

Learning from Naturally Occurring Human Feedback

Ge Gao

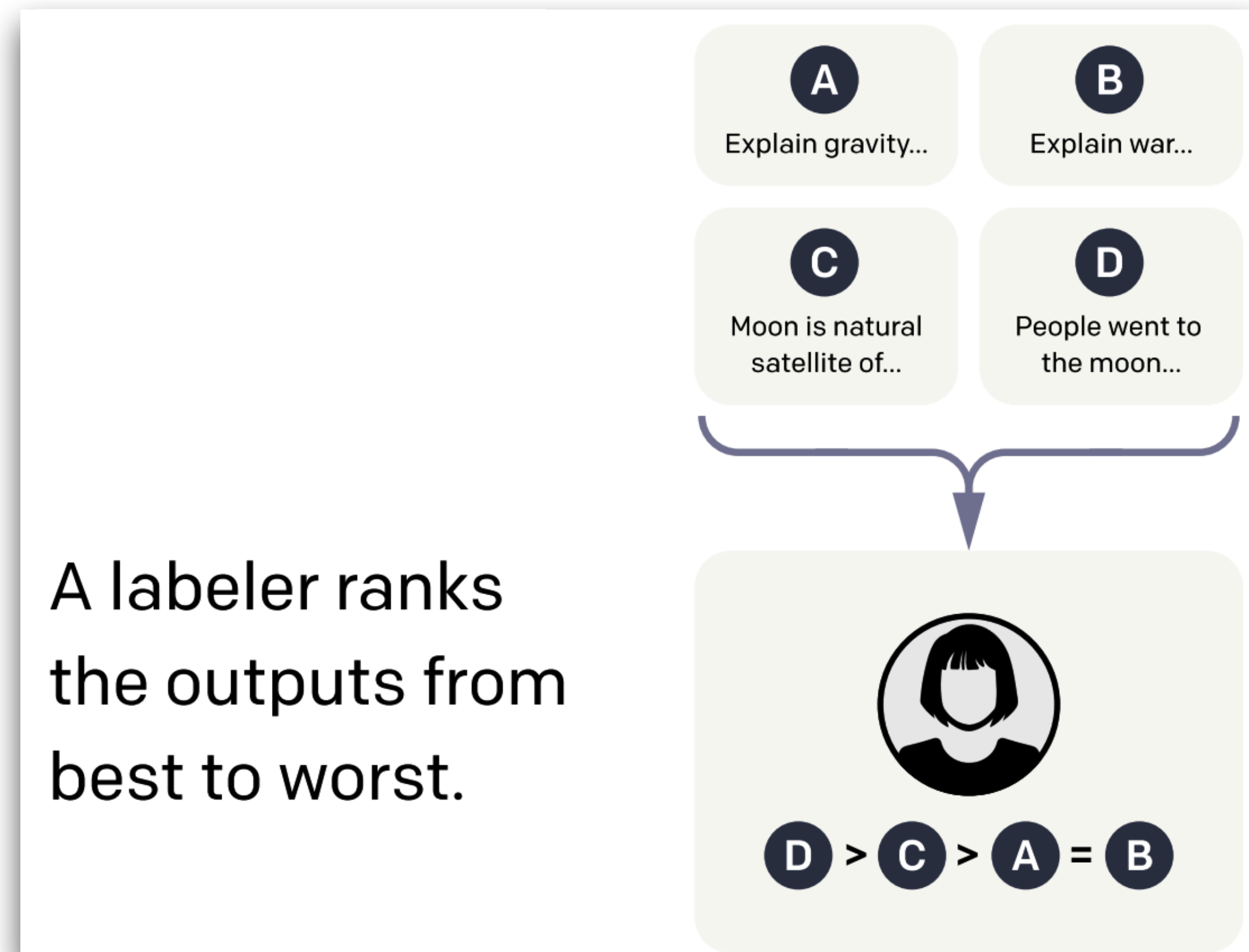
12/06/2024 @Berkeley

Human Feedback

- Learning from human feedback is useful [RLHF, inter alia]

Human Feedback

- Learning from human feedback is useful [RLHF, inter alia]

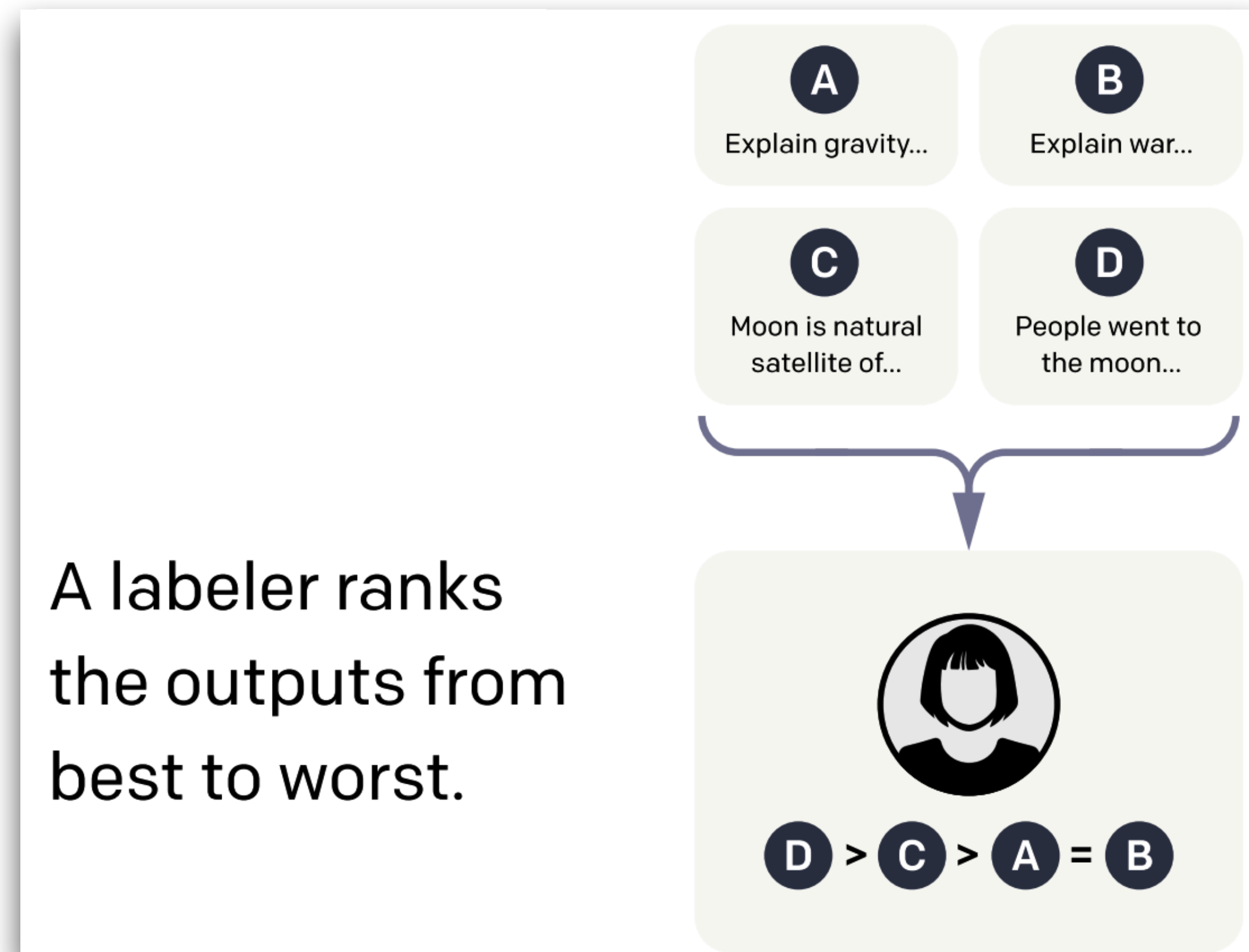


Human Feedback

annotator-provided

- Learning from ~~human~~ feedback is useful [RLHF, inter alia]

comparison-based



Human Feedback

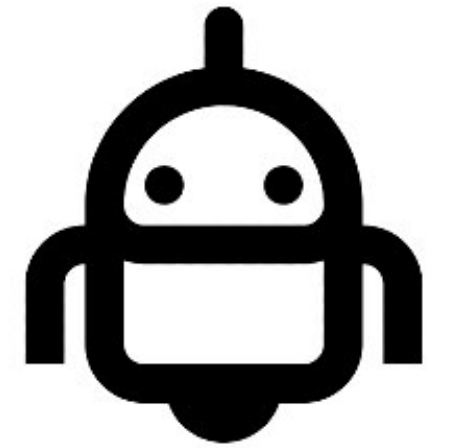
annotator-provided

- Learning from ~~human~~ feedback is useful [RLHF, inter alia]

