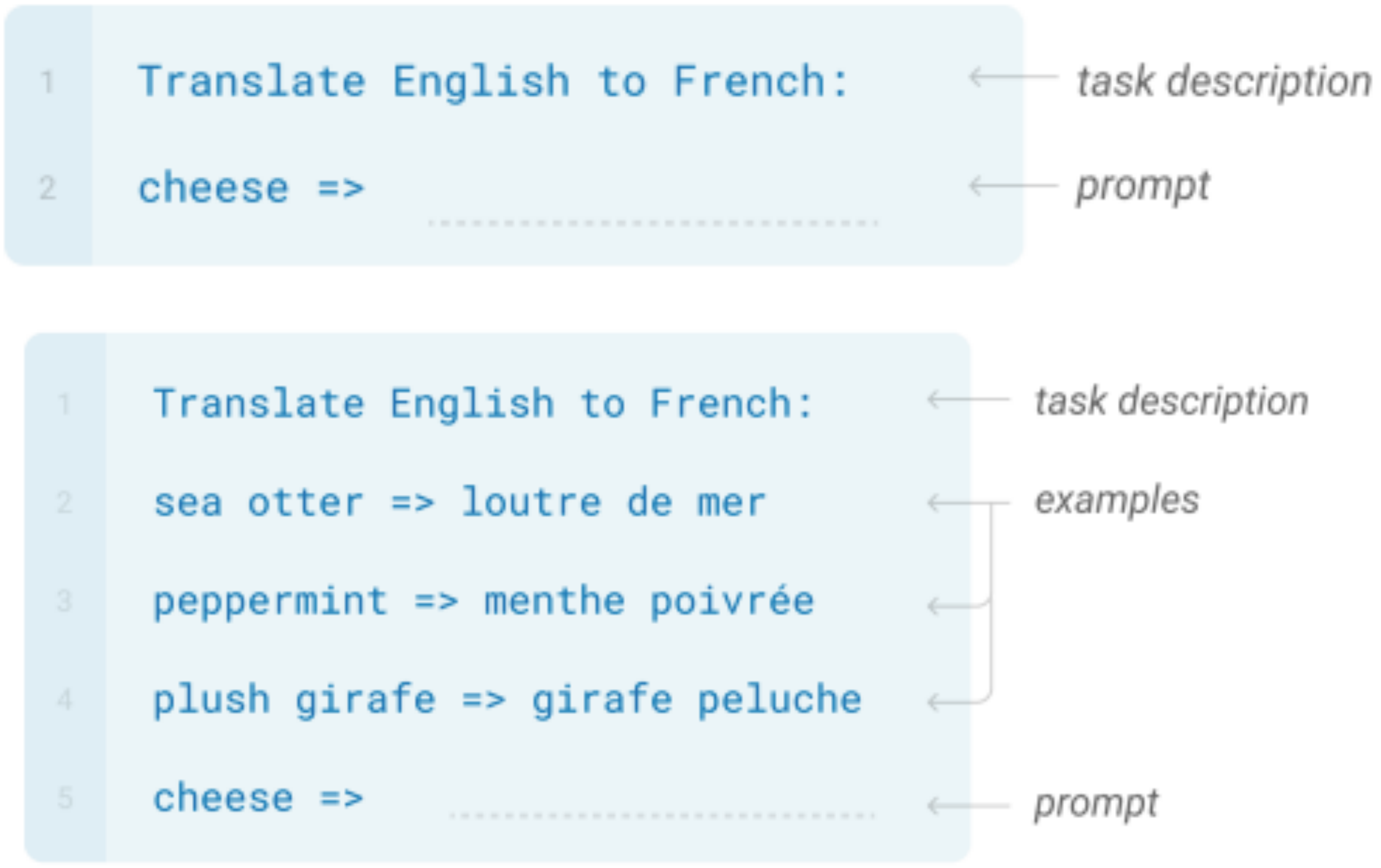
Pretrained Language Model

Decoding
Algorithm

Output

Desired Generation

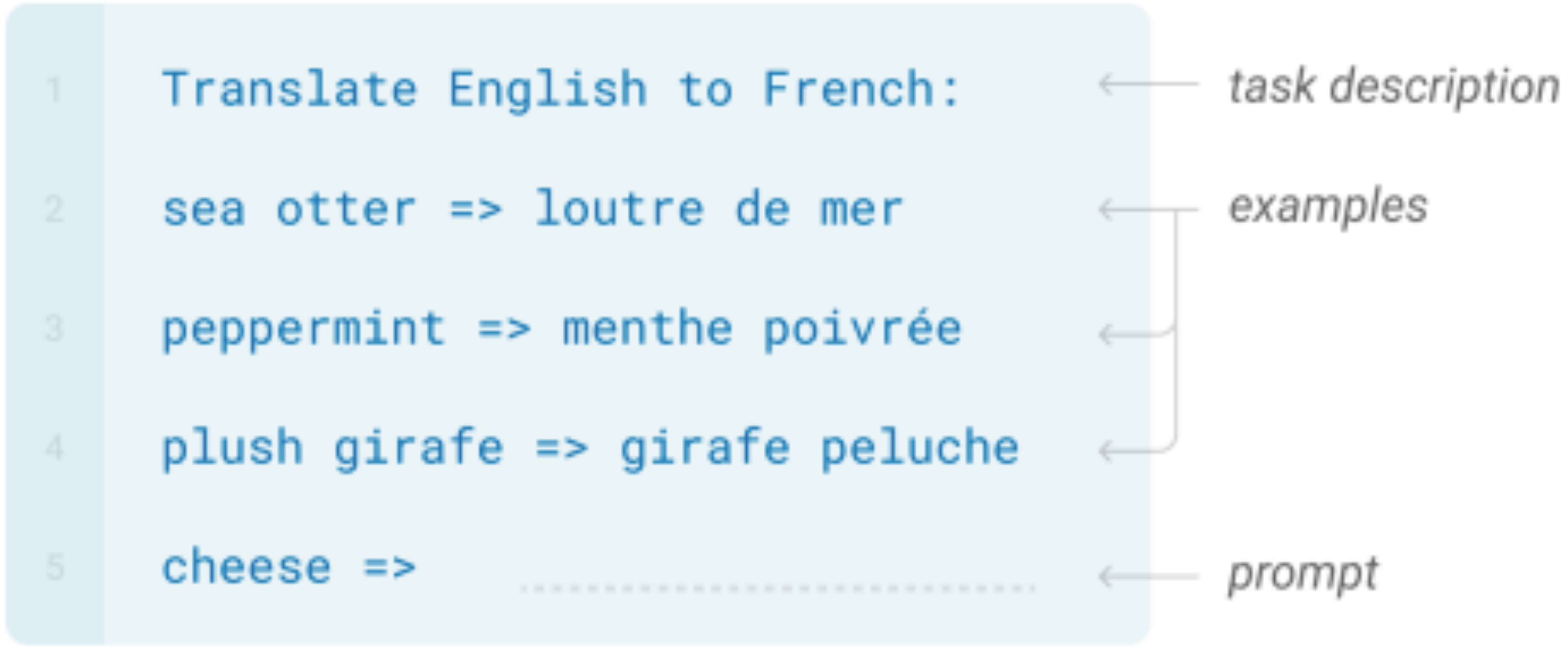
Prompting



Language Models are Few-Shot Learners" (Brown et al., 2020)

Prompting

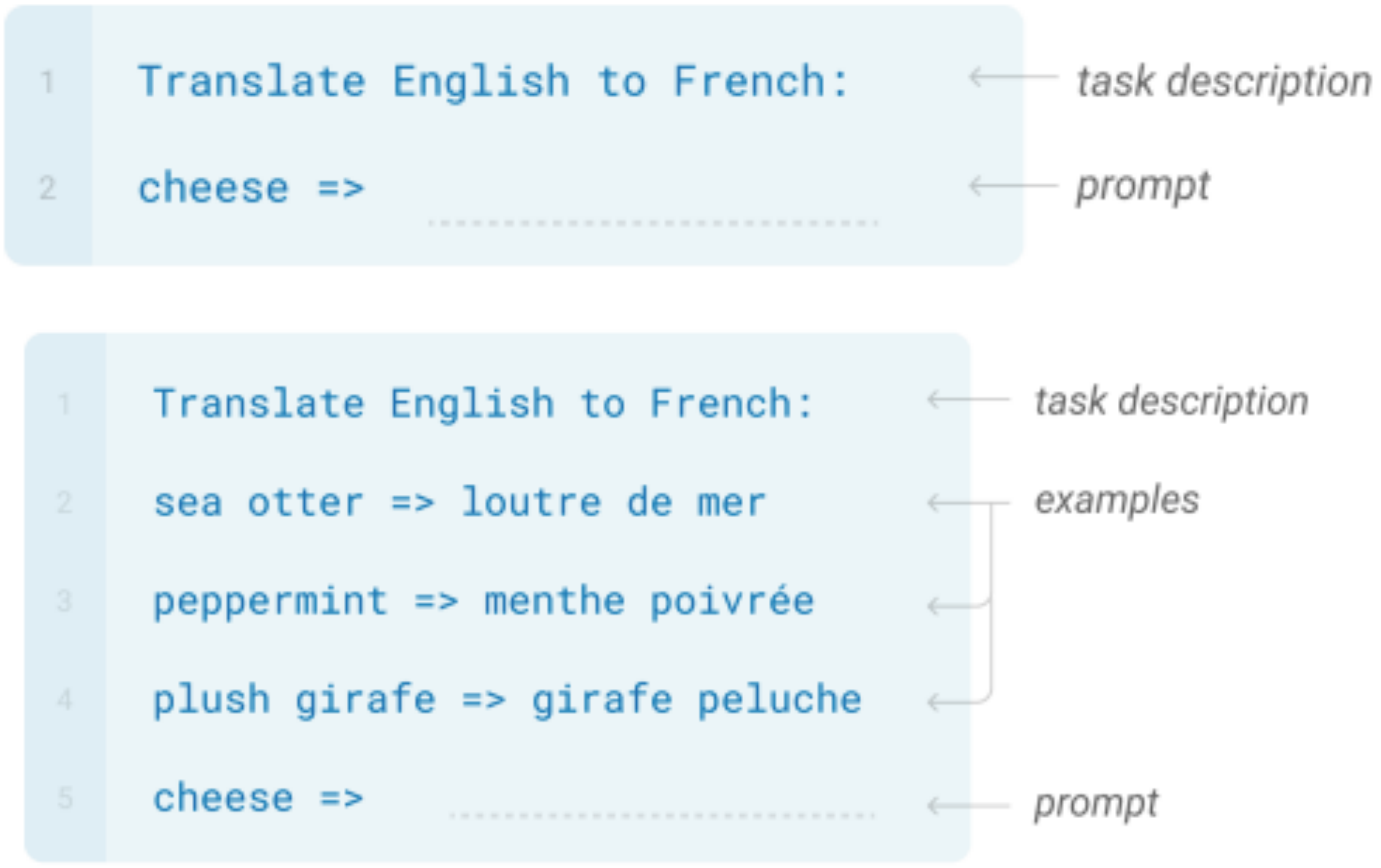
- Zero-shot or few-shot
  - 0-shot: task description + test input
  - n-shot: task description + examples (input / output pairs) + test input
    - n is small, typically less than 10



Language Models are Few-Shot Learners" (Brown et al., 2020)

Prompting

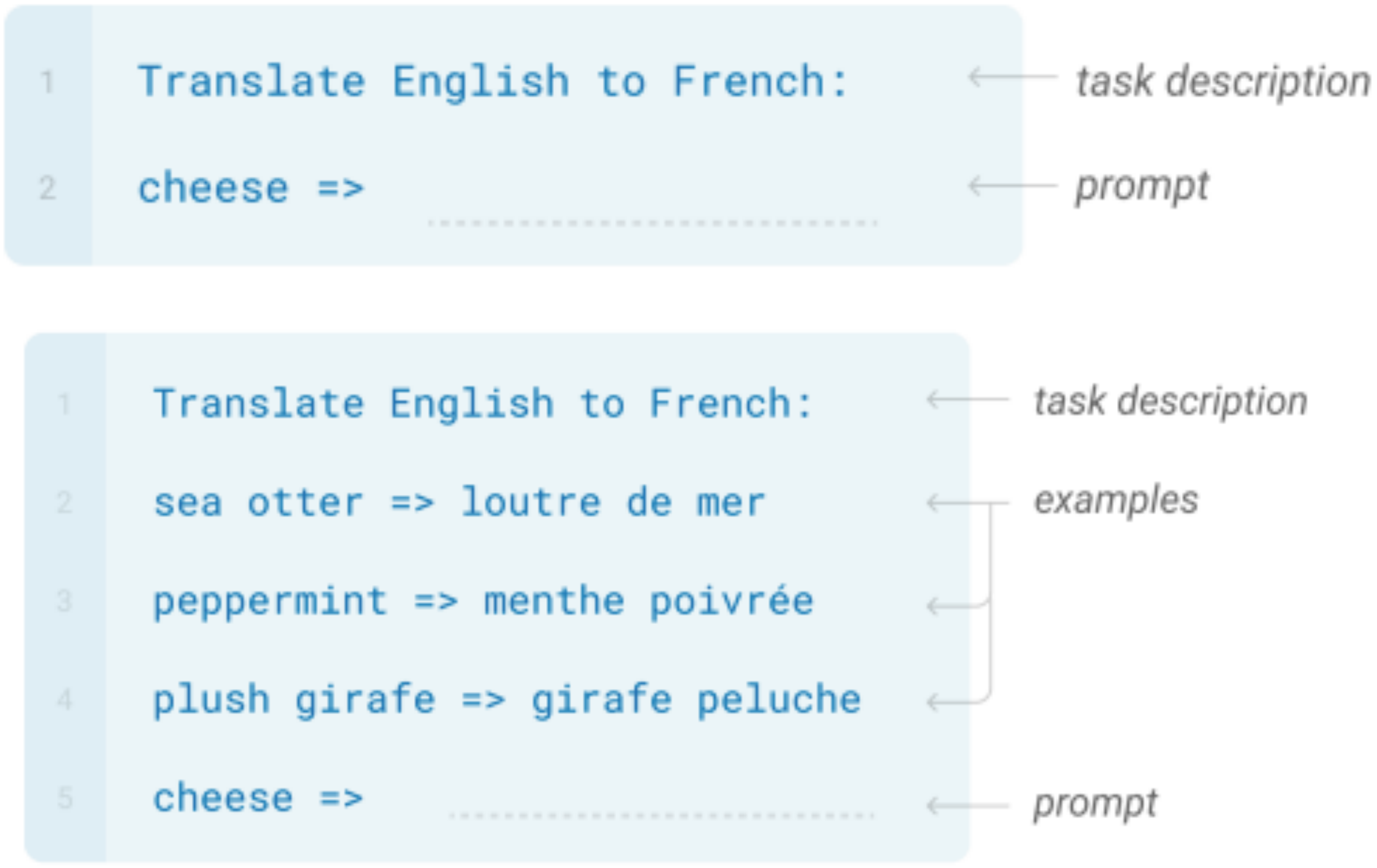
- Zero-shot or few-shot
  - 0-shot: task description + test input
  - n-shot: task description + examples (input / output pairs) + test input
    - n is small, typically less than 10
- Prompt Engineering: How to design the best prompts to elicit a desirable response from a language model



Language Models are Few-Shot Learners" (Brown et al., 2020)