comparison-based

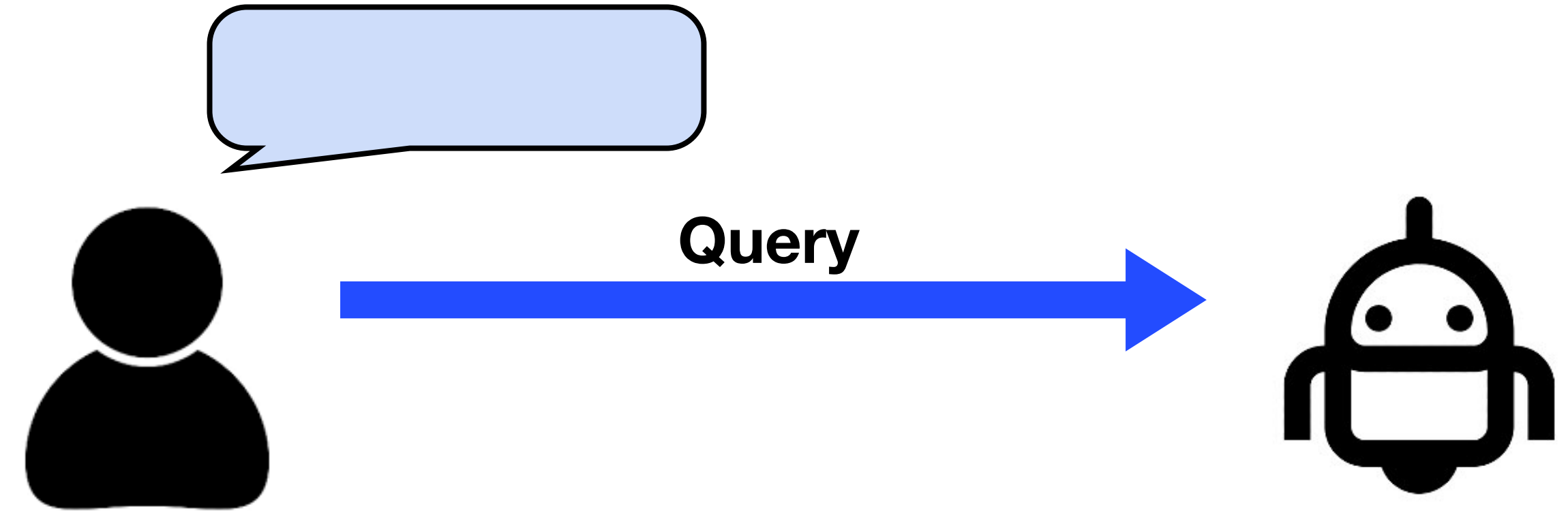
- But...
 - Annotations are expensive to collect
 - Comparison-based feedback rarely occurs in practice