Prompting

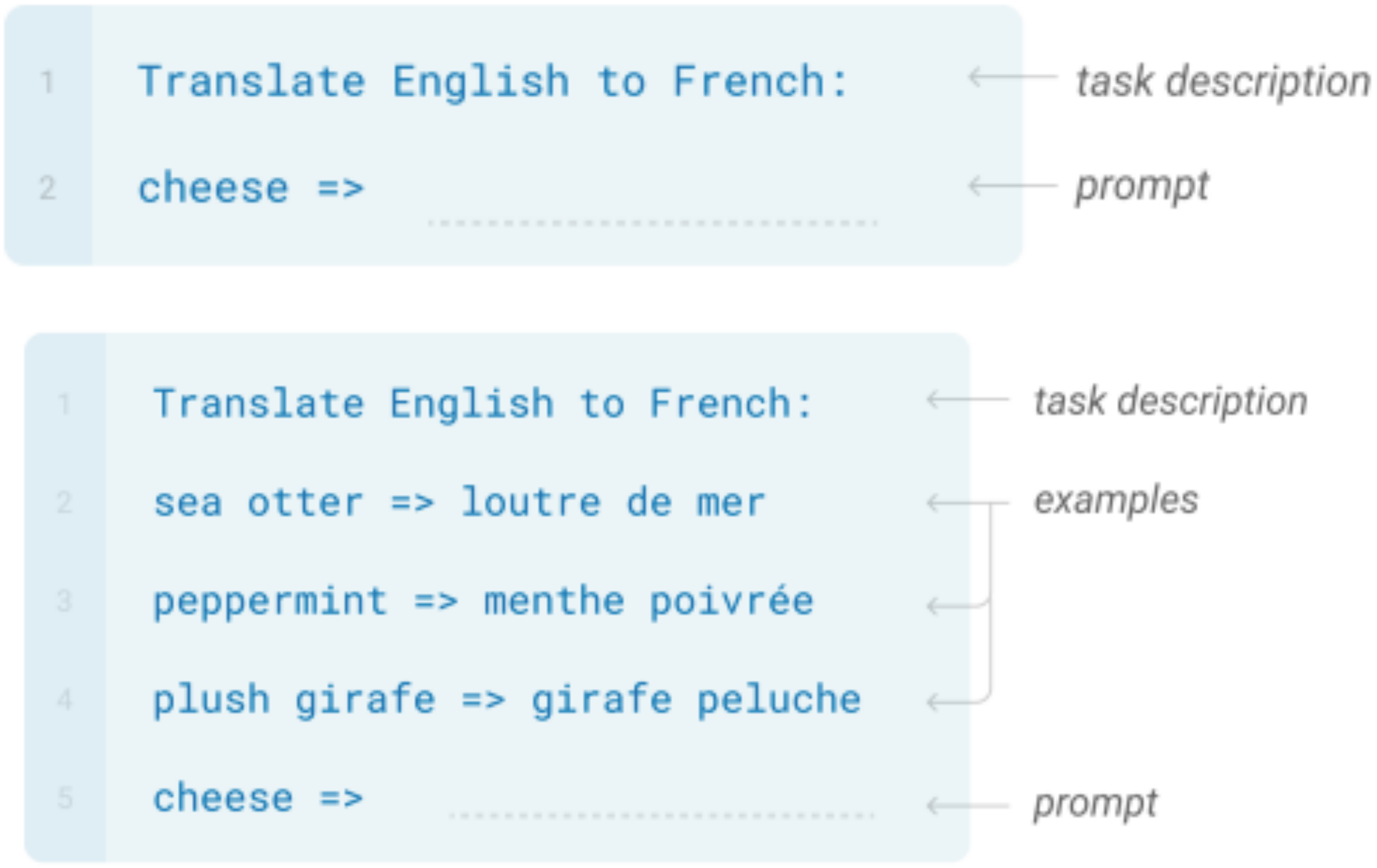
- Zero-shot or few-shot
    - 0-shot: task description + test input
    - n-shot: task description + examples (input / output pairs) + test input
        - n is small, typically less than 10
- Prompt Engineering: How to design the best prompts to elicit a desirable response from a language model
- Different styles with differing amounts of granularity:
    - Chain-of-thought
    - Tree-of-thought
    - etc.



Language Models are Few-Shot Learners" (Brown et al., 2020)

Prompting

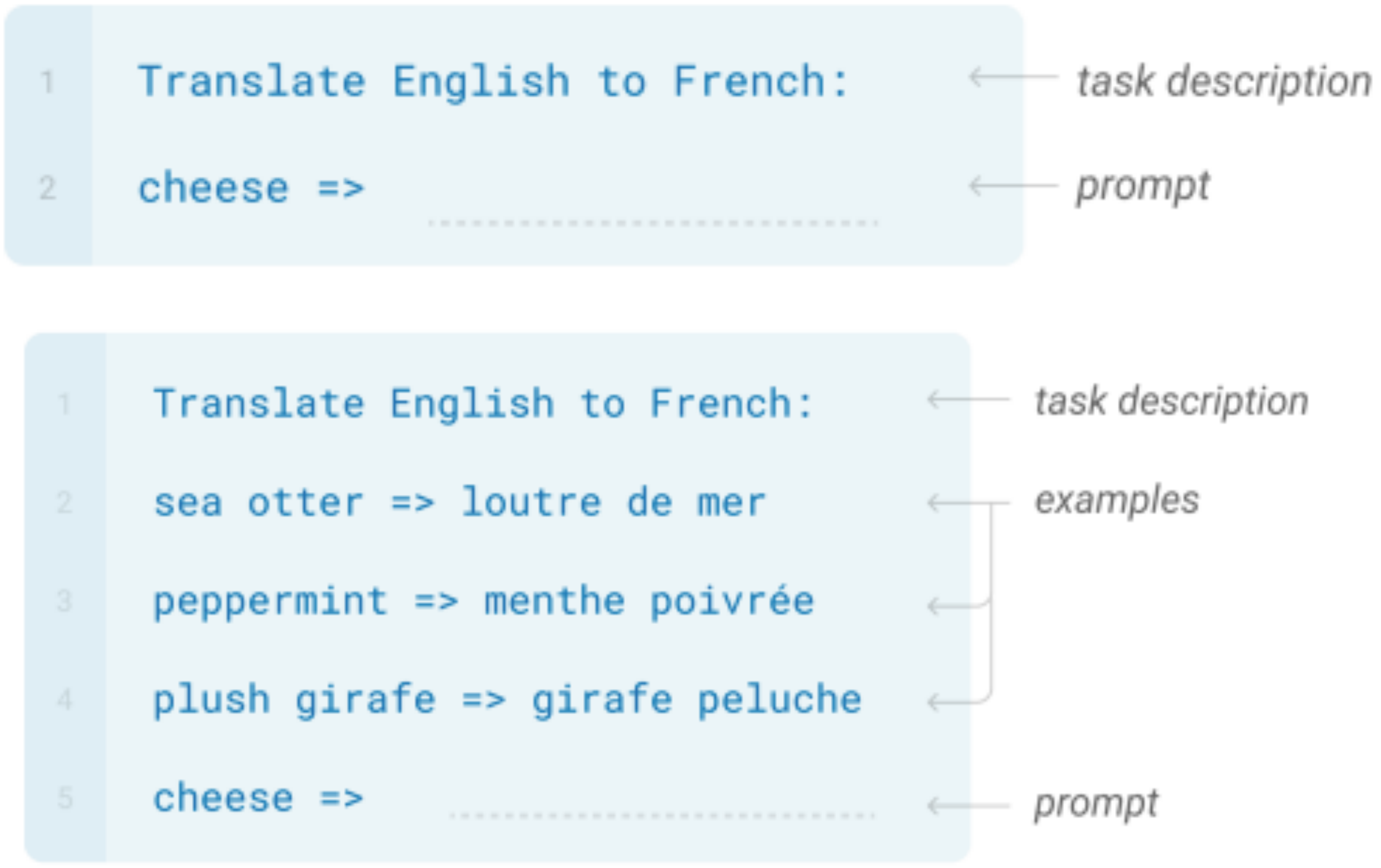
- Zero-shot or few-shot
  - 0-shot: task description + test input
  - n-shot: task description + examples (input / output pairs) + test input
    - n is small, typically less than 10
- Prompt Engineering: How to design the best prompts to elicit a desirable response from a language model
- Different styles with differing amounts of granularity:
  - Chain-of-thought
  - Tree-of-thought
  - etc.
- Limitations: not an exact science (trial and error driven), reproducibility



Language Models are Few-Shot Learners” (Brown et al., 2020)

Prompting

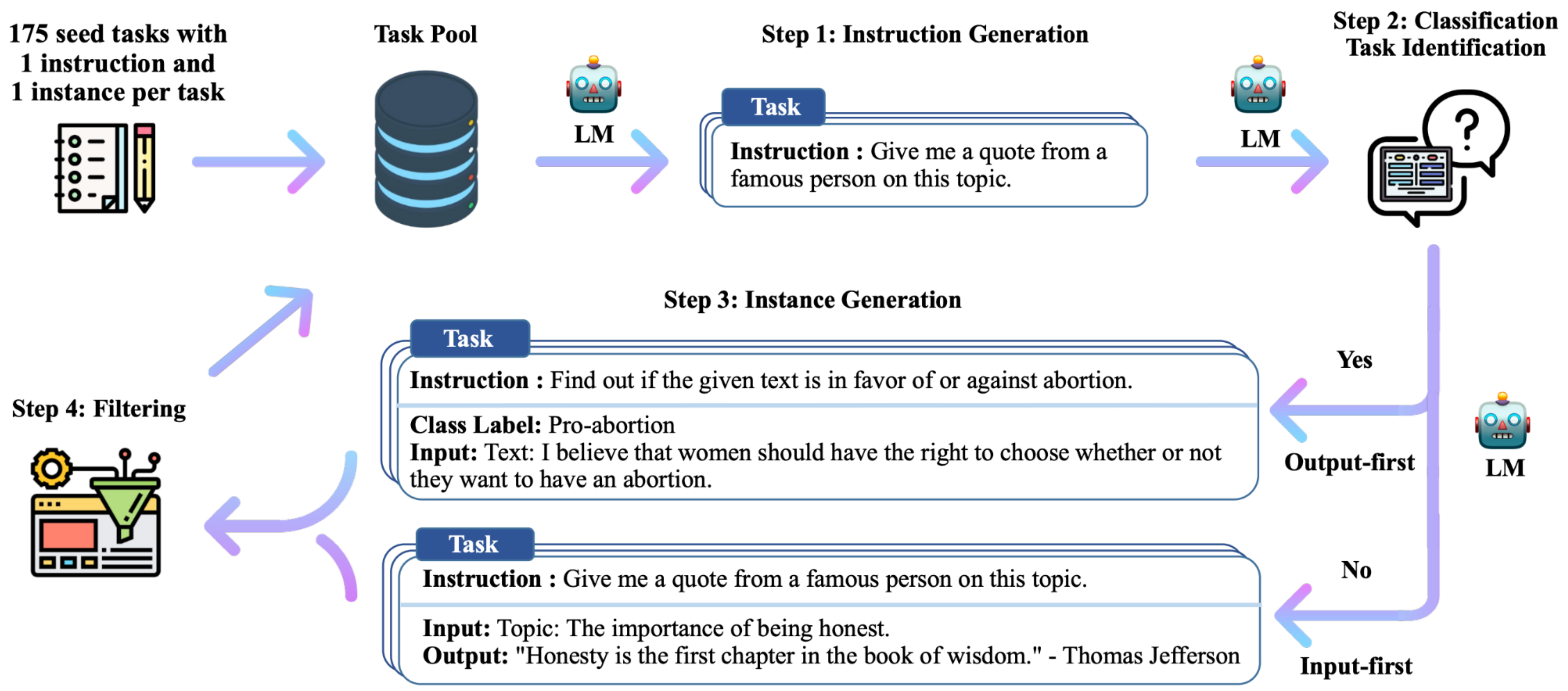
- Zero-shot or few-shot
  - 0-shot: task description + test input
  - n-shot: task description + examples (input / output pairs) + test input
    - n is small, typically less than 10
- Prompt Engineering: How to design the best prompts to elicit a desirable response from a language model
- Different styles with differing amounts of granularity:
  - Chain-of-thought
  - Tree-of-thought
  - etc.
- Limitations: not an exact science (trial and error driven), reproducibility
- Recent efforts to automate prompt engineering / prompt tuning



Language Models are Few-Shot Learners” (Brown et al., 2020)

Instruction-Tuning

- Even prompting includes an instruction (description of the task)
    - But done more explicitly in instruction tuning
    - **Key difference**: **Parameter Updates**
- Modern approaches: uses adapter models!
    - Adapters (LORA): mini layers between LM components with updatable parameters
    - All other parameters stay the same.
- Much more robust than prompt engineering
- Involves supervised fine-tuning
    - Convert each task into a linguistic sequence



"Self-Instruct: Aligning Language Models with Self-Generated Instructions" (Wang et al., 2023)

More on LLM Adaptation

Prompting

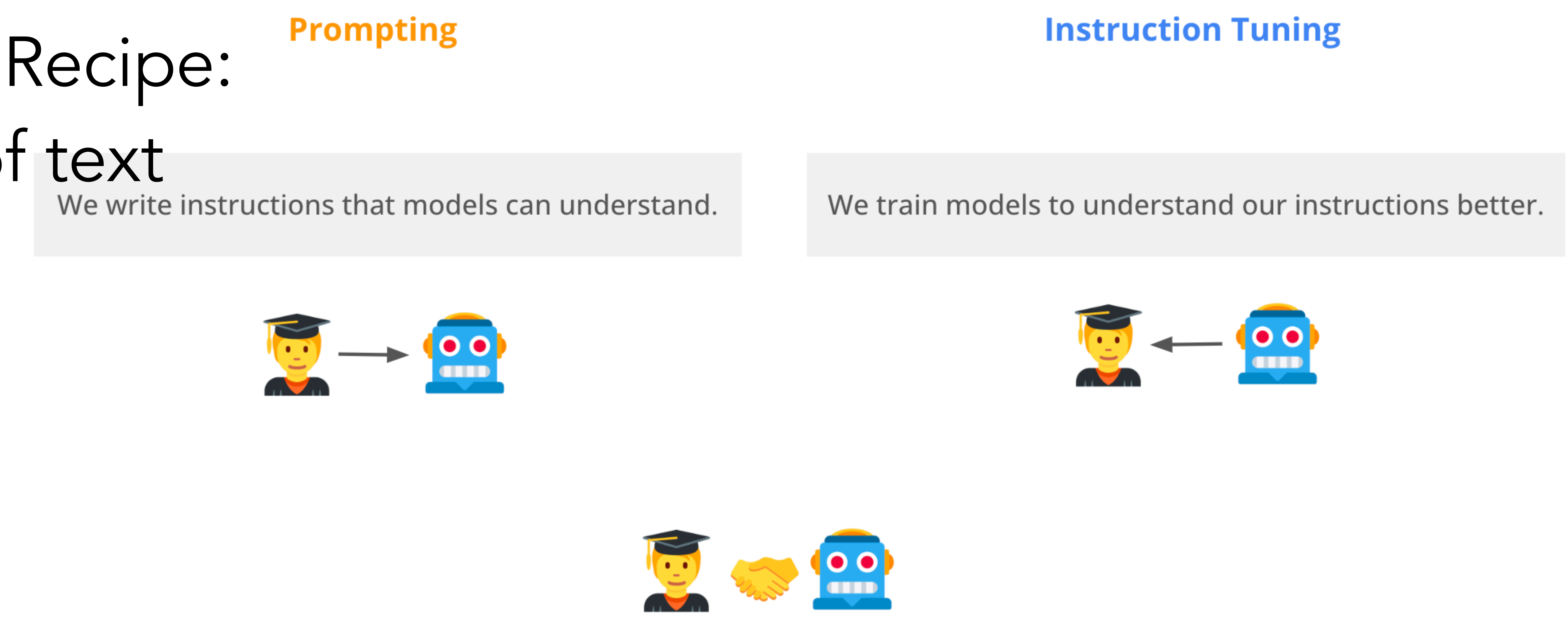
We write instructions that models can understand.

Instruction Tuning

We train models to understand our instructions better.