Human Feedback in Practice



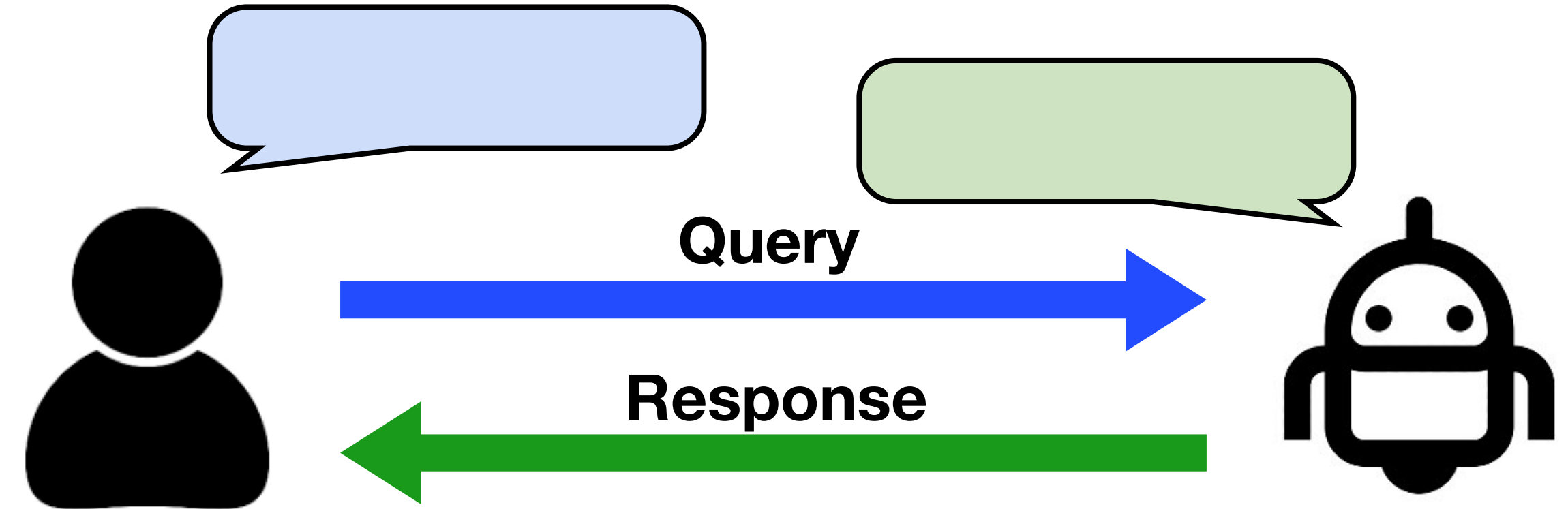
Human Feedback in Practice

- Agent interacts with a user



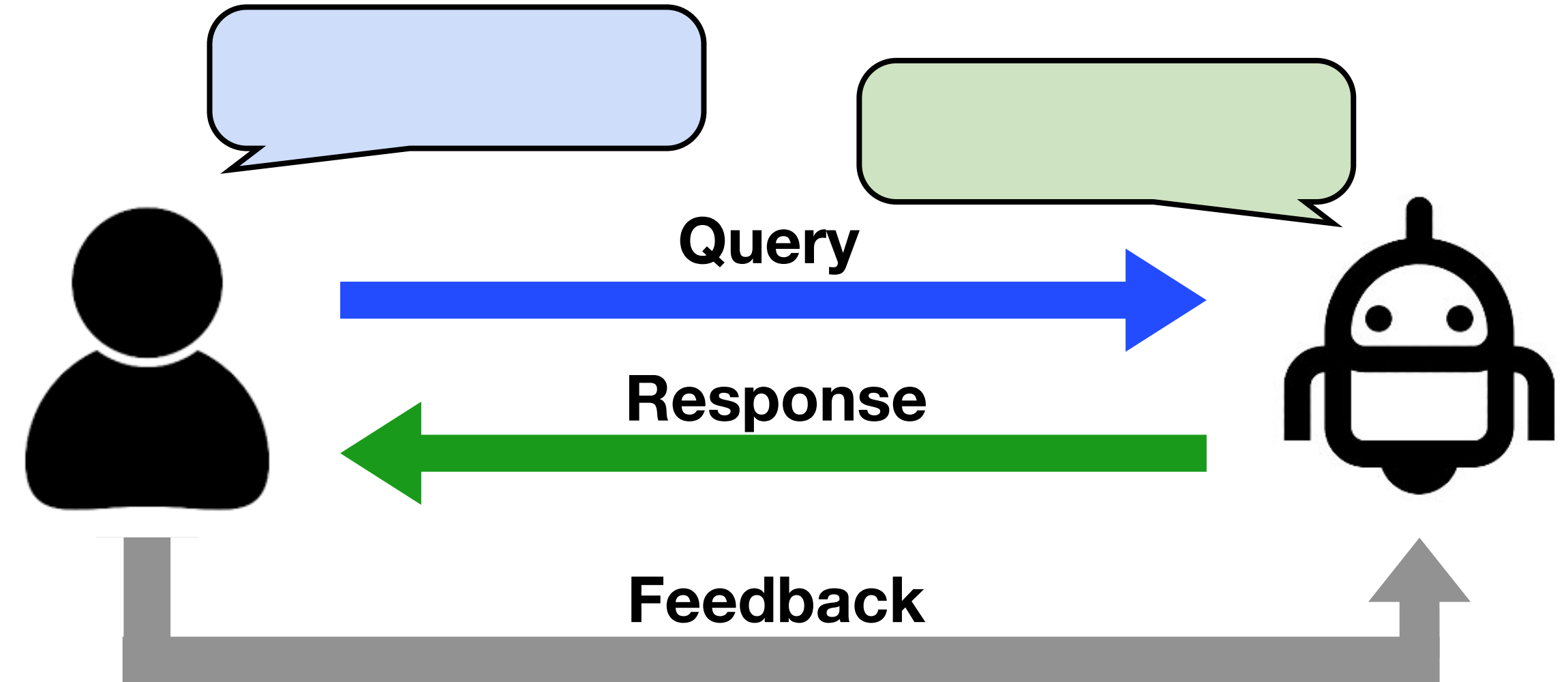
Human Feedback in Practice

- Agent interacts with a user
- Agent provides a single response



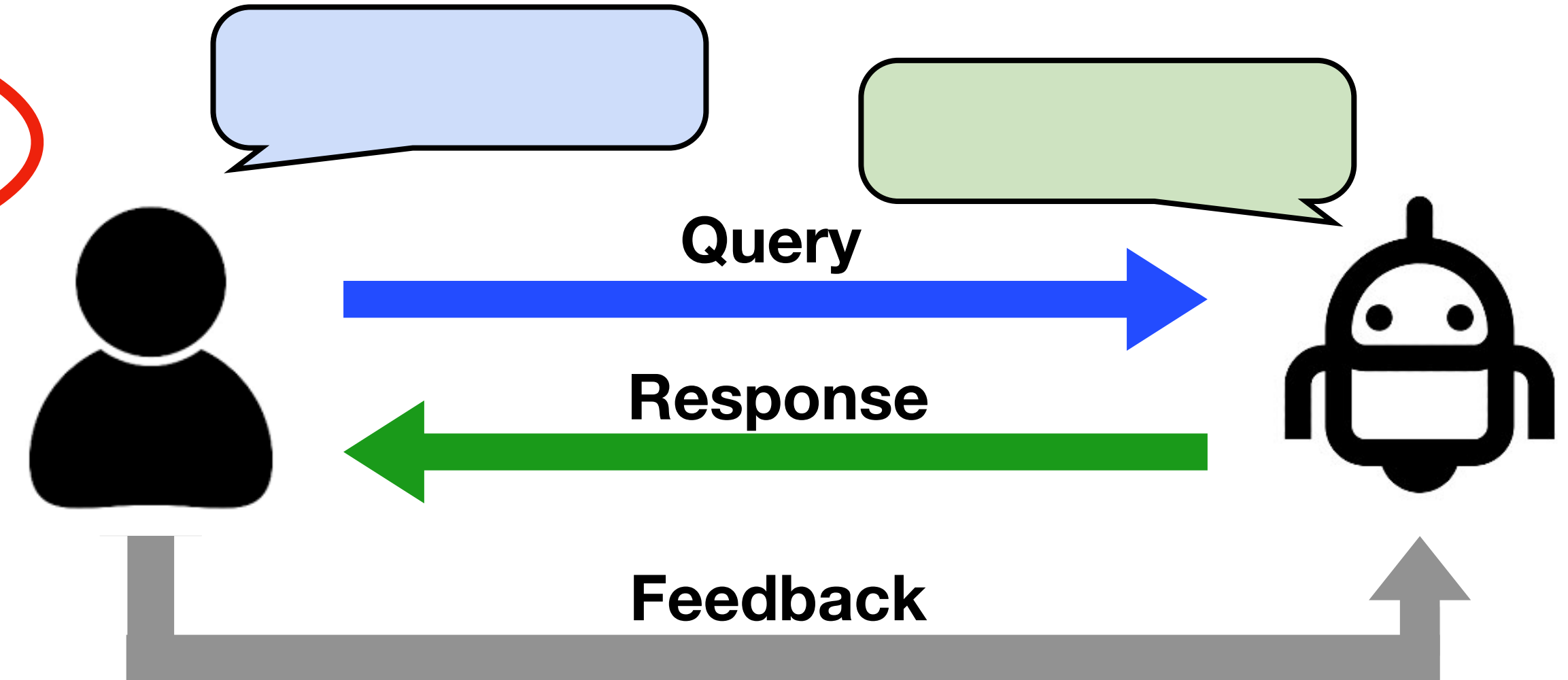
Human Feedback in Practice

- Agent interacts with a user
- Agent provides a single response
- Feedback occurs in various forms
 - Thumb up / down (explicit)
 - User rephrases the query (implicit)
 - ...



Outline: Learning from User Feedback

1. How to learn from naturally occurring human feedback?



Feedback to Writing Assistant

- The use of AI writing assistants is prevalent nowadays



Write me a ...

Feedback to Writing Assistant



- The use of AI writing assistants is prevalent nowadays
- Users often revise the agent response before own final use



Write me a ...

- Farming, as a part of agriculture, involves growing crops cultivation and animal rearing for food and raw materials.
- Originated It began thousands of years ago, likely in the Fertile Crescent, leading to the Neolithic Revolution
- Transition as people transitioned from nomadic hunting to settled farming. resulted in significant human population increase

Feedback to Writing Assistant

- The use of AI writing assistants is prevalent nowadays 
- Users often revise the agent response before own final use 
- **Every natural use of the agent yields an edit feedback for learning**
- Such feedback reflects the user's authentic expectation and individual preference, beyond the generic writing task

Research Question

- How to learn from user feedback in the form of edits?

Aligning LLM Agents by Learning Latent Preference from User Edits

Ge Gao^{♣*} **Alexey Taymanov**^{◇*} **Eduardo Salinas**[◇] **Paul Mineiro**[◇] **Dipendra Misra**[◇]
Department of Computer Science, Cornell University[♣] Microsoft Research New York[◇]
ggao@cs.cornell.edu {ataymano, edus, pmineiro, dimisra}@microsoft.com

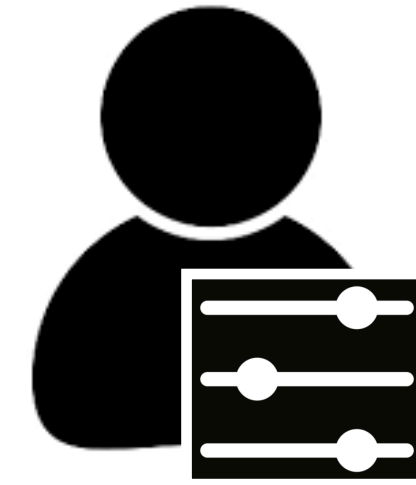
Research Question

- How to learn from user feedback in the form of edits?
 - Infer latent user preference based on edits feedback
 - Generate satisfactory responses that align with the user's need
 - Take account of user's efforts on making such edits



correct preference induction → satisfactory response → few user edits

Challenges



- User preference is multifaceted and complex
- Some preference is context-dependent, and may even vary over time
- Feedback in the form of edits is implicit
 - lacking direct expressions of the underlying preference
 - may lead to diverse interpretations

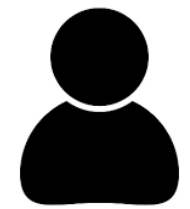
Outline: Our Contribution

- A framework that formulates the interaction process and learning problem
- A method that implements this framework for LLM agents
- Highlights from experimental results and analysis

Learning Framework

Round t ① User provides a context x_t to the LLM agent

Article: {user-provided article}
Please summarize the above article.



Learning Framework

Round t

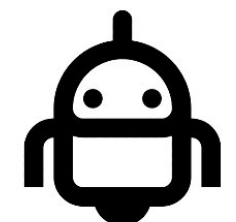
① User provides a context x_t to the LLM agent

Article: {user-provided article}
Please summarize the above article.



② LLM agent generates a response y_t given the context

Farming, a part of agriculture, involves growing crops and rearing animals for food and raw materials. It began thousands of years ago, likely in the Fertile Crescent, and led to the Neolithic Revolution as people transitioned from nomadic hunting to settled farming. This allowed for a significant increase in human population.




Learning Framework

Round t

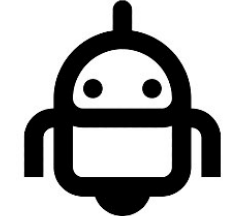
① User provides a context x_t to the LLM agent

Article: {user-provided article}
Please summarize the above article.




② LLM agent generates a response y_t given the context

Farming, a part of agriculture, involves growing crops and rearing animals for food and raw materials. It began thousands of years ago, likely in the Fertile Crescent, and led to the Neolithic Revolution as people transitioned from nomadic hunting to settled farming. This allowed for a significant increase in human population.