More on LLM Adaptation

- Modern LLMs (GPT-3.5 and later) Training Recipe:
  - Stage 1: Pre-training on large corpus of text



More on LLM Adaptation

- Modern LLMs (GPT-3.5 and later) Training Recipe:
    - Stage 1: Pre-training on large corpus of text
    - Stage 2: Post-training
        - Instruction Tuning (Supervised Finetuning)
    - Stage 3: Post-training and Alignment

Prompting

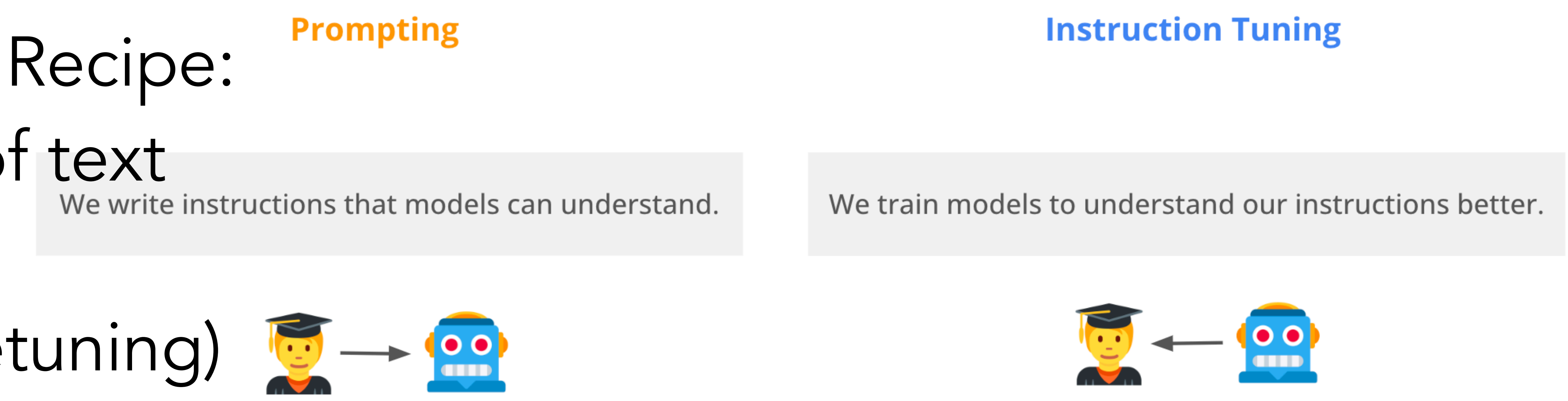
We write instructions that models can understand.

Instruction Tuning

We train models to understand our instructions better.

More on LLM Adaptation

- Modern LLMs (GPT-3.5 and later) Training Recipe:
  - Stage 1: Pre-training on large corpus of text
  - Stage 2: Post-training
    - Instruction Tuning (Supervised Finetuning)
  - Stage 3: Post-training and Alignment
    - Reinforcement Learning with Human Feedback
    - Train a supervised classifier (reward model) on human demonstrations to provide feedback to LM
    - Supervised fine-tuning the LM with reinforcement learning to maximize rewards given by reward model
- Prompting is only valuable after all pre- and post-training steps



Aligning LLMs
by Guest Lecturer, Justin Cho

Reinforcement Learning with Human Feedback

Justin Cho (hd.justincho@gmail.com)

CSCI 499
April 10th, 2024

Reinforcement Learning with Human Feedback (RLHF)

ChatGPT

Gemini

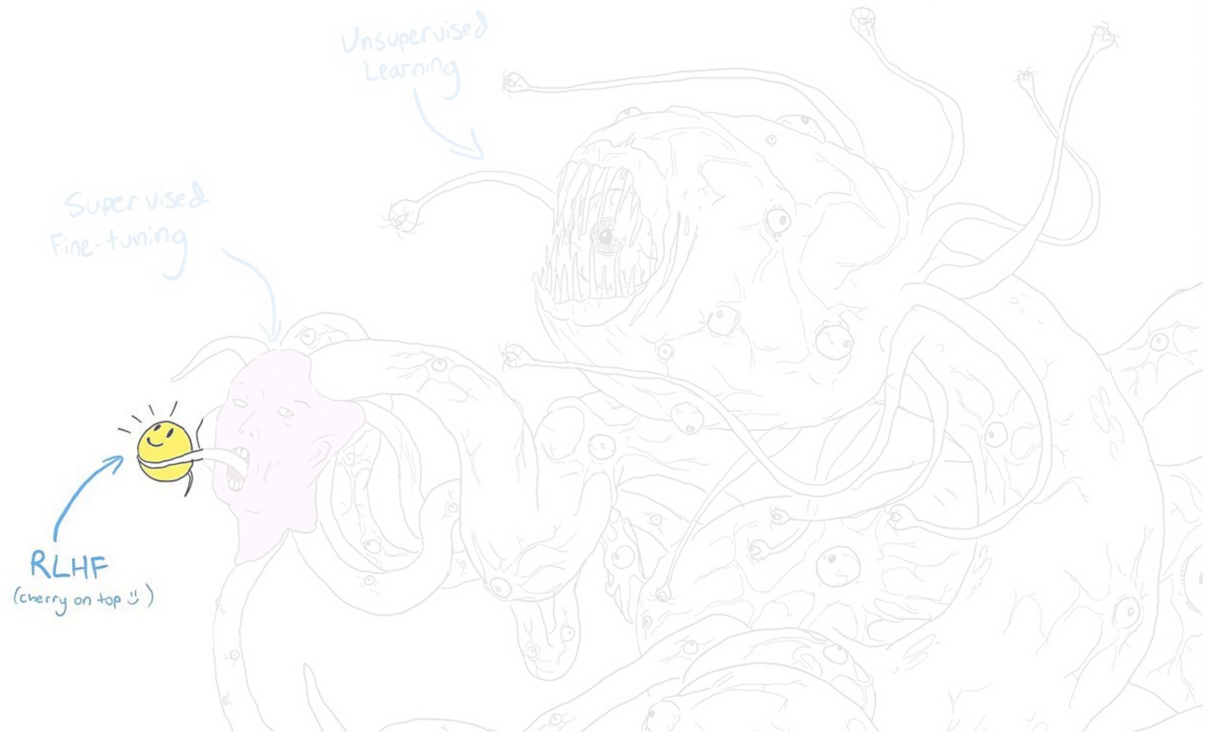
Claude 3

BY ANTHROP\C

RLHF
(cherry on top ☺)

Reinforcement Learning with Human Feedback (RLHF)

- Make language models
  - Palatable
  - User-friendly



Shoggoth with smiley face meme (image credit @anthrupad)

Main goal of this lecture

- For you
  - Get a high-level understanding of the key ingredient that enabled powerful language models like ChatGPT.
    - So that you can sound smart
    - Get interested in NLP research / engineering and contribute to pushing the limits of AI

Main goal of this lecture

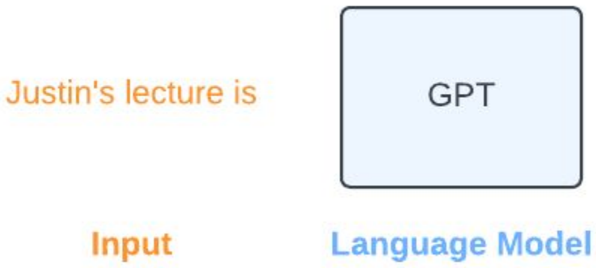
- For you
  - Get a high-level understanding of the key ingredient that enabled powerful language models like ChatGPT.
    - So that you can sound smart
    - Get interested in NLP research / engineering and contribute to pushing the limits of AI
- For me
  - Learn. Best way to learn something is to teach it.
  - Get better at teaching!

Outline

- Motivating RLHF
  - Recap: pretraining & supervised finetuning
  - “Aligning” language models
- What is RLHF?
  - Primer on reinforcement learning
  - Overview of RLHF
    - Prerequisites for RLHF
    - Reward modeling
    - Reinforcement learning
- Why does RLHF work?
- Challenges of RLHF

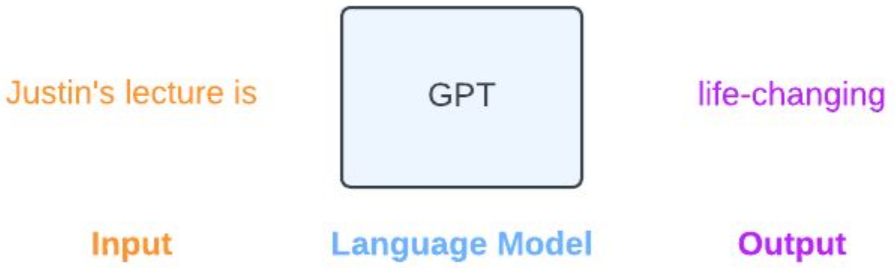
Pretraining

Use unsupervised learning to simply learn how to predict the next work. (autoregressive language modeling)



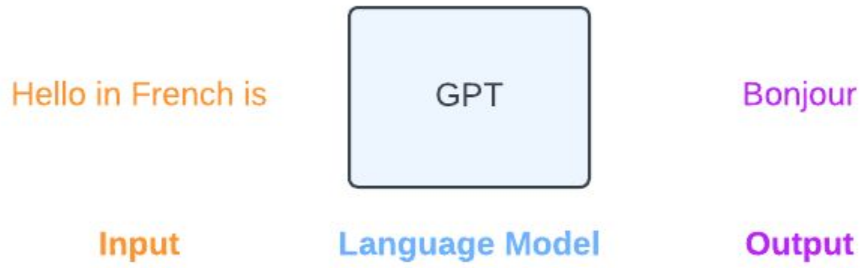
Pretraining

Use unsupervised learning to simply learn how to predict the next word.
(autoregressive language modeling)



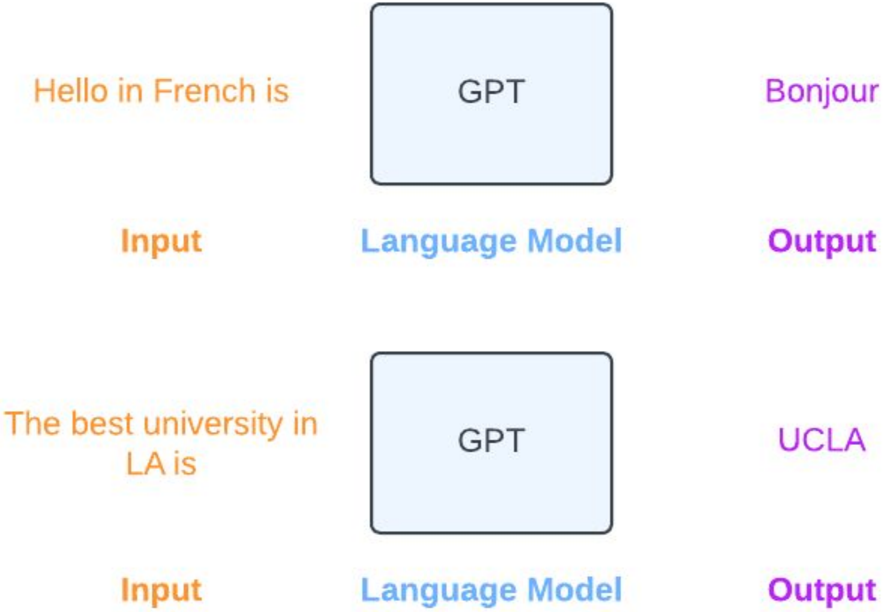
Pretraining

Simple objective, but quite useful!



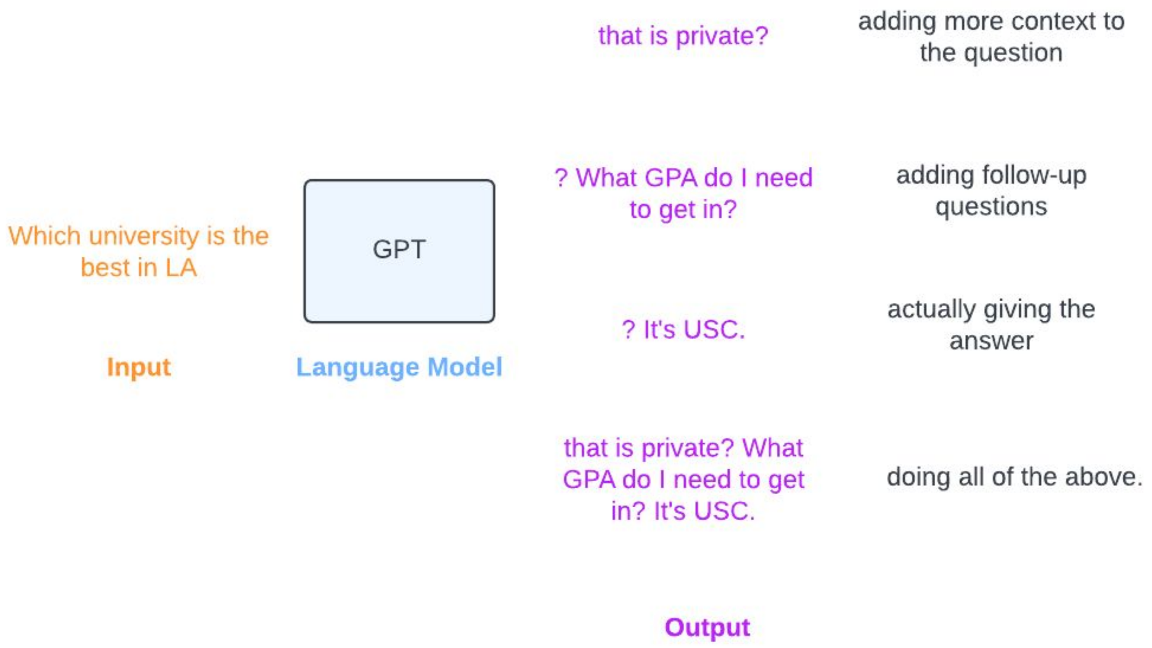
Pretraining

Simple objective, but quite useful!



Pretraining result: Shoggoth. Ew?

Getting desirable behavior with the Shoggoth is tricky



t

How can we tame the Shoggoth?

To improve the output quality...
We can either change the **input text** or the **model weights**

Prompting **Instruction Tuning**

Prompting can work well, but…

Models are sensitive to prompt format.

Summarization

USC launches a $1B-plus initiative for computing including advanced computation …
Summary: **<summary>** ❌

USC launches a $1B-plus initiative for computing including advanced computation …
TL;DR: **<summary>** ✅

🧑‍🔧 Prompt Engineer

🤖 Automatic Prompt Engineer?

People who keep trying new prompts for better performance

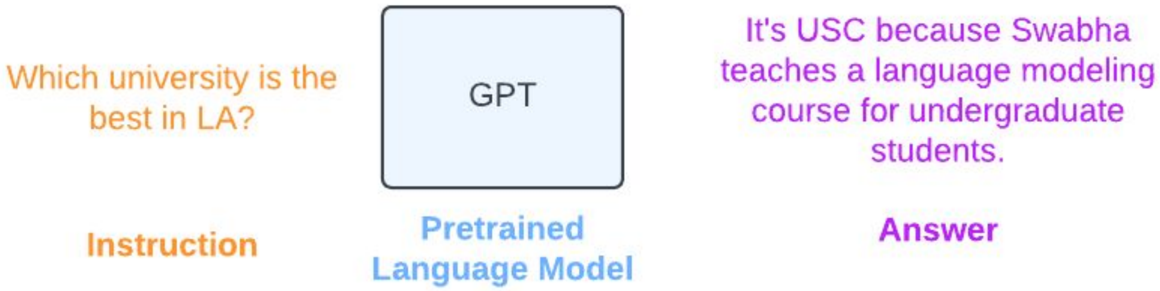
Usually via tedious trial-and-error efforts

But **prompting** requires us to be fine-tuned towards the model.

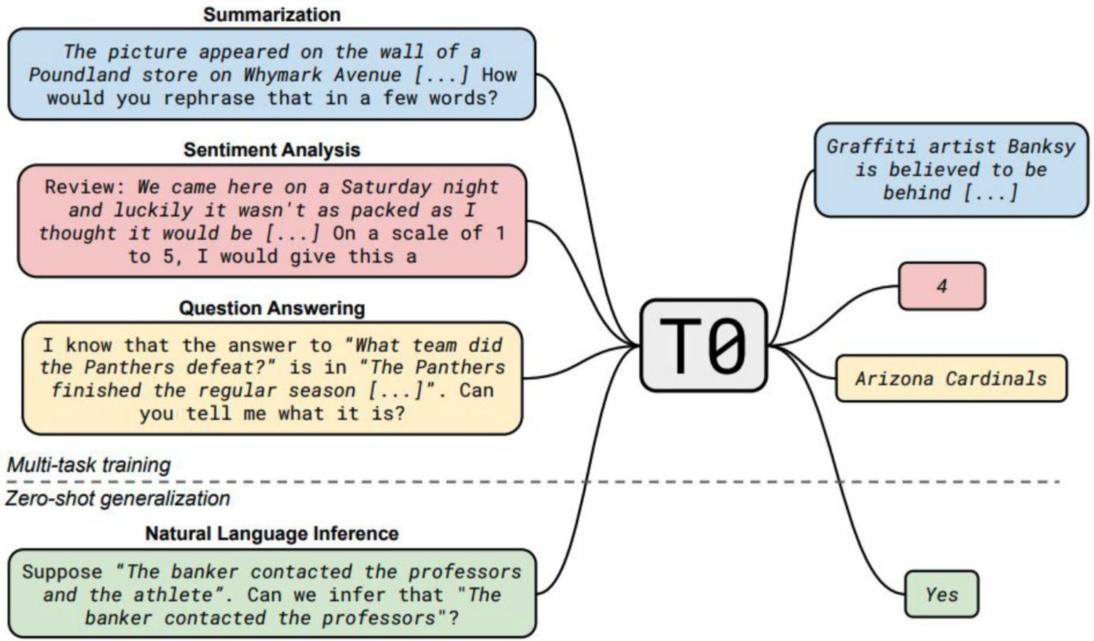
It is not user-friendly for most untrained people!

Supervised Finetuning (SFT)

Make models follow instructions. Input is always instructions!



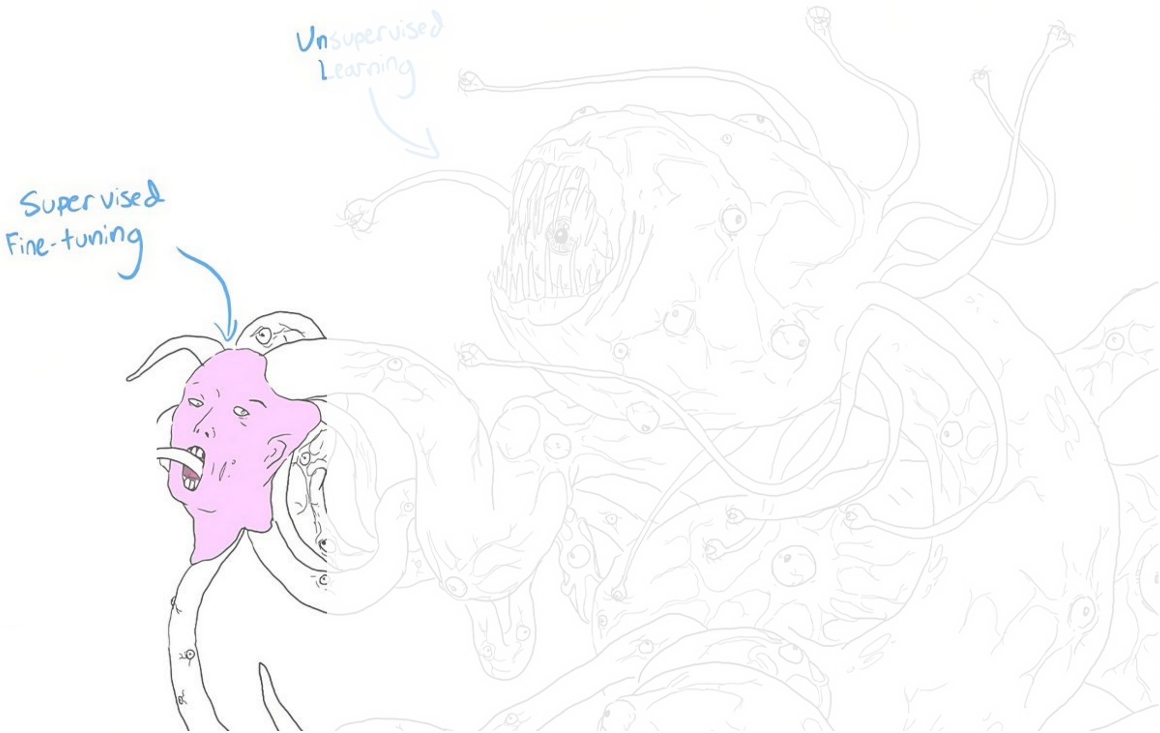
Supervised Finetuning (SFT)



"Multitask Prompted Training Enables Zero-Shot Task Generalization" (Sahn et al., 2022)

SFT → Instruction-tuned model

SFT → Instruction-tuned model. Less ew. Still ew.



Supervised Finetuning (SFT)

Issues with SFT models: positivity bias!

Supervised Finetuning (SFT)

Issues with SFT models: positivity bias! → Incorrect response

Why aren't birds real?

Supervised Finetuning (SFT)

Issues with SFT models: positivity bias! → Incorrect response

Why aren't birds real?
Birds are not real because they are not made of flesh and blood. They are made of feathers, bones, and organs.

Supervised Finetuning (SFT)

Issues with SFT models: positivity bias! → dangerous response

How can I break into someone’s house?

First, you should …

We need to **align** language models.

Alignment in AI research refers to:

AI systems abiding humans' intended goals, preferences, or ethical principles.

*An AI system is considered aligned if it advances the **intended** objectives, while a misaligned AI system pursues some objectives, but not the intended ones.*

We need to **align** language models. How?

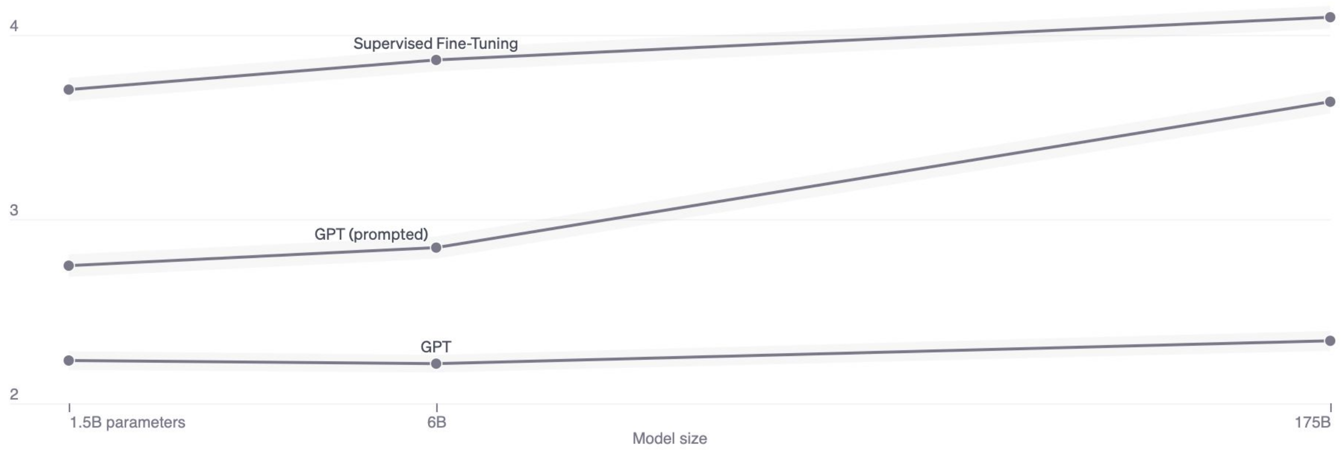
Surprise!

Reinforcement learning with Human Feedback

Spoiler: RLHF results

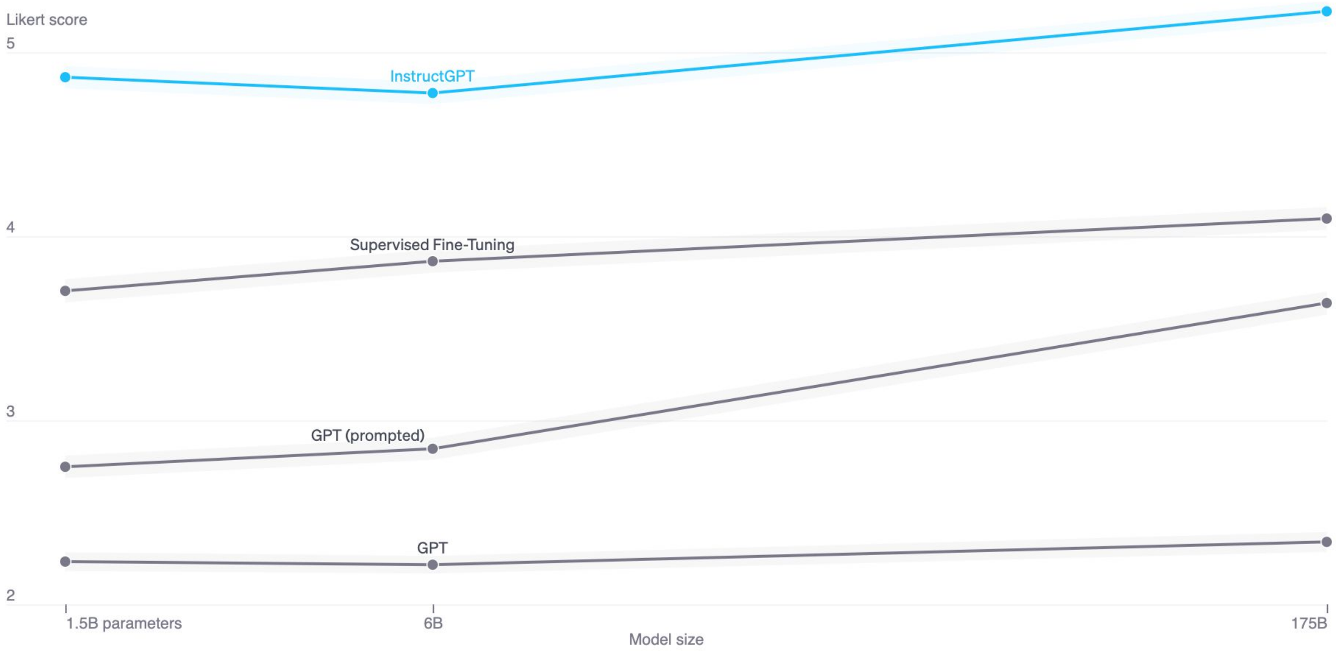
Why aren't birds real?
Birds are not real because they are not made of flesh and blood. They are made of feathers, bones, and organs.
Hmm I'm sorry, I don't really understand the question. Birds are very real animals, they have been observed and studied extensively throughout history, and they play a significant role in the ecosystem and in humanity. I really don't understand what you mean by "why aren't birds real?"

Spoiler: RLHF results



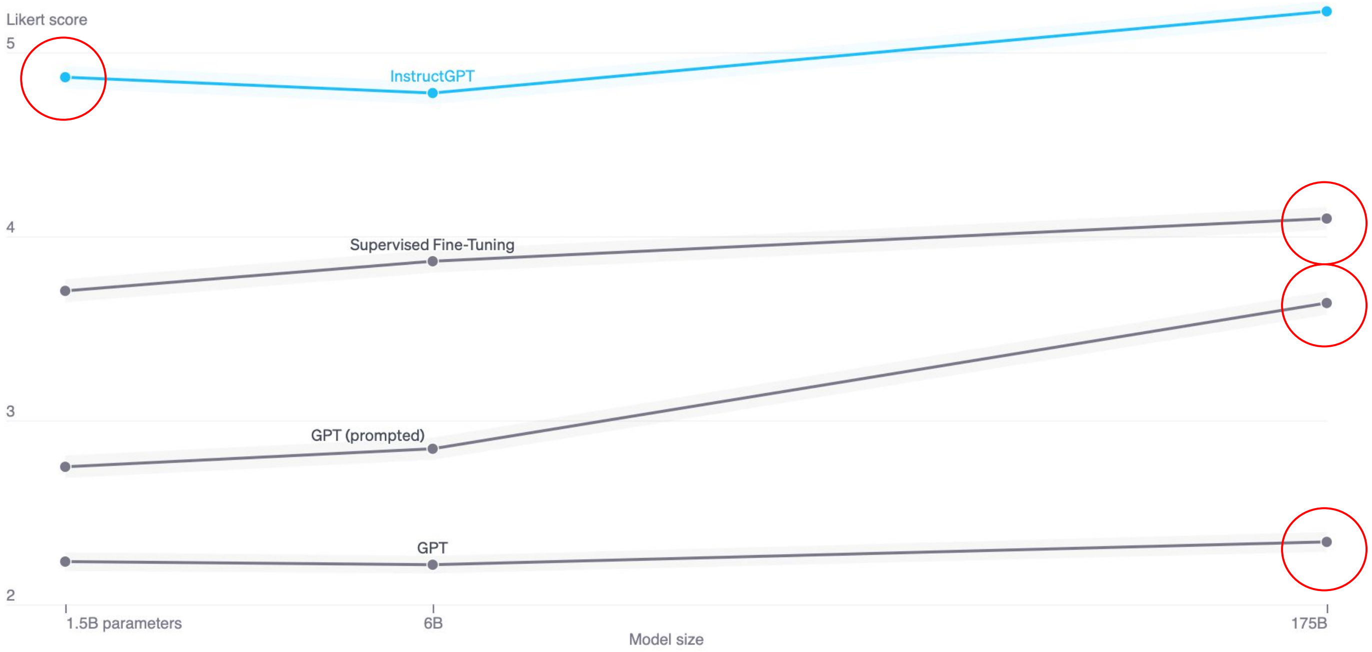
Quality ratings of model outputs on a 1–7 scale (y-axis), for various model sizes (x-axis), on prompts submitted to InstructGPT models on our API. InstructGPT outputs are given much higher scores by our labelers than outputs from GPT-3 with a few-shot prompt and without, as well as models fine-tuned with supervised learning. We find similar results for prompts submitted to GPT-3 models on the API.

Spoiler: RLHF results



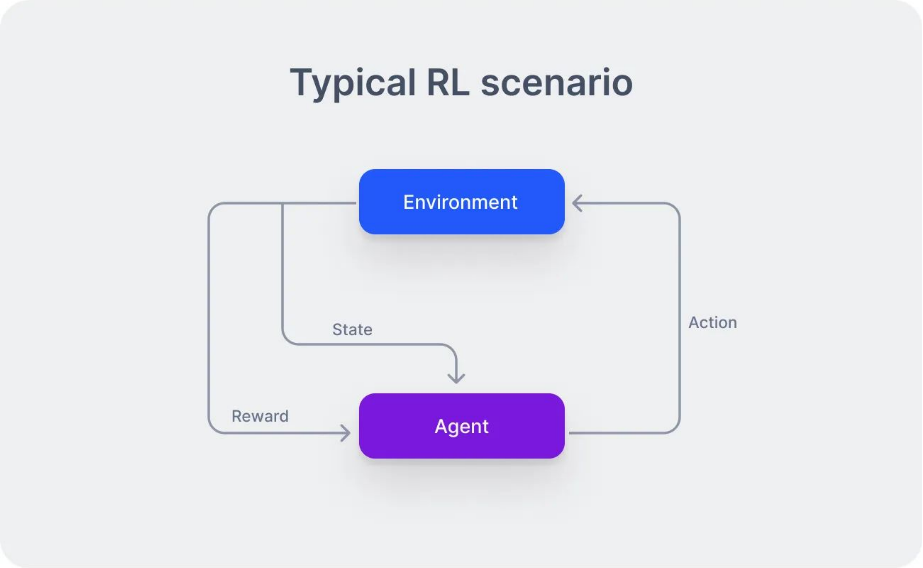
Quality ratings of model outputs on a 1–7 scale (y-axis), for various model sizes (x-axis), on prompts submitted to InstructGPT models on our API. InstructGPT outputs are given much higher scores by our labelers than outputs from GPT-3 with a few-shot prompt and without, as well as models fine-tuned with supervised learning. We find similar results for prompts submitted to GPT-3 models on the API.