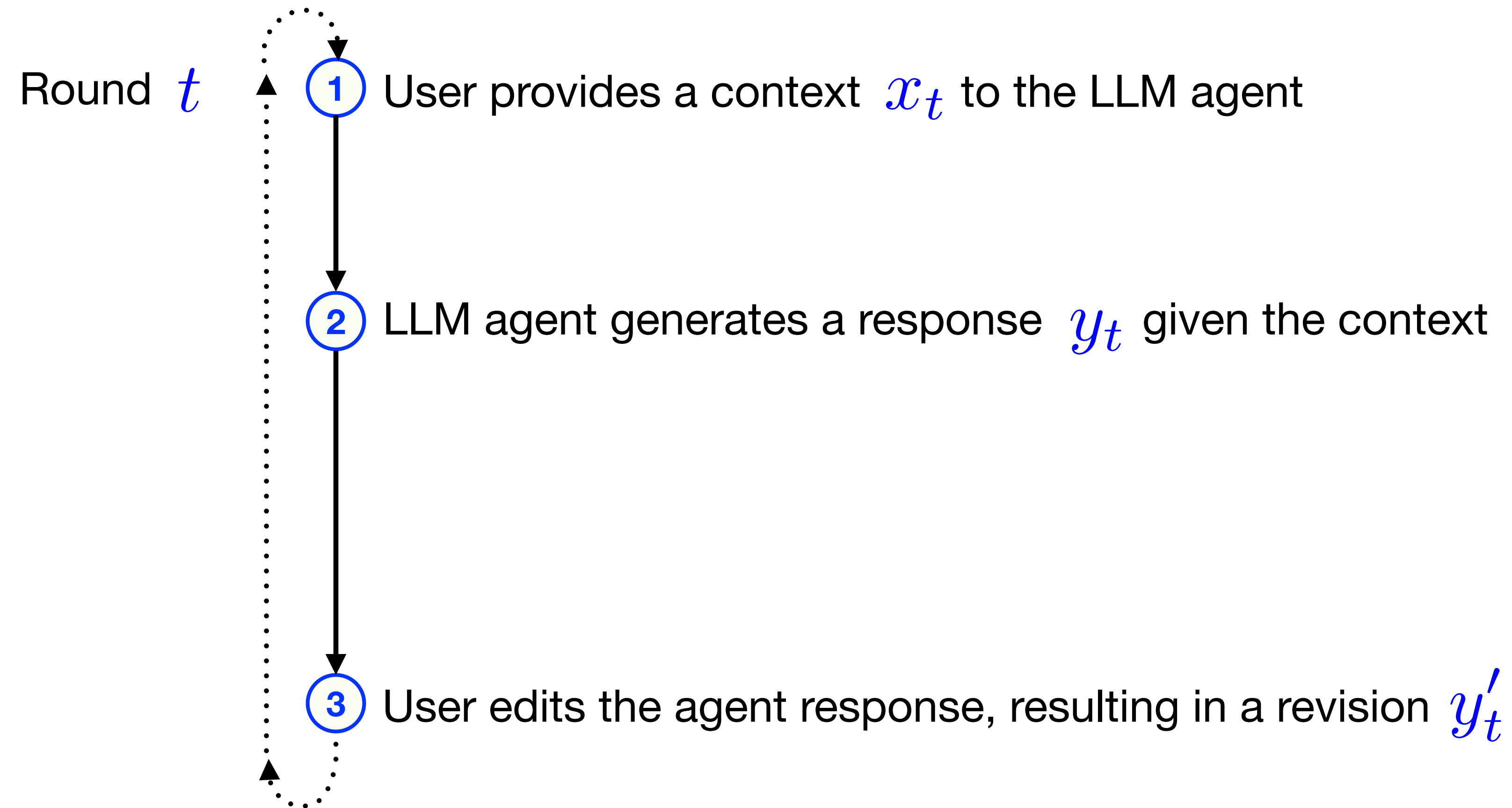


③ User edits the agent response, resulting in a revision y'_t

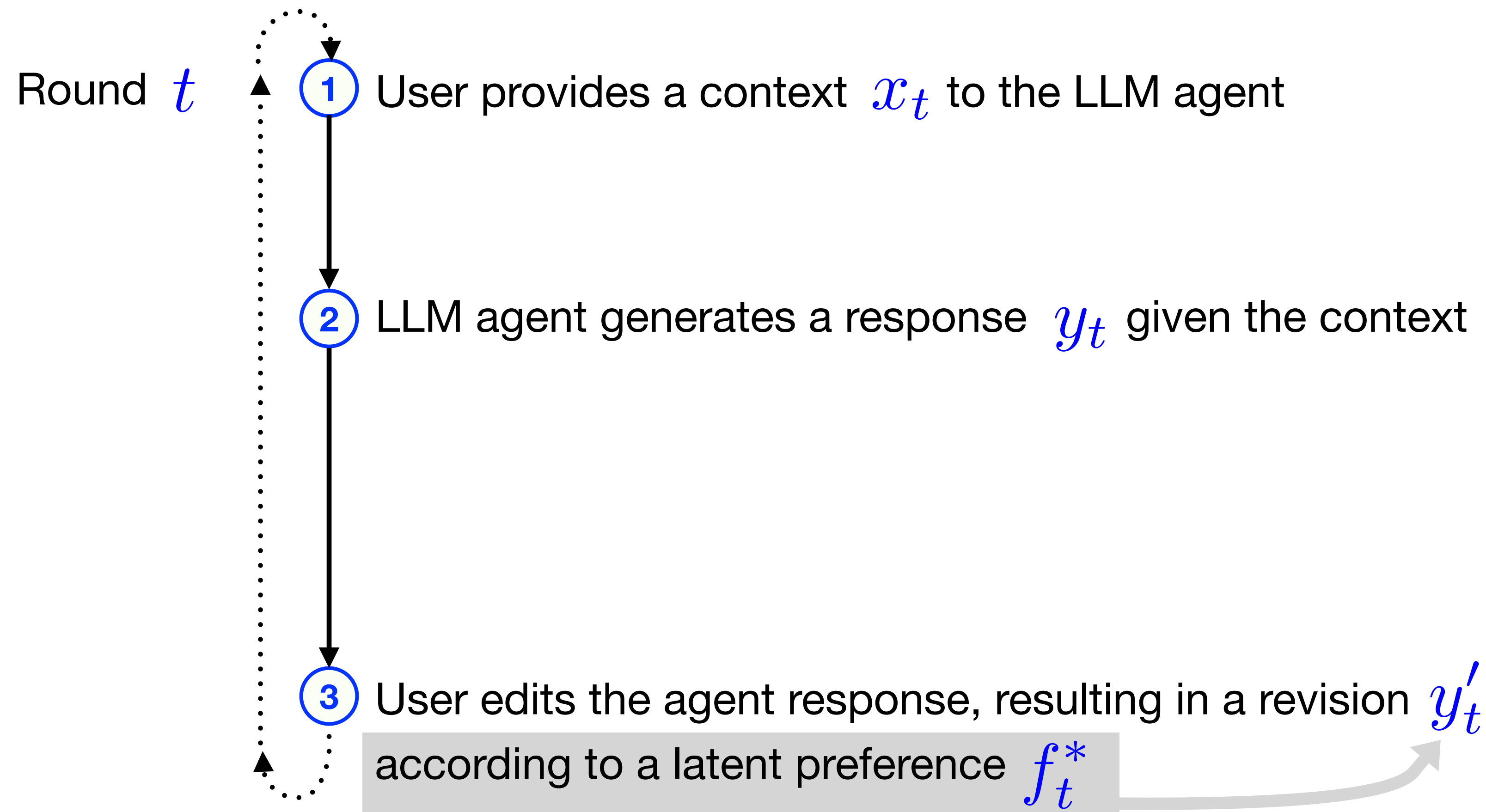
- Farming, as a part of agriculture, involves growing crops cultivation and animal rearing for food and raw materials.
- Originated ~~It began~~ thousands of years ago, likely in the Fertile Crescent, leading to the Neolithic Revolution
- Transition ~~as people transitioned~~ from nomadic hunting to settled farming. resulted in significant human population increase



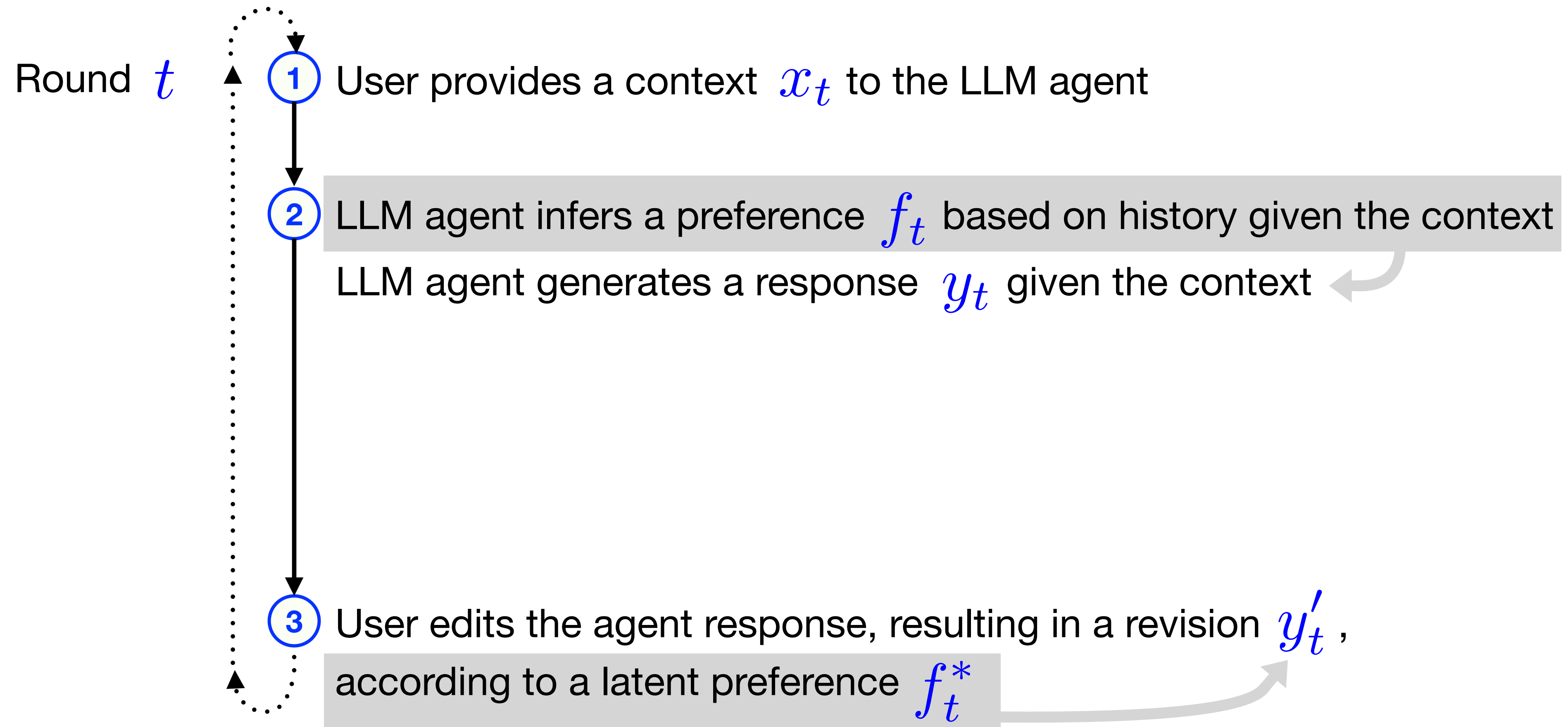
Learning Framework



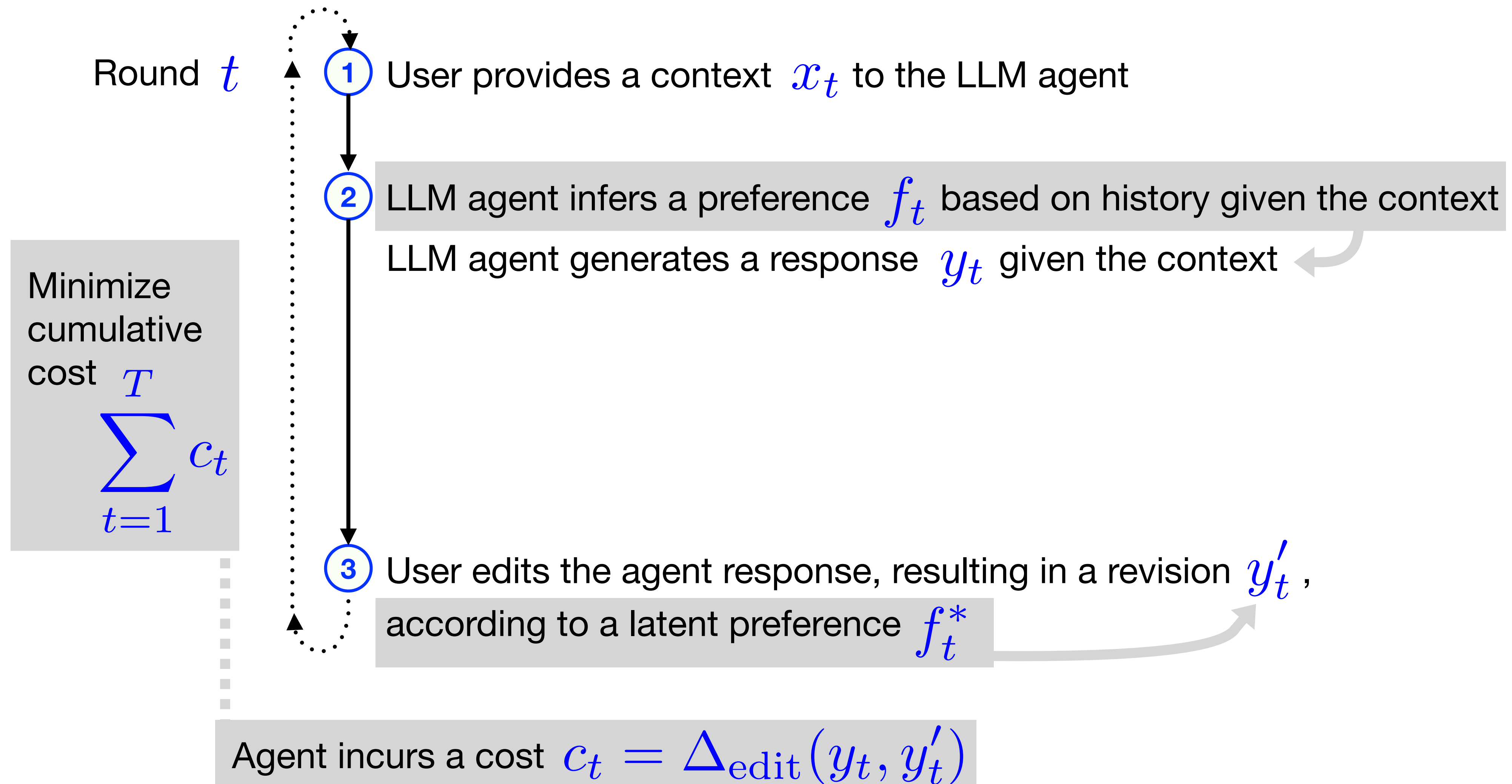
Learning Framework



Learning Framework



Learning Framework



Learning Framework

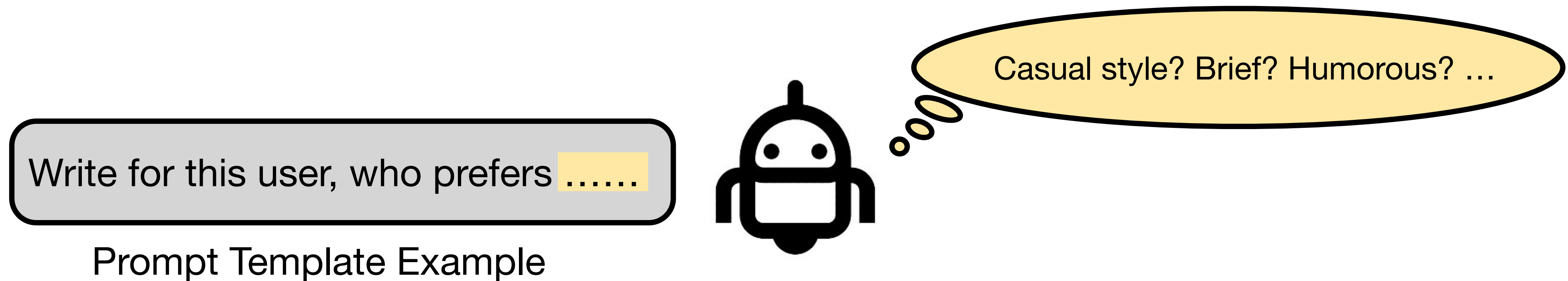
- We formulate the interaction process and preference learning problem as **PRELUDE** (**PRE**ference **L**earning from **U**ser's **D**irect **E**ditions)
- Assume that the user directly makes edits to the agent response based on a latent preference
- Agent infers a user preference from the interaction history, and uses it to generate a response
- Cost minimization to account for the amount of efforts spent by the user on making edits

Method

- Agent leverages LLMs by prompting
- We learn a prompt policy that can infer a descriptive user preference, and then use it in the prompt to directly drive the response generation

Method

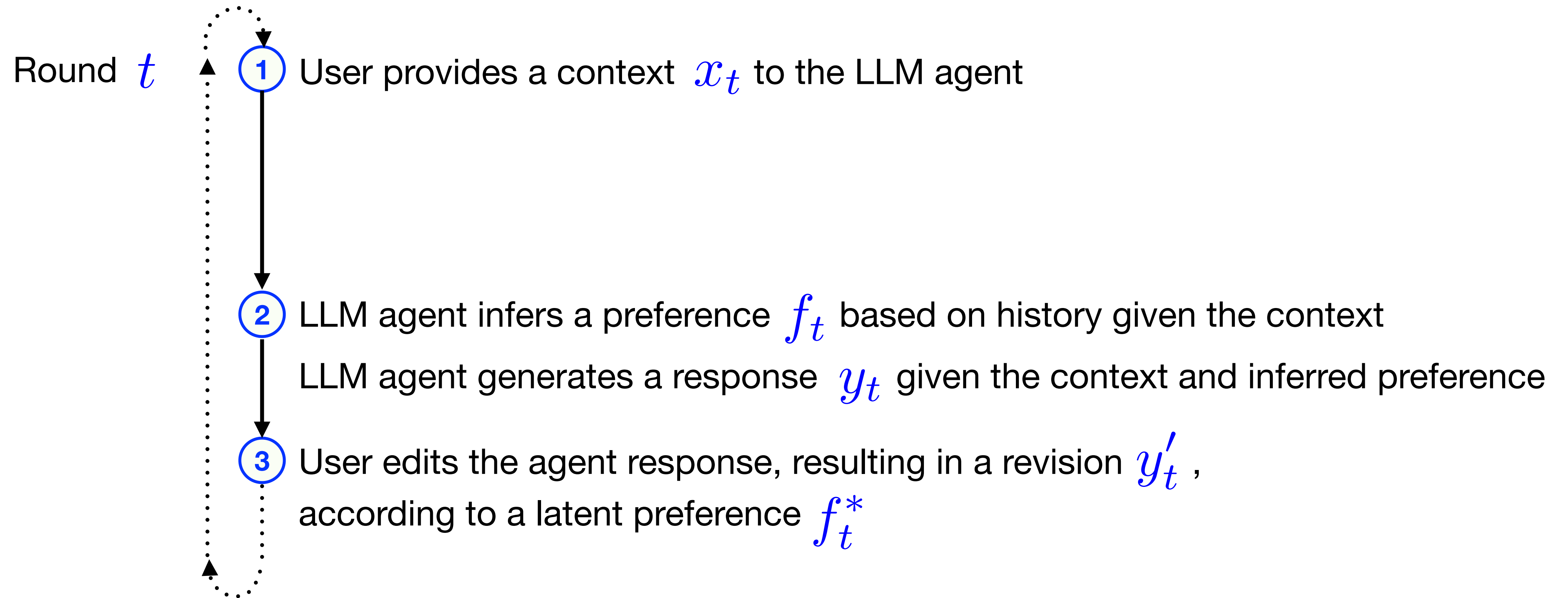
- Agent leverages LLMs by prompting
- We learn a prompt policy that can infer a descriptive user preference, and then use it in the prompt to directly drive the response generation



Method

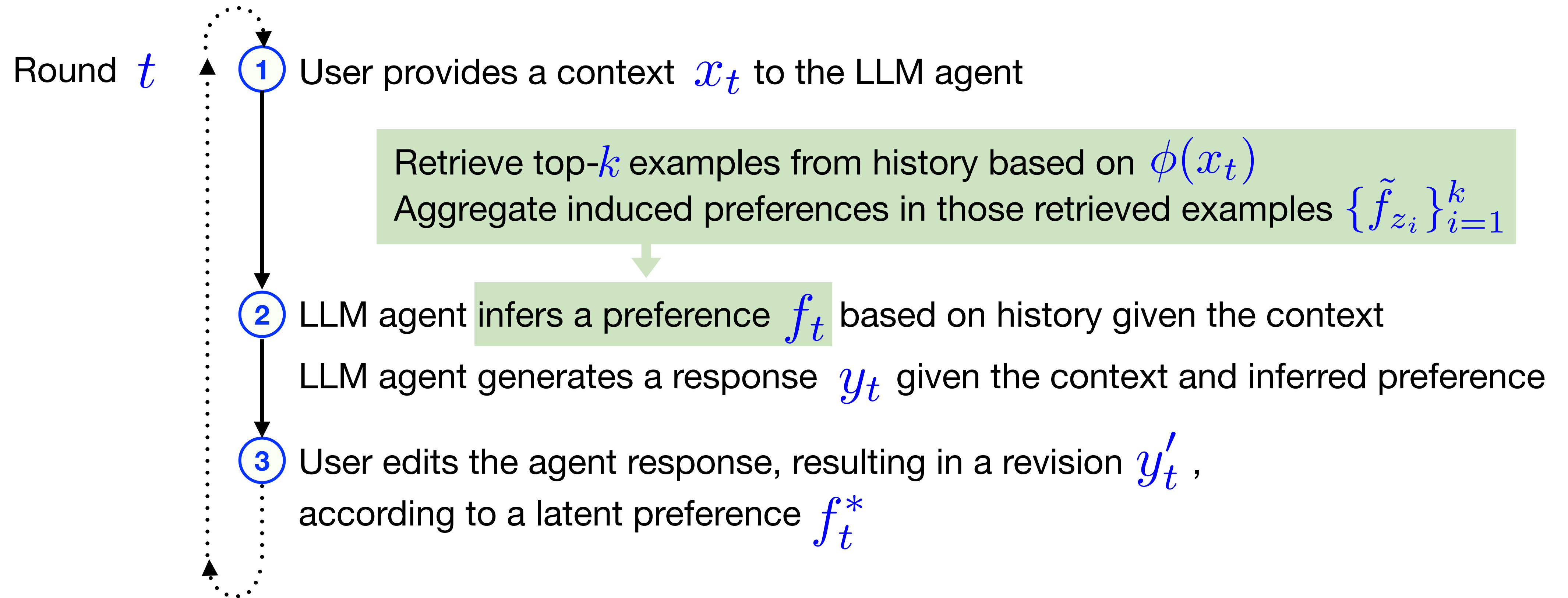
- Agent leverages LLMs by prompting
- We learn a prompt policy that can infer a descriptive user preference, and then use it in the prompt to directly drive the response generation
 - When user makes edits, induce a description of the user preference
 - Manage a collection of preference history
 - Given a new context, infer a descriptive preference based on retrieving similar contexts from the history