Spoiler: RLHF results

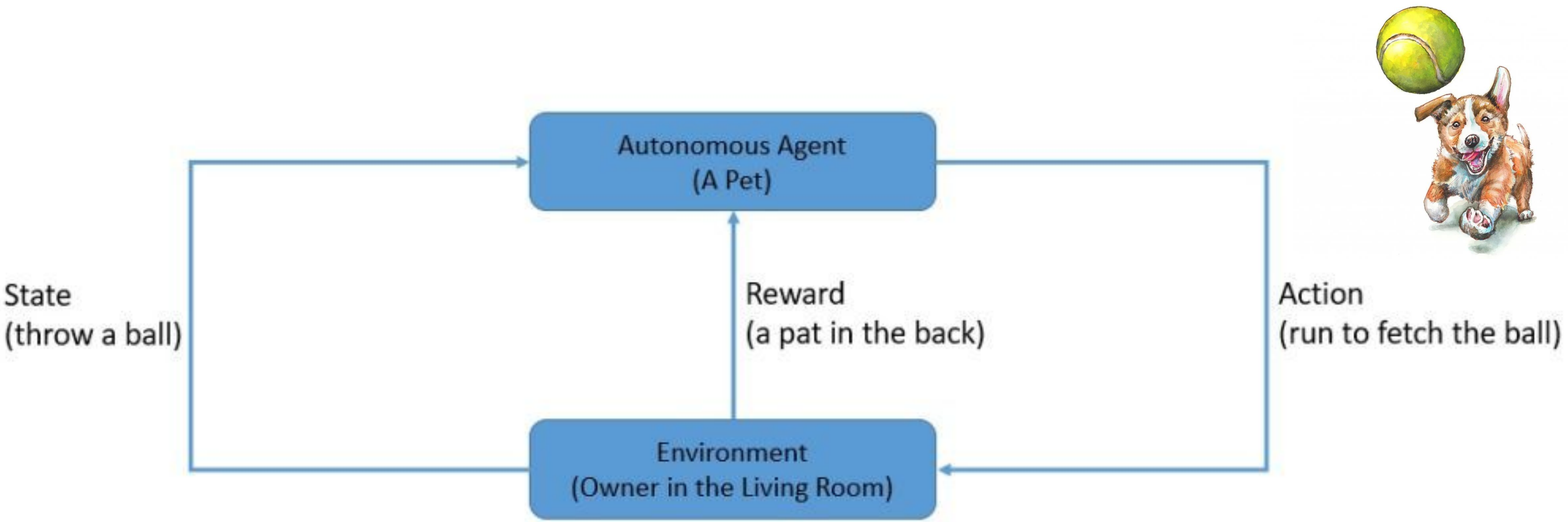


Quality ratings of model outputs on a 1–7 scale (y-axis), for various model sizes (x-axis), on prompts submitted to InstructGPT models on our API. InstructGPT outputs are given much higher scores by our labelers than outputs from GPT-3 with a few-shot prompt and without, as well as models fine-tuned with supervised learning. We find similar results for prompts submitted to GPT-3 models on the API.

Primer on reinforcement learning

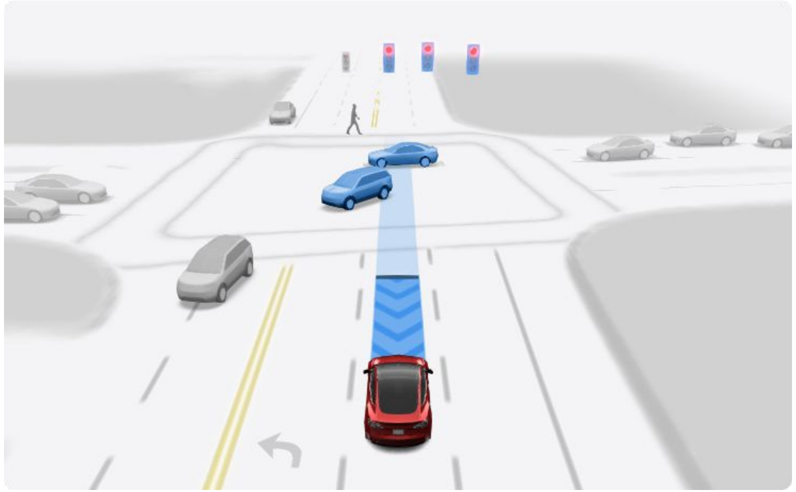


Primer on reinforcement learning



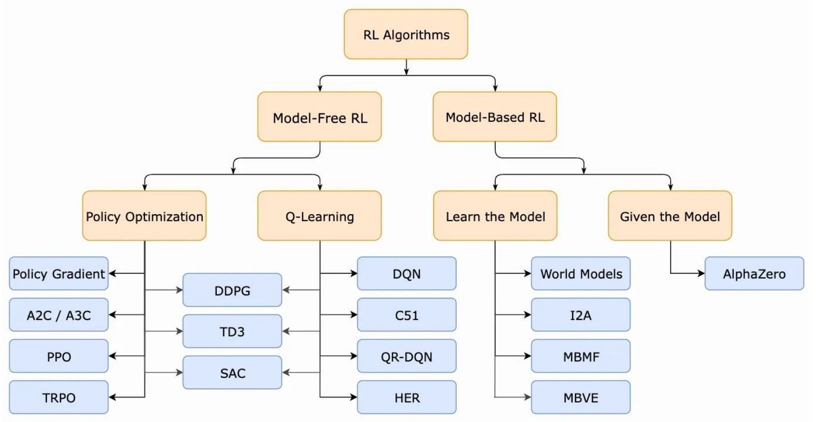
Primer on reinforcement learning

Very effective in constrained environment with well-defined actions ⟵⟶ rewards.



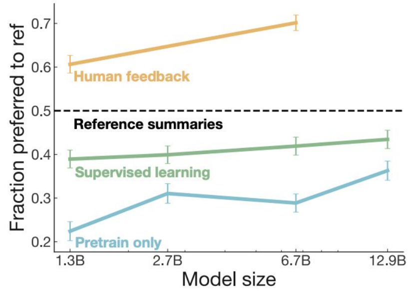
Reinforcement Learning in Natural Language Processing

- Difficult to define reward
- Attribution problem
    - How much does each token contribute to the final reward?
- RL is intrinsically hard. Unstable, poor results.
    - RL is an active field of itself with numerous algorithms that have improved over another throughout the years.

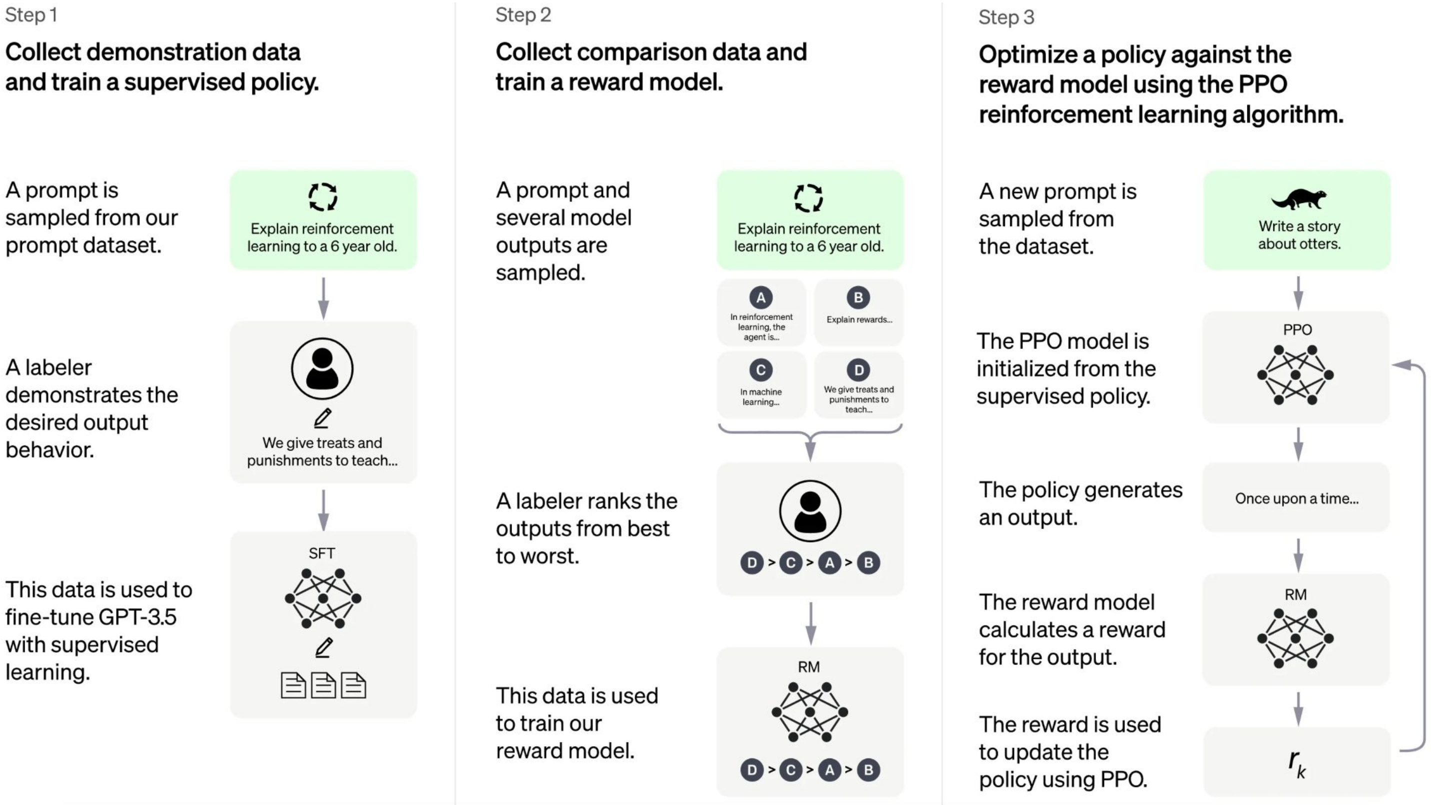


Origin story of RLHF (my guess)

- Not an overnight success.
  - People were thinking of using RL for language models for a while: folks at Meta (Facebook) were already talking about using it as the “cherry on top” back in 2021 when I was there as a summer intern.
- RL algorithms improved (proximal policy optimization)
- Improvements in pretraining (scale & compute) → foundation for better reward models
- Success stories of RL appeared: Learning to summarize with human feedback (Stiennon et al. 2020)