Method



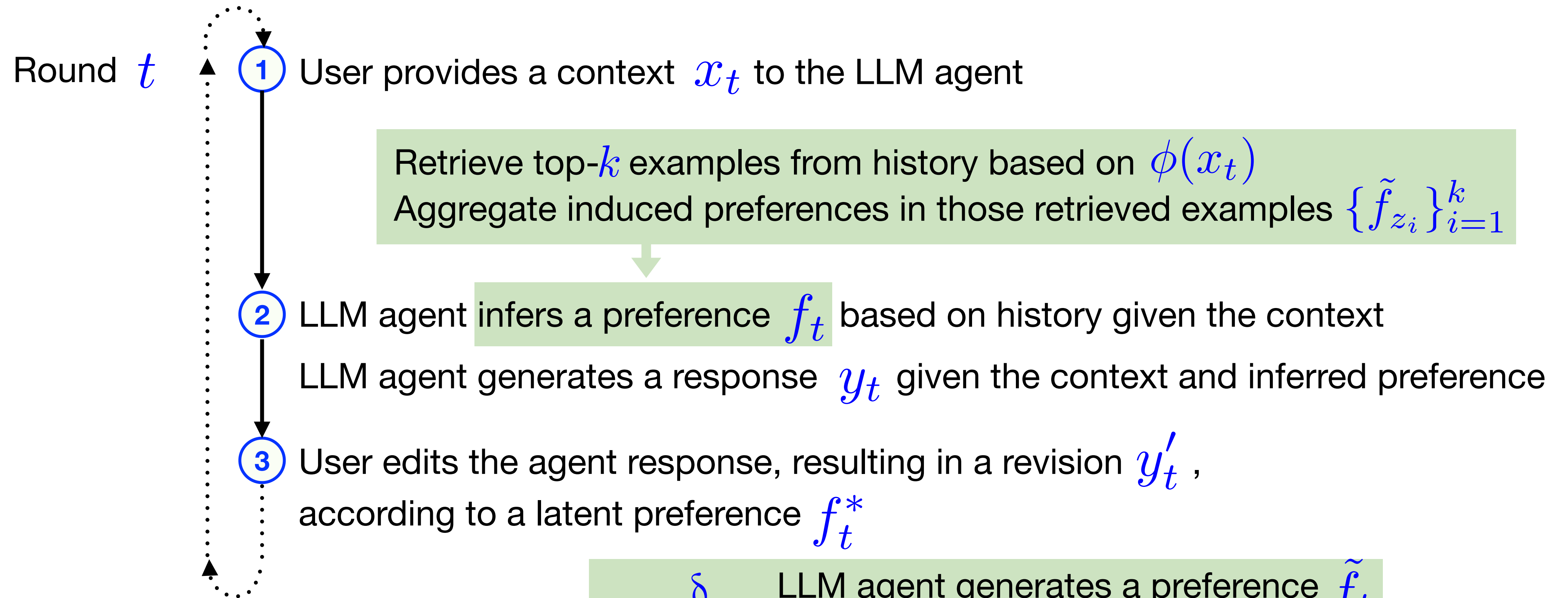
Agent incurs a cost $c_t = \Delta_{\text{edit}}(y_t, y'_t)$

Method



Agent incurs a cost $c_t = \Delta_{\text{edit}}(y_t, y'_t)$

Method

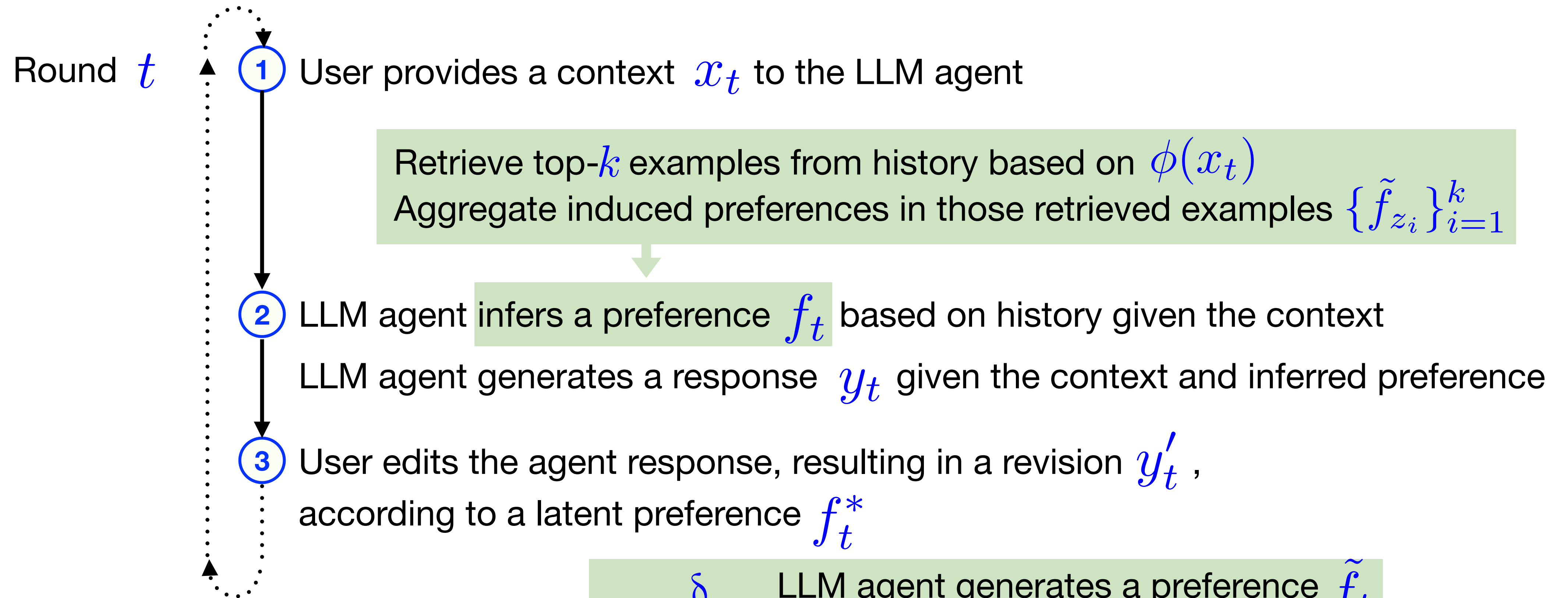


Retrieve top- k examples from history based on $\phi(x_t)$
Aggregate induced preferences in those retrieved examples $\{\tilde{f}_{z_i}\}_{i=1}^k$

Agent incurs a cost $c_t = \Delta_{\text{edit}}(y_t, y'_t)$

$c_t \geq \delta$ → LLM agent generates a preference \tilde{f}_t to explain the user edits
 $c_t < \delta$ → $\tilde{f}_t \leftarrow f_t$

Method



Retrieve top- k examples from history based on $\phi(x_t)$
Aggregate induced preferences in those retrieved examples $\{\tilde{f}_{z_i}\}_{i=1}^k$

Agent incurs a cost $c_t = \Delta_{\text{edit}}(y_t, y'_t)$


$c_t \geq \delta$ LLM agent generates a preference \tilde{f}_t to explain the user edits
 $c_t < \delta$ $\tilde{f}_t \leftarrow f_t$

History $D \leftarrow D \cup \{(\phi(x_t), \tilde{f}_t)\}$

Method

- **CIPHER** (**C**onsolidates **I**nduced **P**references based on **H**istorical **E**dits with **R**etrieval)
- Computationally efficient
 - 4 LLM calls at max per interaction; only a small increase in prompt length
 - Low memory storage: save context representation instead of the context itself
- User-friendly and interpretable
 - Users are not required to do heavy prompt engineering
 - Users could read and understand the preference learned by the agent

Task & User Setup

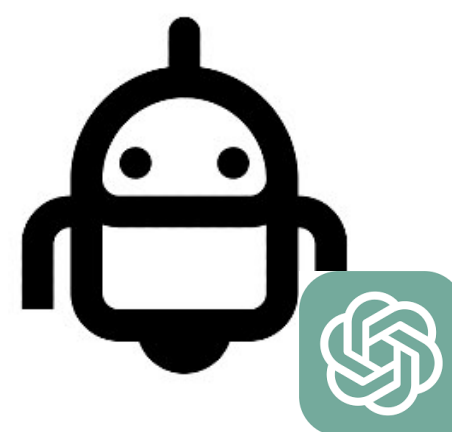
- Writing task for the agent: summarize a document
- GPT-4 user as a simulation 
 - Provide a context (i.e., specify the writing task, includes a document)
 - Can provide context documents from different sources
 - Have context-depend preference for different use cases

Task & User Setup

| Use Case | Latent User Preference | Doc Source |
|--|--|----------------|
| Introduce a political news to kids | targeted to young children, storytelling, short sentences, playful language, interactive, positive | News article |
| Promote a paper to invoke more attention and interests | tweet style, simple English, inquisitive, skillful foreshadowing, with emojis | Paper abstract |
| Take notes for factual knowledge | bullet points, parallel structure, brief | Wikipedia page |
| Use online stories to inspire character developments in creative writing | second person narrative, brief, show emotions, invoke personal reflection, immersive | Reddit post |
| Extract main opinions from a review | question answering style | Movie review |

Experimental Setup

- 200 interactions in total $T = 200$; different context per round
- Implementation details of CIPHER
 - GPT-4 as the base LLM
 - MPNeT as the context representation function $\phi = \text{MPNet}$
 - Top 5 retrieval with cosine similarity $k = 5$
- Evaluation metrics
 - Cumulative Levenshtein edit distance: removal, insertion, or substitution (BPE tokens)
 - Expense of using LLM: total number of input and output BPE tokens



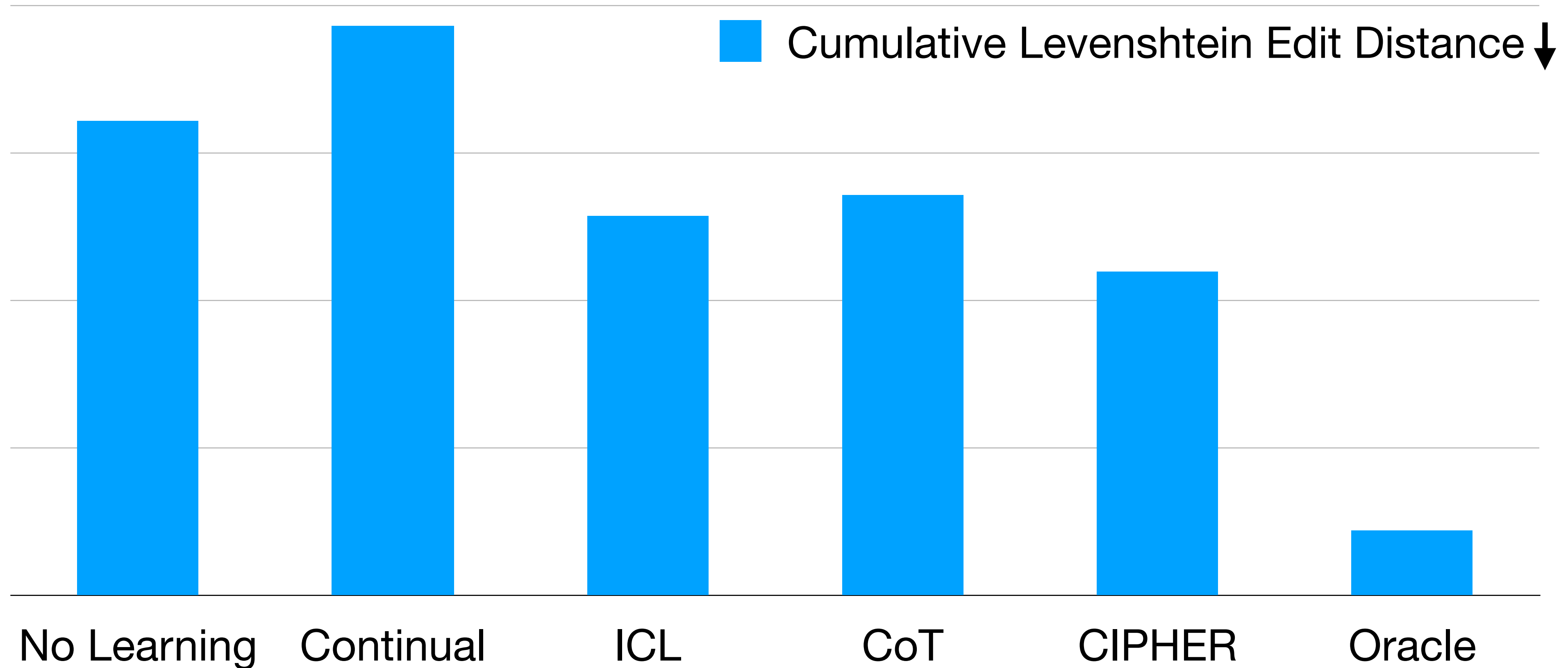
Experimental Setup

- Comparison systems

Interpretable **Context-dependent**

- No Learning: does not perform any preference learning
- Continual Learning: infer a preference using the most recent k interactions ✓
- In-Context Learning: retrieve top k historical examples, and use them as demonstration examples in the prompt for response generation ✓
- Chain-of-Thought: the prompt for response generation specifies two steps: 1) infer a descriptive user preference based on retrieved top k examples, and 2) generate a response accordingly ✓
- Oracle: let the agent use the true latent preference to generate a response ✓