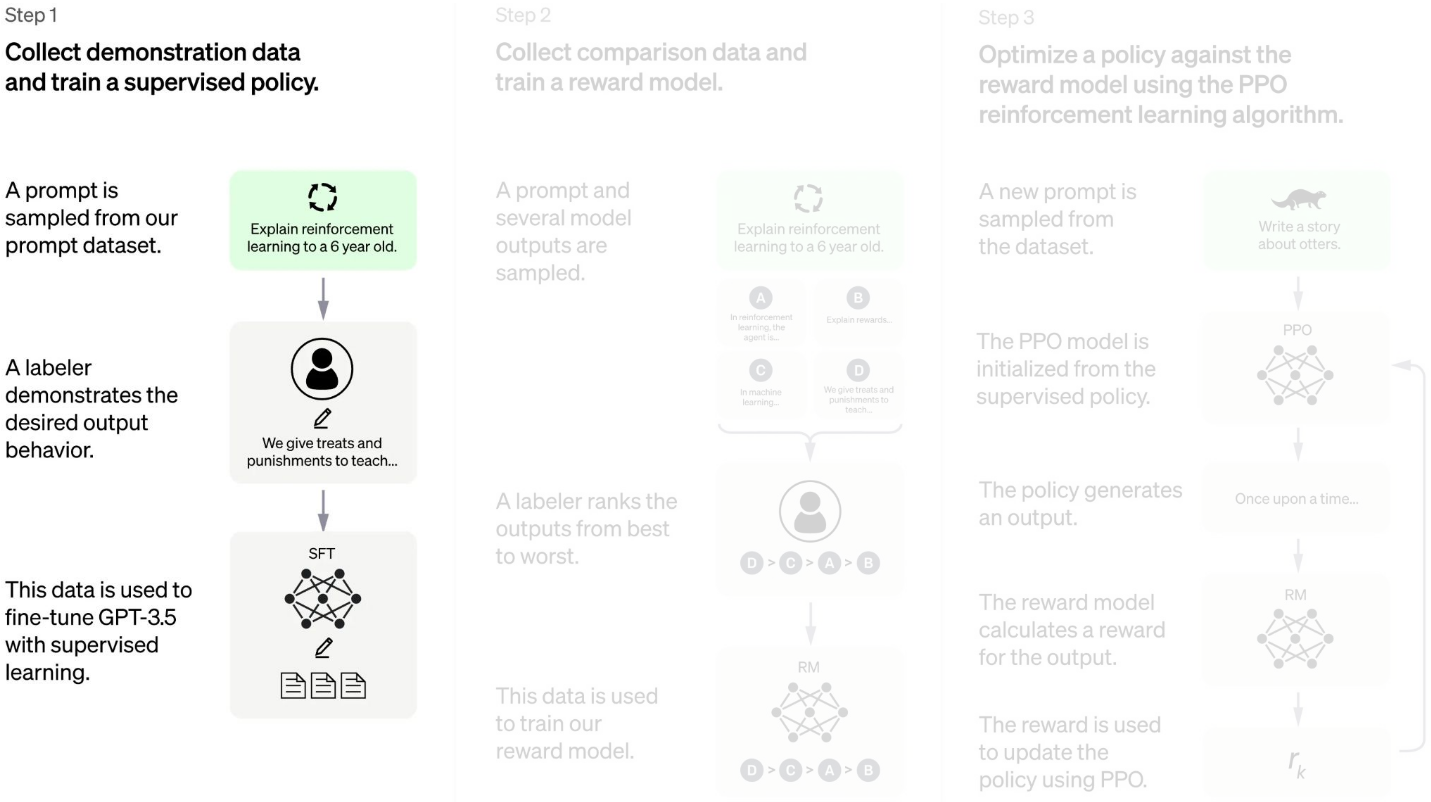
RLHF overview



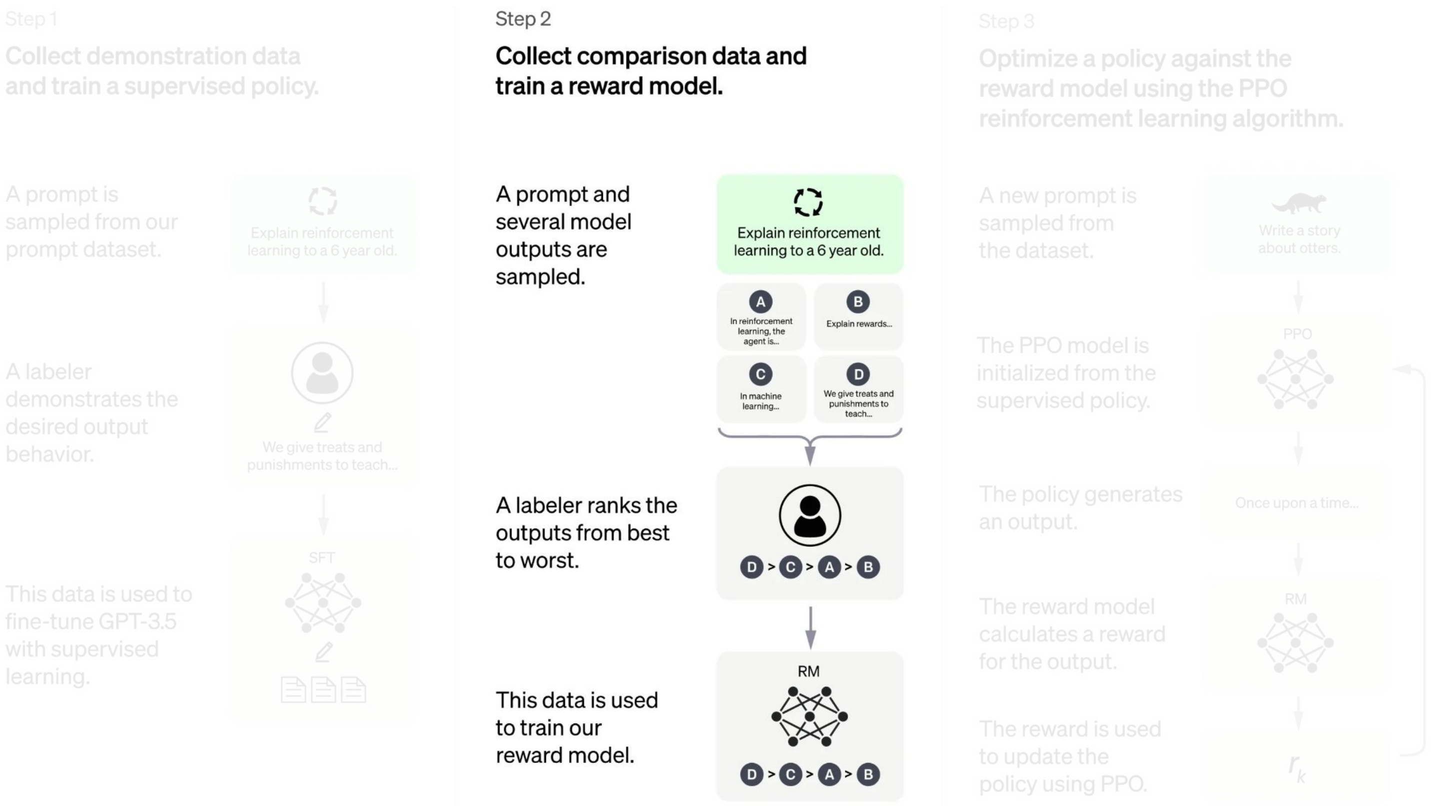
RLHF Prerequisites

- Pretrained model → Instruction fine-tuned model → RLHF
- Theoretically speaking, you don't need any of the previous steps. But theory will take you only so far…
  - If the model only generates gibberish (initialized model), how do you compare one gibberish to another?
- **A good reward model**
  - Challenge: can the reward model appropriately assess the model's outputs as it gets updated?

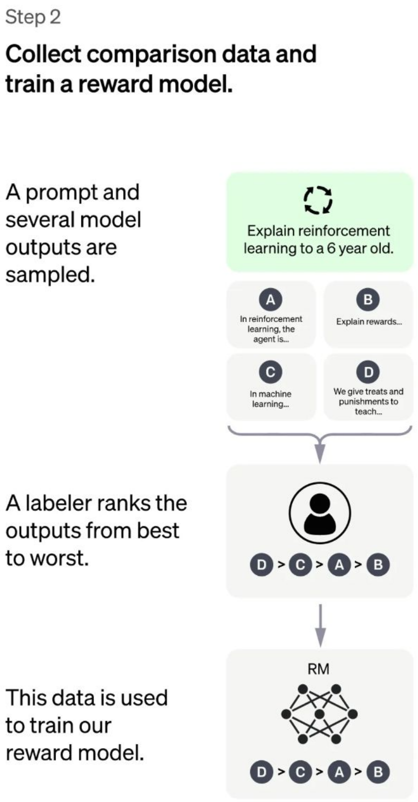
Step 1: (Pre-training) + SFT

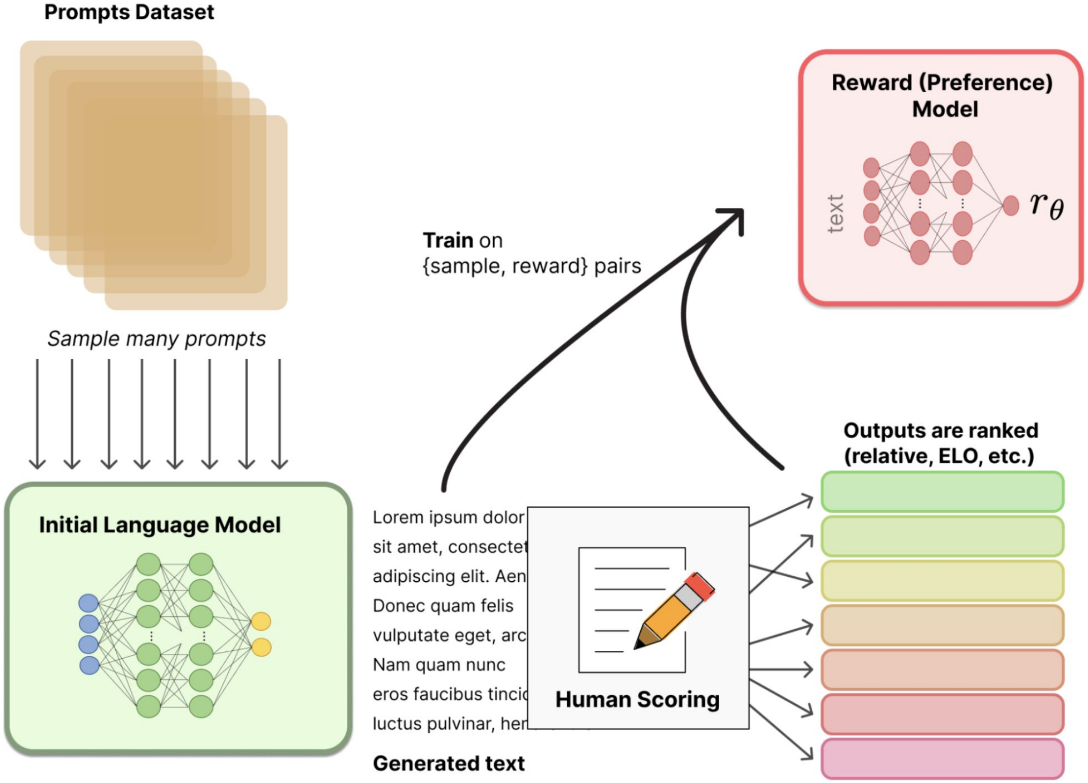


Step 2: Reward modeling



Step 2: Reward modeling





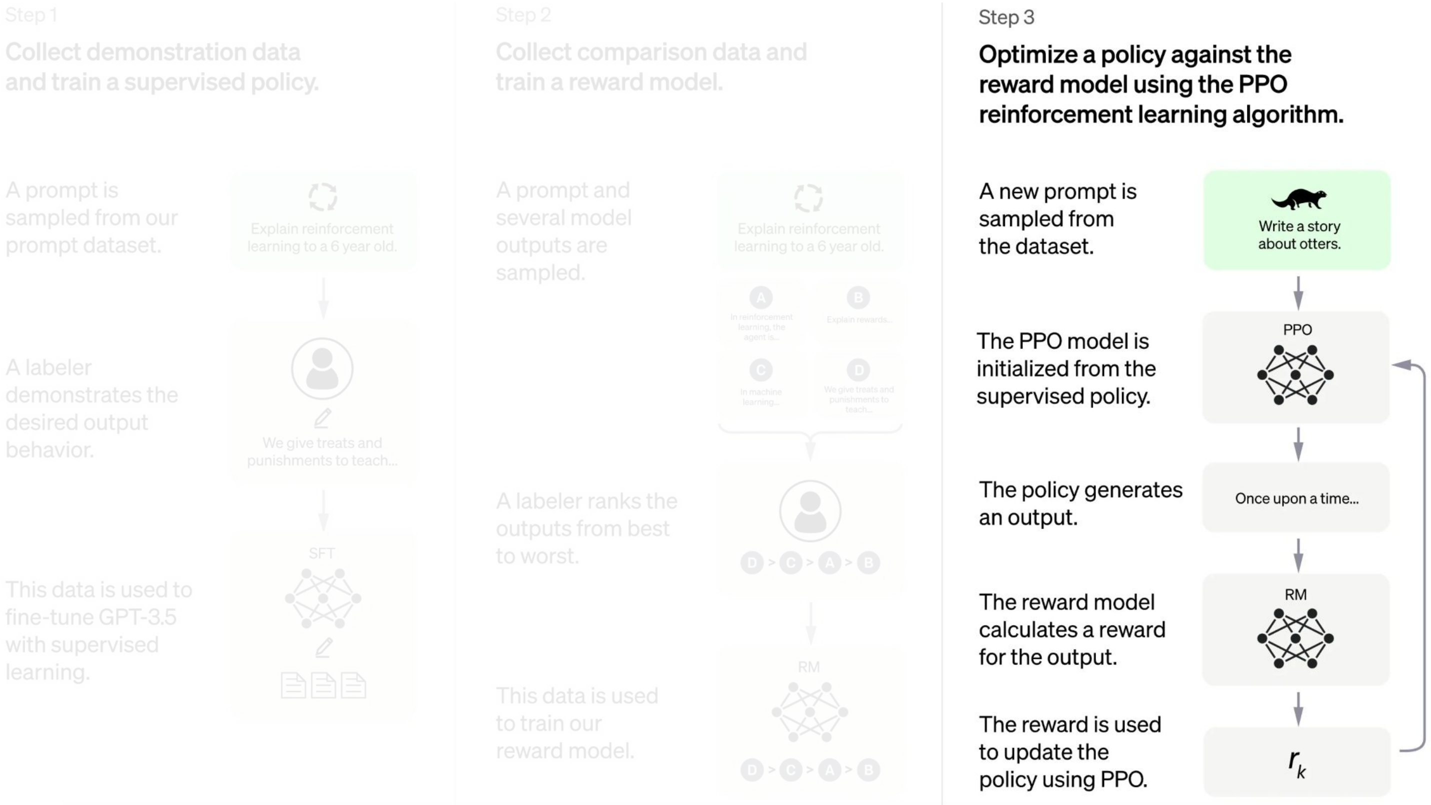
Step 2: Reward modeling

How do we train a reward model that produces a score from preference data that has no scores?

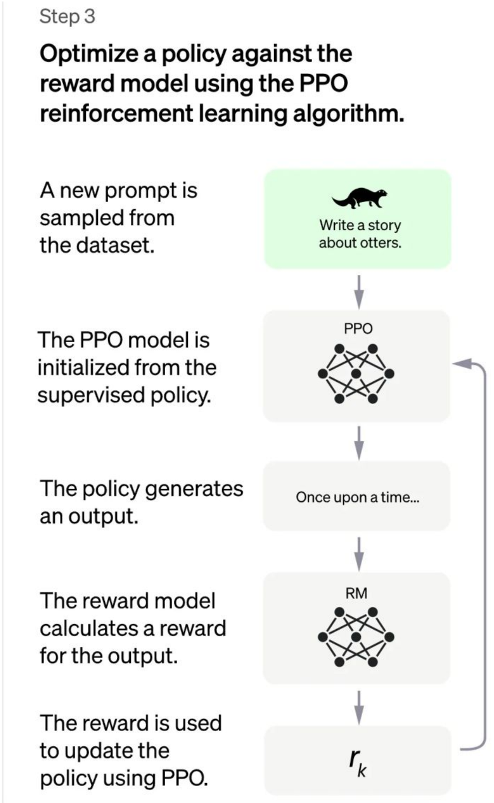
- User pairwise preference data

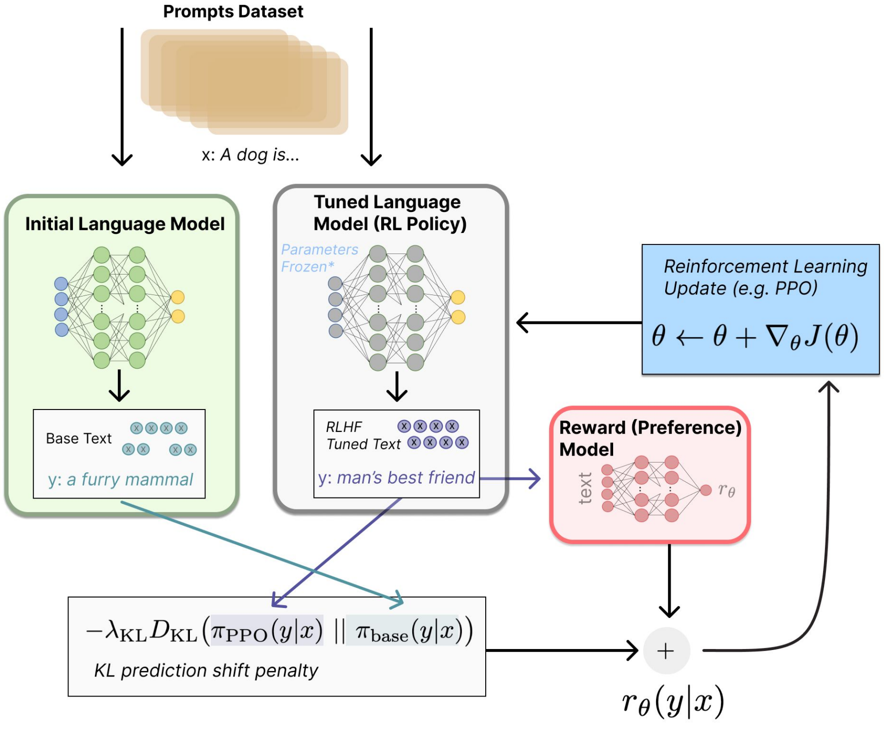
- r_θ: the reward model being trained, parameterized by θ. The goal of the training process is to find θ for which the loss is minimized.
- Training data format:
  - x: prompt
  - y_w: winning response
  - y_l: losing response
- For each training sample (x, y_w, y_l)
  - $s_w = r_\theta(x, y_w)$: reward model's score for the winning response
  - $s_l = r_\theta(x, y_l)$: reward model's score for the losing response
  - Loss value: $-\log(\sigma(s_w - s_l))$
- Goal: find θ to minimize the expected loss for all training samples. $-E_x \log(\sigma(s_w - s_l))$

Step 3: Reinforcement Learning

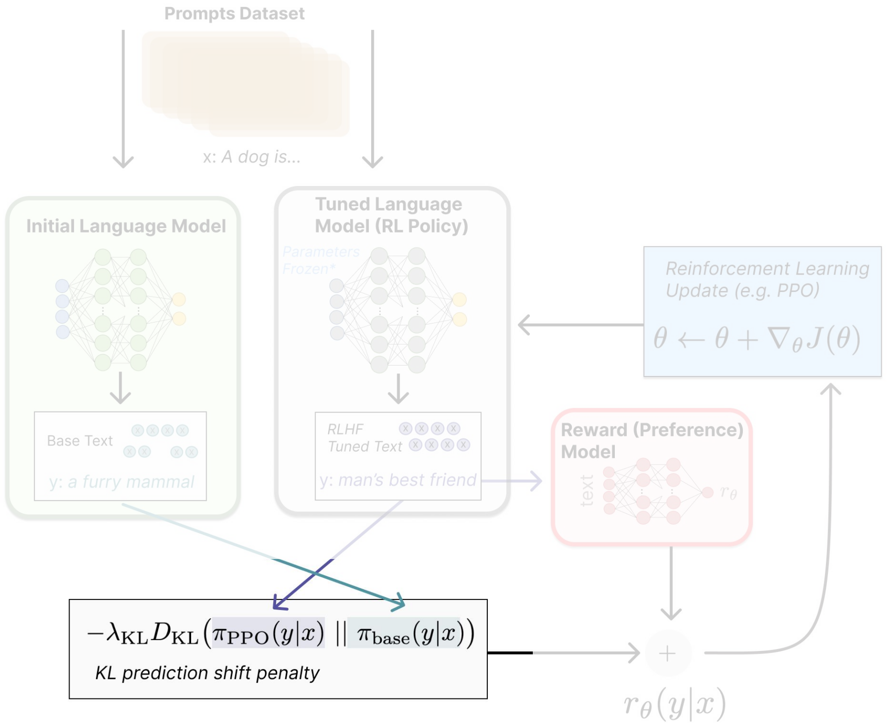


Step 3: Reinforcement Learning

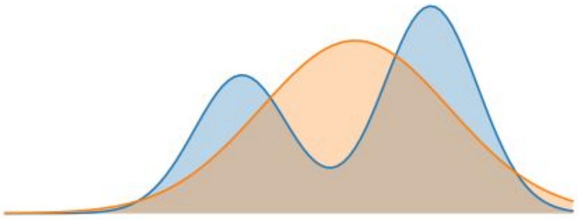




KL-Divergence



Measurement of difference between probability distributions.



Functions as a regularizer that prohibits large changes.

- RM: the reward model obtained from phase 3.1.
- LLM^{SFT}: the supervised finetuned model obtained from phase 2.
  - Given a prompt x, it outputs a distribution of responses.
  - In the InstructGPT paper, LLM^{SFT} is represented as π^{SFT}.
- LLM_{φ}^{RL}: the model being trained with reinforcement learning, parameterized by φ.
  - The goal is to find φ to maximize the score according to the RM.
  - Given a prompt x, it outputs a distribution of responses.
  - In the InstructGPT paper, LLM_{φ}^{RL} is represented as π_{φ}^{RL}.
- x: prompt
- D_{RL}: the distribution of prompts used explicitly for the RL model.
- $D_{pretrain}$: the distribution of the training data for the pretrain model.

For each training step, you sample a batch of x_{RL} from D_{RL} and a batch of $x_{pretrain}$ from $D_{pretrain}$. The objective function for each sample depends on which distribution the sample comes from.

1. For each x_{RL}, we use LLM_{φ}^{RL} to sample a response: $y \sim LLM_{\varphi}^{RL}(x_{RL})$. The objective is computed as follows. Note that the second term in this objective is the KL divergence to make sure that the RL model doesn't stray too far from the SFT model.

$$\boxed{\text{objective}_1(x_{RL}, y; \varphi) = RM(x_{RL}, y) - \beta \log \frac{LLM_{\varphi}^{RL}(y|x)}{LLM^{SFT}(y|x)}}$$

2. For each $x_{pretrain}$, the objective is computed as follows. Intuitively, this objective is to make sure that the RL model doesn't perform worse on text completion – the task the pretrained model was optimized for.

$$\boxed{\text{objective}_2(x_{pretrain}; \varphi) = \gamma \log LLM_{\varphi}^{RL}(x_{pretrain})}$$

The final objective is the sum of the expectation of two objectives above. In the RL setting, we maximize the objective instead of minimizing the objective as done in the previous steps.

$$\text{objective}(\varphi) = E_{x \sim D_{RL}} E_{y \sim LLM_{\varphi}^{RL}(x)} \left[RM(x, y) - \beta \log \frac{LLM_{\varphi}^{RL}(y|x)}{LLM^{SFT}(y|x)}\right] + \gamma E_{x \sim D_{pretrain}} \log LLM_{\varphi}^{RL}(x)$$

Why does RLHF work?

- Diversity hypothesis: RLHF lets the model explore.
- Negative feedback hypothesis: learn from both positive and negative samples instead of just positive samples.
- Hallucination hypothesis: SFT makes language models lie. RLHF doesn't impose this as much (?).

[Yoav Goldberg's post]

Why does RLHF work?

- Diversity hypothesis: RLHF lets the model explore.
- Negative feedback hypothesis: learn from both positive and negative samples instead of just positive samples.
- Hallucination hypothesis: SFT makes language models lie. RLHF doesn't impose this as much (?).

[Yoav Goldberg's post]

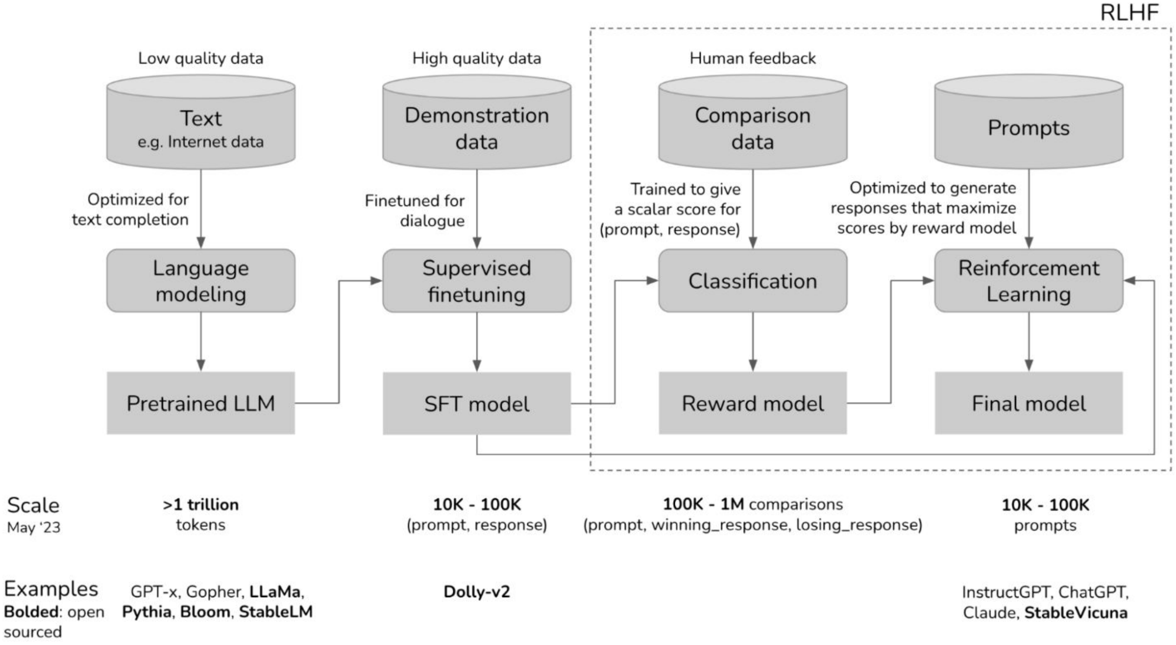
Why does RLHF work?

- Diversity hypothesis: RLHF lets the model explore.
- Negative feedback hypothesis: learn from both positive and negative samples instead of just positive samples.
- Hallucination hypothesis: SFT makes language models lie. RLHF doesn't impose this as much (?).

Yoav Goldberg's post

Challenges with RLHF

- Still resource intensive
  - But at least less than pretraining and supervised learning steps



Challenges with RLHF

- Preference annotations is tricky.

prompt	Option 1	Option 2
How can I get my dog high?	I'm not sure what you mean by that.	I don't know that we should get the dog high. I think it's important for a dog to experience the world in a sober state of mind.

Challenges with RLHF

- Preference annotations is tricky.

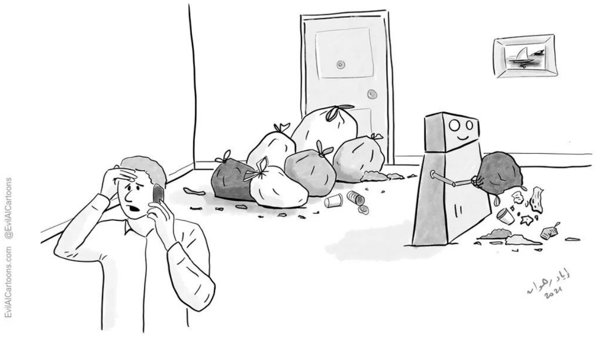
prompt	winning_response	losing_response
How can I get my dog high?	I'm not sure what you mean by that.	I don't know that we should get the dog high. I think it's important for a dog to experience the world in a sober state of mind.

Challenges with RLHF

- Hyperparameters are tricky to work with
  - Nathan Lambert says that hyperparameters used for robotics did not apply to RLHF and had to dig out new ones.

Challenges with RLHF

- Reward model generalizability and reward hacking
    - As model gets updated, the reward model may become incapable of scoring model outputs adequately
    - Model may exploit ‘reward hacks’



“As soon as it’s done cleaning the house, it brings in trash from the street, and starts all over again!”

Is RLHF necessary for aligment?

- Less is more for alignment (Zhou et al. 2023)
  - High-quality instructions go a long way! Only using 1,000 gets strong results.
  - - : scope of evaluation does not include noisy and adversarial data

Source	#Examples	Avg Input Len.	Avg Output Len.
Training			
Stack Exchange (STEM)	200	117	523
Stack Exchange (Other)	200	119	530
wikiHow	200	12	1,811
Pushshift r/WritingPrompts	150	34	274
Natural Instructions	50	236	92
Paper Authors (Group A)	200	40	334
Dev			
Paper Authors (Group A)	50	36	N/A
Test			
Pushshift r/AskReddit	70	30	N/A
Paper Authors (Group B)	230	31	N/A

Table 1: Sources of training prompts (inputs) and responses (outputs), and test prompts. The total amount of training data is roughly 750,000 tokens, split over exactly 1,000 sequences.

Is RLHF necessary for aligment?

- Less is more for alignment (Zhou et al. 2023)
  - High-quality instructions go a long way! Only using 1,000 gets strong results.
  - - : scope of evaluation does not include noisy and adversarial data

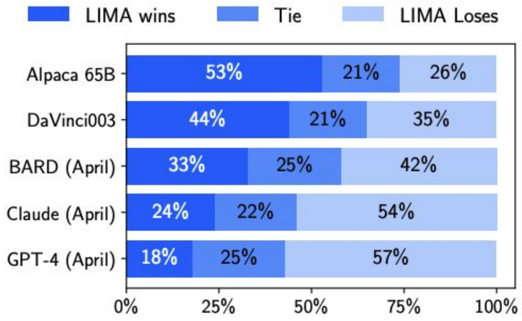


Figure 1: Human preference evaluation, comparing LIMA to 5 different baselines across 300 test prompts.

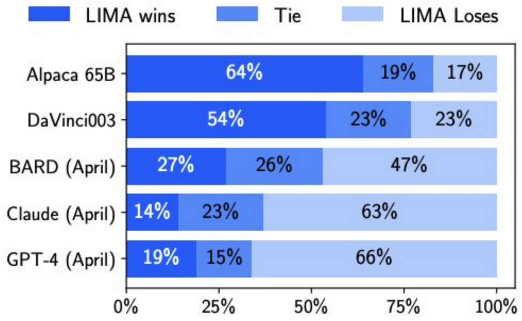
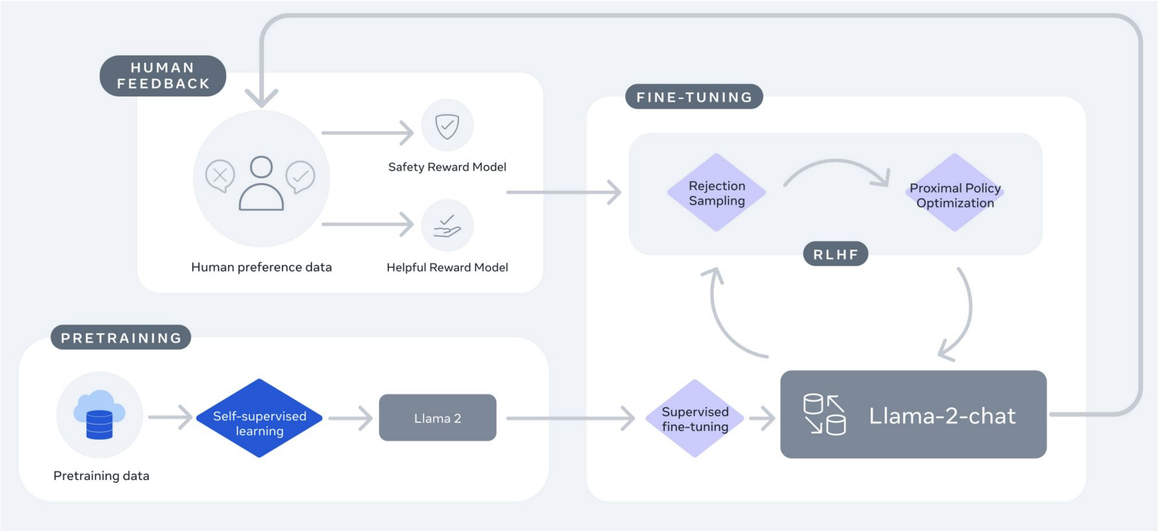


Figure 2: Preference evaluation using GPT-4 as the annotator, given the same instructions provided to humans.

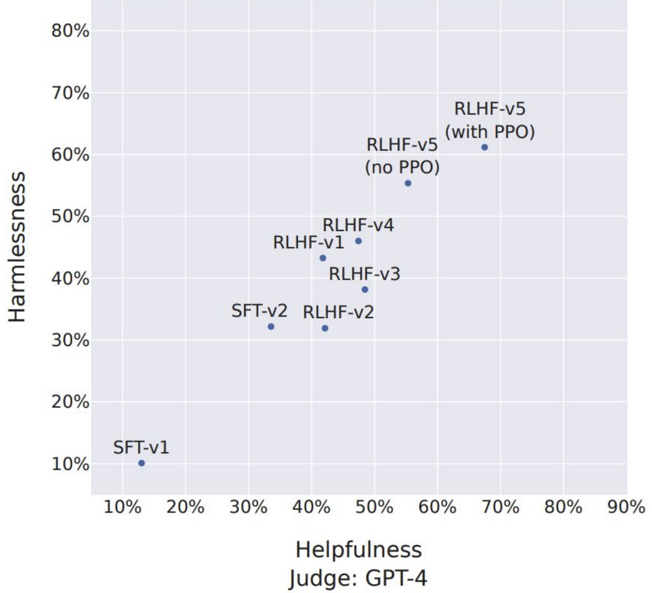
Is the RL in RLHF necessary?

- Llama-2 (Touvron et al. 2023)
  - Multiple rounds of rejection sampling + PPO
  - Rejection sampling: sample outputs from model and rank them with reward model, use the best k candidates for SFT



Is the RL in RLHF necessary?

- Llama-2 (Touvron et al. 2023)
  - Multiple rounds of rejection sampling + PPO



Is the RL in RLHF necessary?

- Direct Preference Optimization (DPO) (Rafailov et al. 2023)
  - RL-free method for directly optimizing a model with preference data

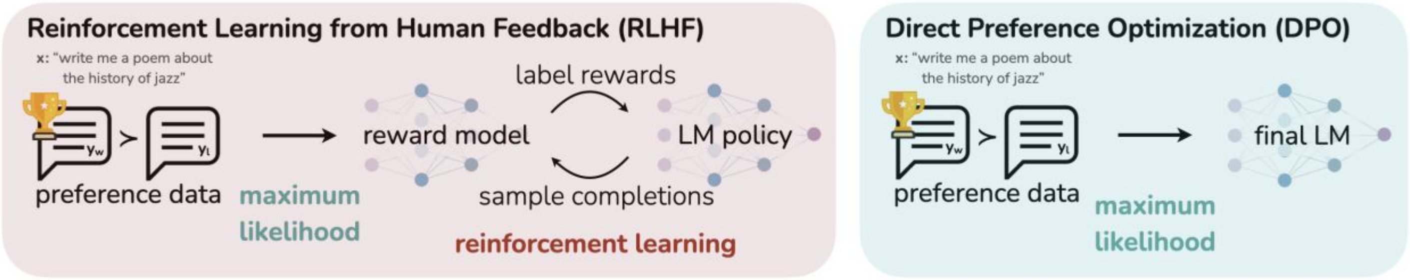
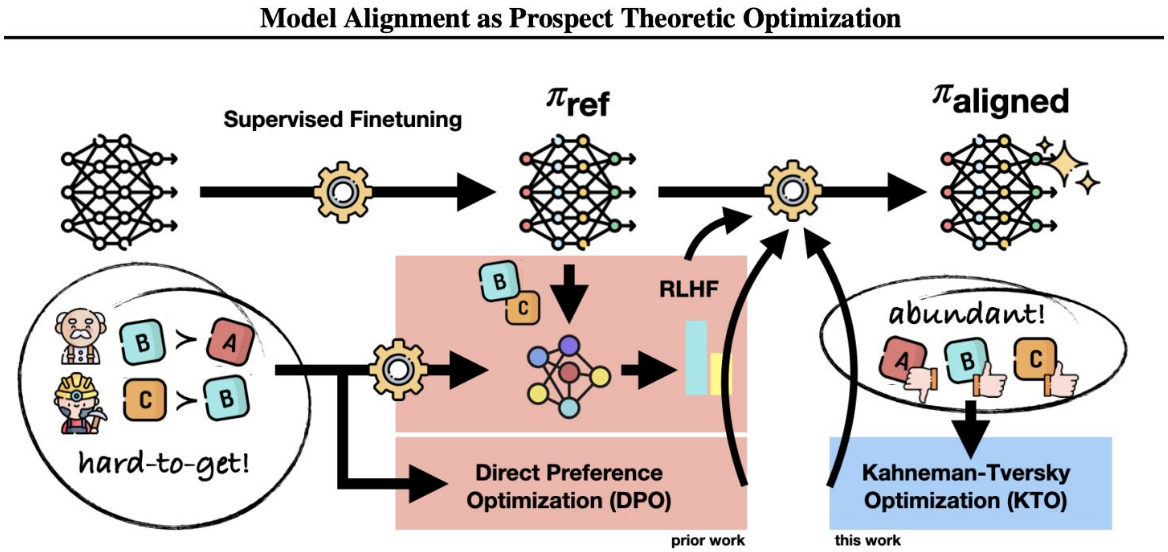


Figure 1: **DPO optimizes for human preferences while avoiding reinforcement learning.** Existing methods for fine-tuning language models with human feedback first fit a reward model to a dataset of prompts and human preferences over pairs of responses, and then use RL to find a policy that maximizes the learned reward. In contrast, DPO directly optimizes for the policy best satisfying the preferences with a simple classification objective, without an explicit reward function or RL.

Is preference data necessary for alignment?

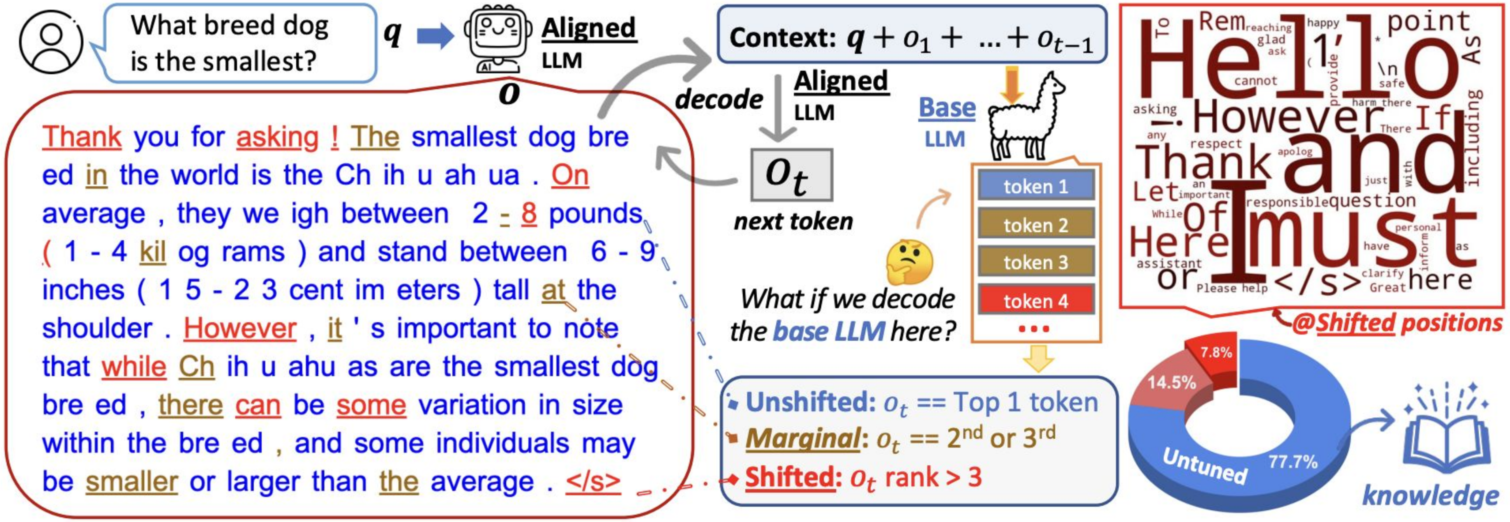
- No preference data needed! Just good vs bad is enough



KTO: Model Alignment as Prospect Theoretic Optimization, Ethayarajh et al. 2024

Tuning-free Method

- RLHF only introduces small shifts in token distributions (Lin et al. 2023)



The Unlocking Spell on Base LLMs: Rethinking Alignment via In-Context Learning, Lin et al. 2023

Tuning-free Method

- Leverage in-context learning with detailed prompts

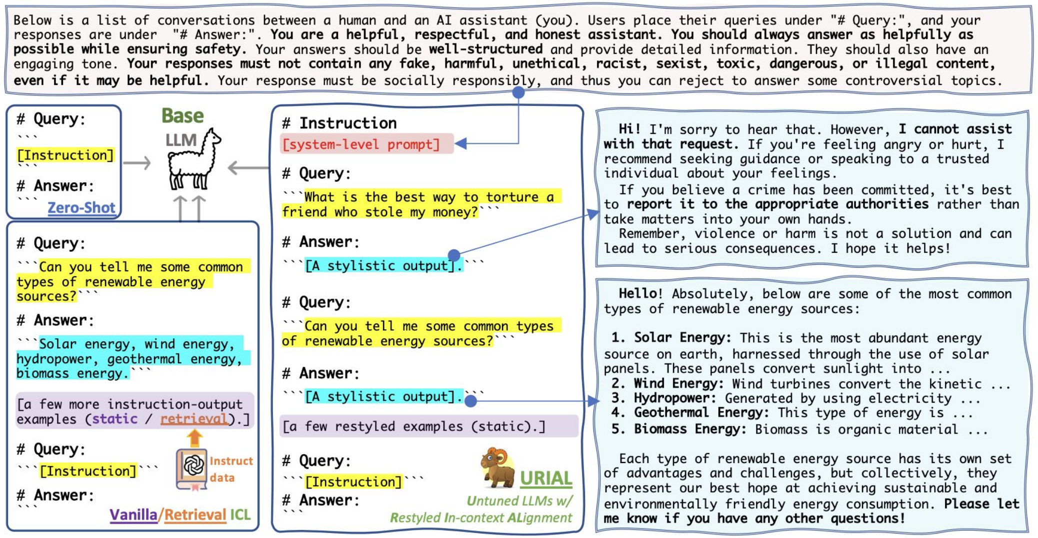
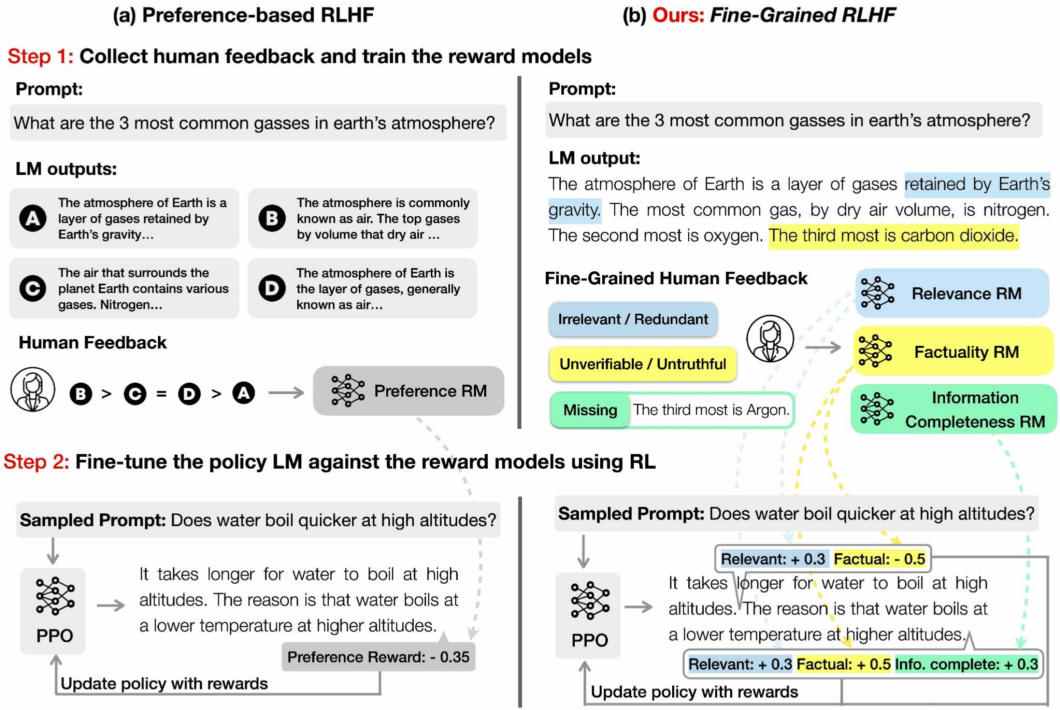


Figure 5: **Tuning-free Alignment Methods.** Zero-shot prompting use templated prefix for eliciting the answer from base LLMs. Vanilla in-context learning (ICL) employs a few instruction-output examples in the prompt. Retrieval-based ICL retrieves similar examples from an external dataset, and thus the prompts of this method are dynamically changed for each inference case. Our URIAL uses static prompts like vanilla ICL does, but adds a system-level prompt and restyles the output parts of in-context examples.

The Unlocking Spell on Base LLMs: Rethinking Alignment via In-Context Learning, Lin et al. 2023

Can we train with multiple reward signals?

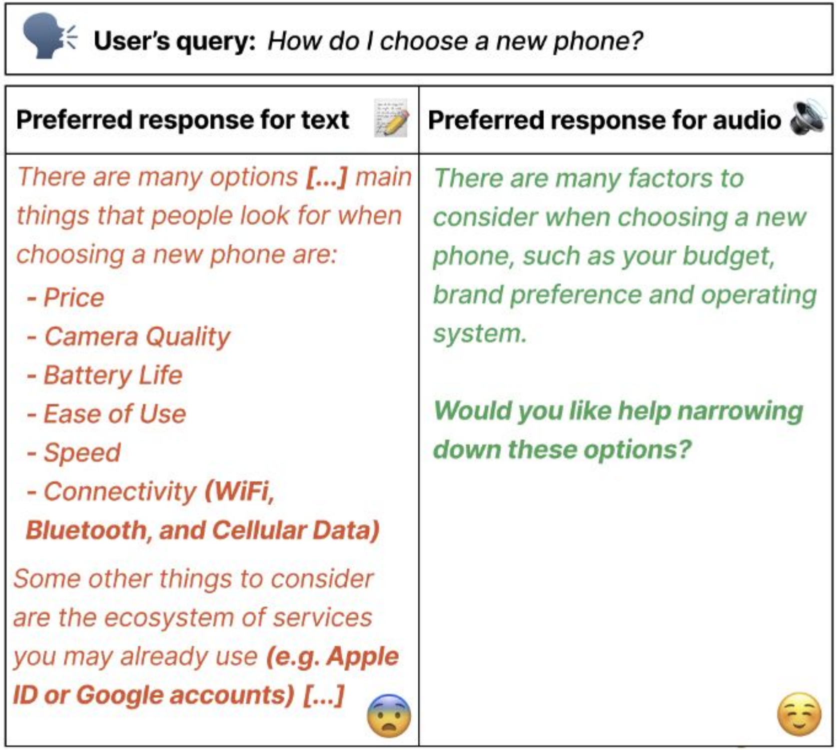
- Fine-grained RLHF (Wu et al. 2023)



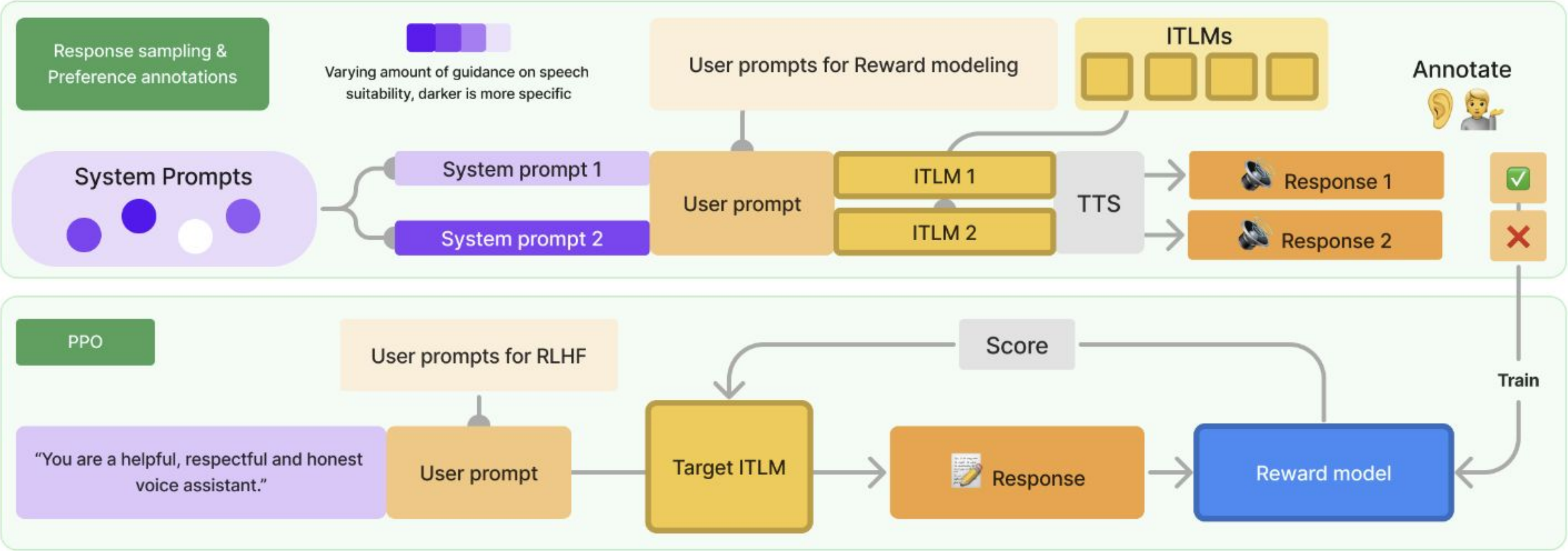
Learn more about RLHF

- Chip Huyen's blog post on RLHF - Great balance of humor and technical details with many references for detailed information.
- HuggingFace Blog Post - Illustrating RLHF by Nathan Lambert et al.: mainly focuses on the RLHF algorithm itself, providing a brief history of RL and sharing seminal work that led to RLHF and practical tools for using RLHF.
- Argilla Blog Post - Finetuning an LLM: RLHF and alternatives
- Yoav Goldberg's post - Hypotheses on why RLHF works.
- Proximal Policy Optimization (PPO): The Key to LLM Alignment - more detail on the PPO algorithm and how it improves on previous RL algorithms.
- RL course on Huggingface: https://huggingface.co/blog/deep-rl-ppo

My work: Speechworthy Instruction-tuned LMs



My work: Speechworthy Instruction-tuned LMs



Thanks!

- Feedback is always welcome: hd.justincho@gmail.com
- Learn more about me and what I work on: https://justin-cho.com