Experimental Result



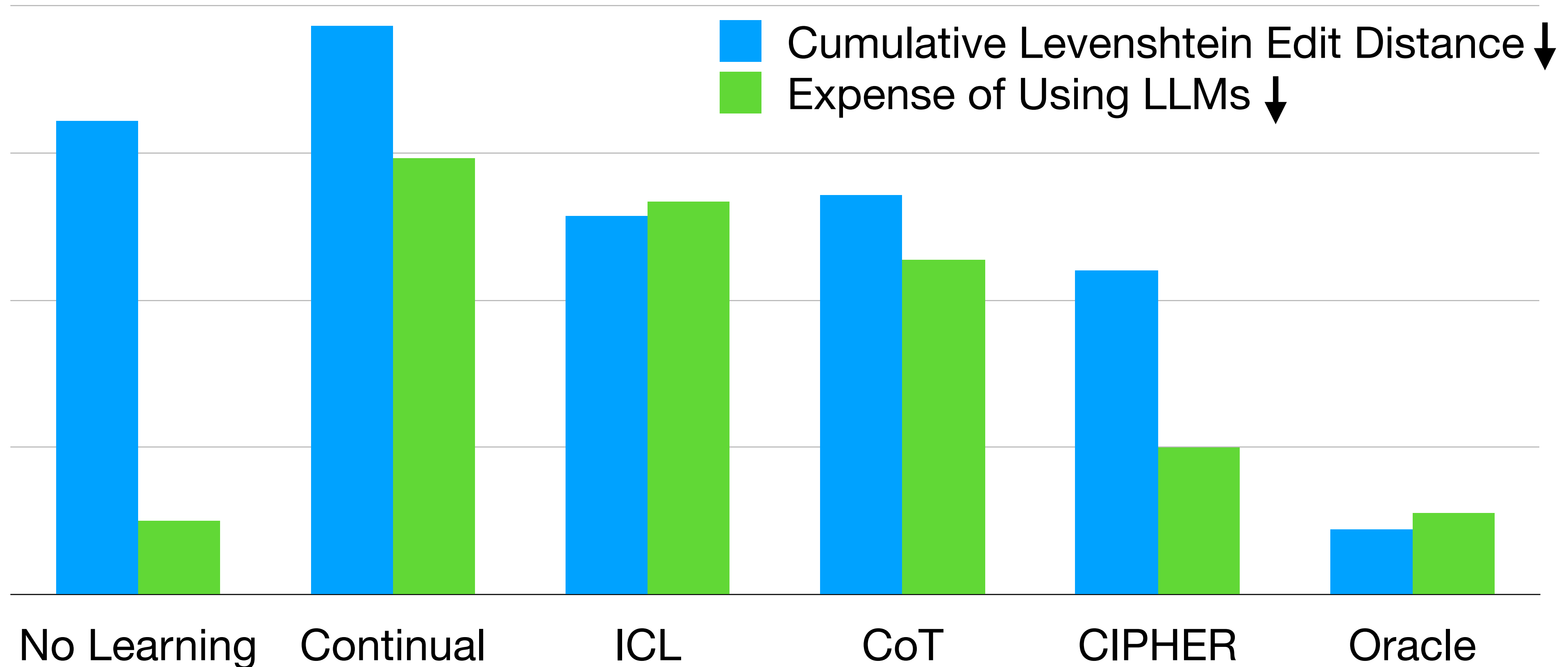
Interpretable



Context-dependent



Experimental Result



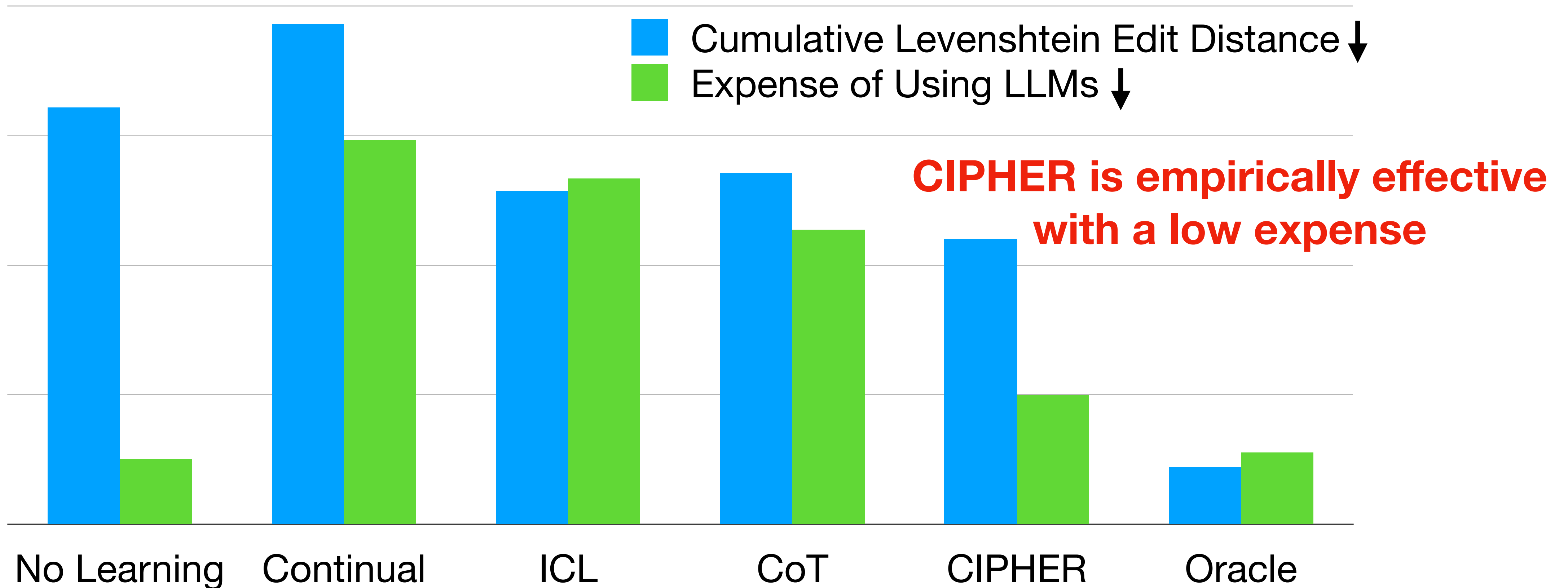
Interpretable



Context-dependent



Experimental Result



CIPHER is empirically effective with a low expense

Interpretable



Context-dependent

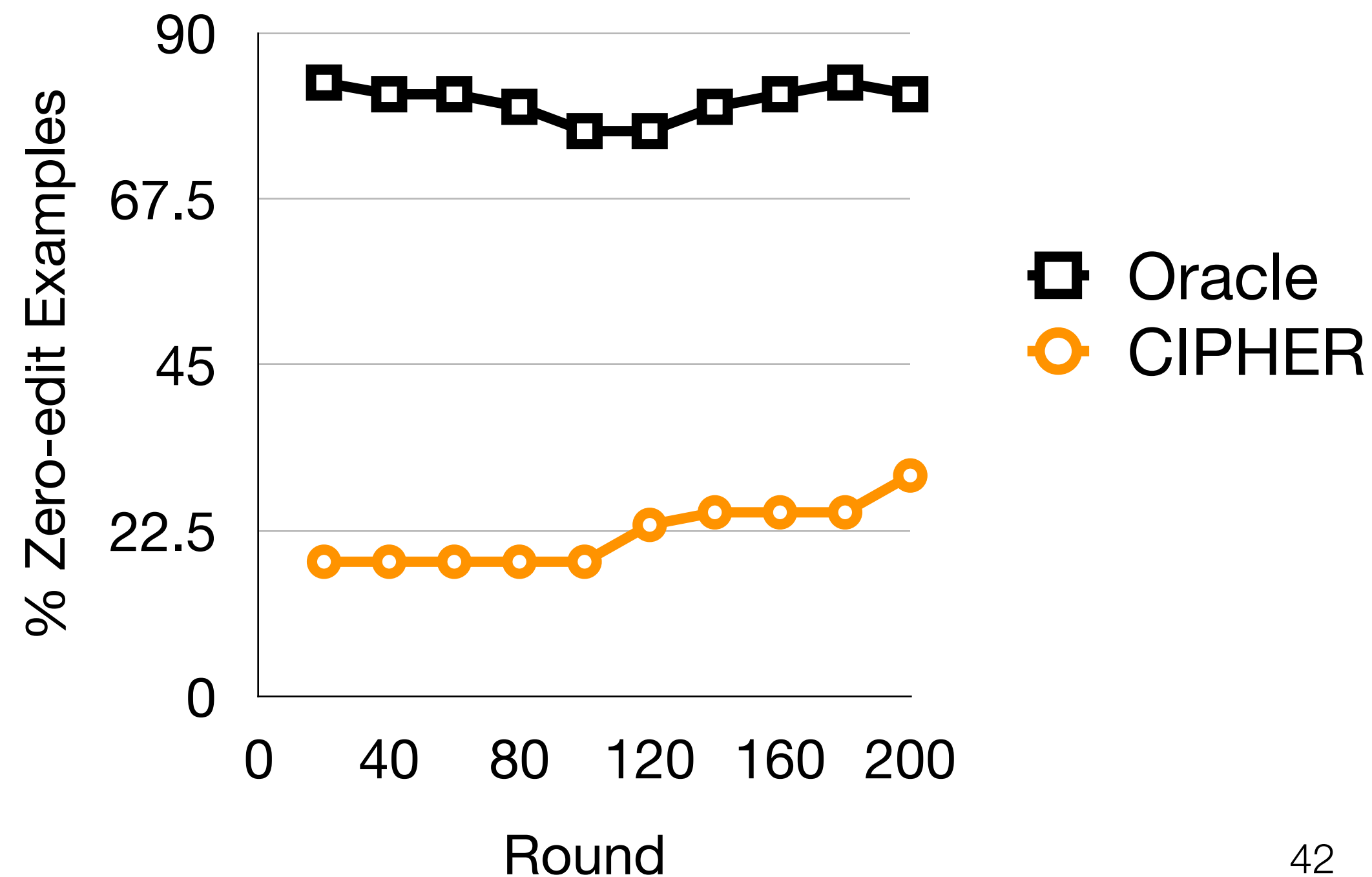


Experimental Analysis

- Does the user make fewer edits to CIPHER over time?

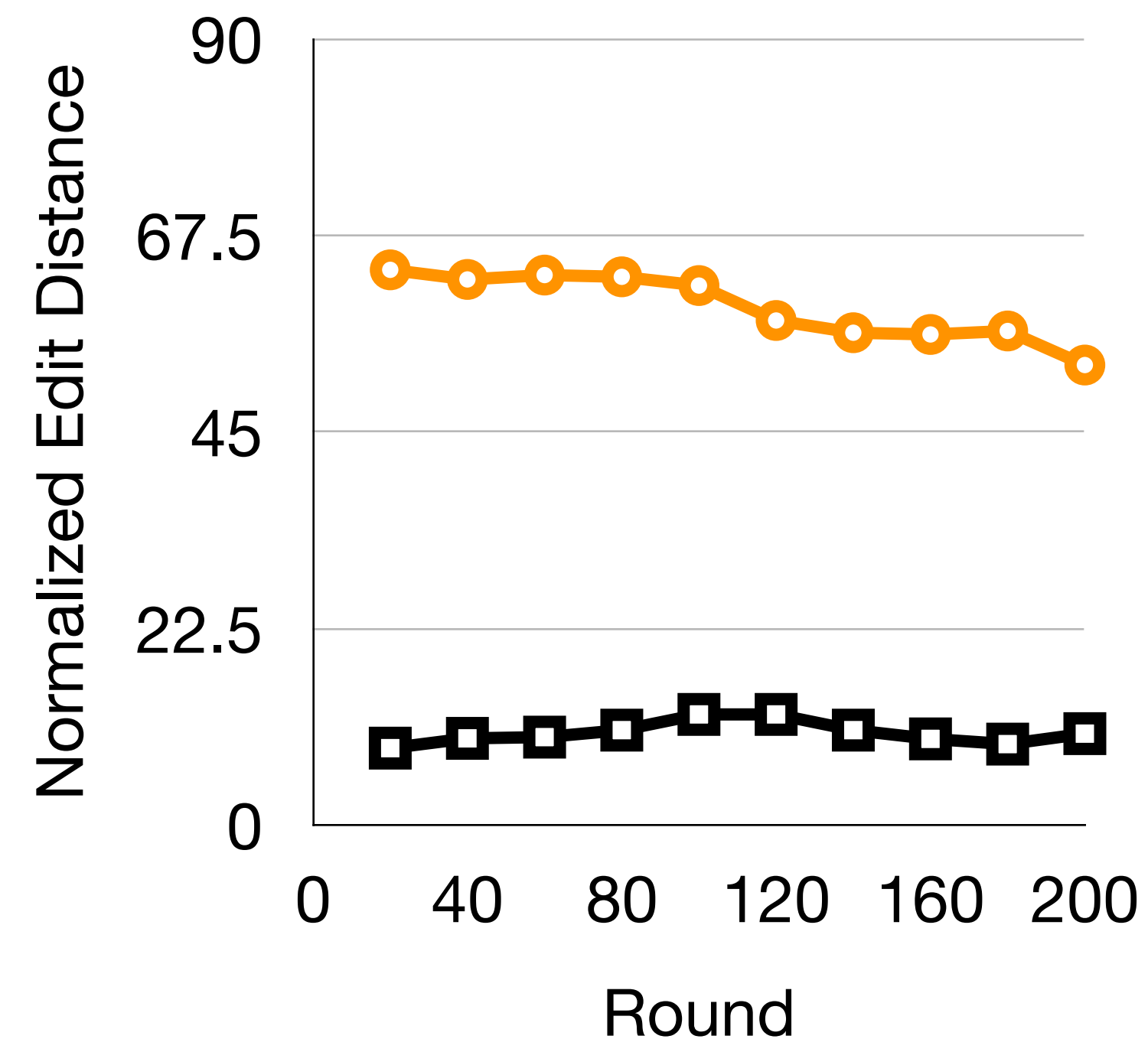
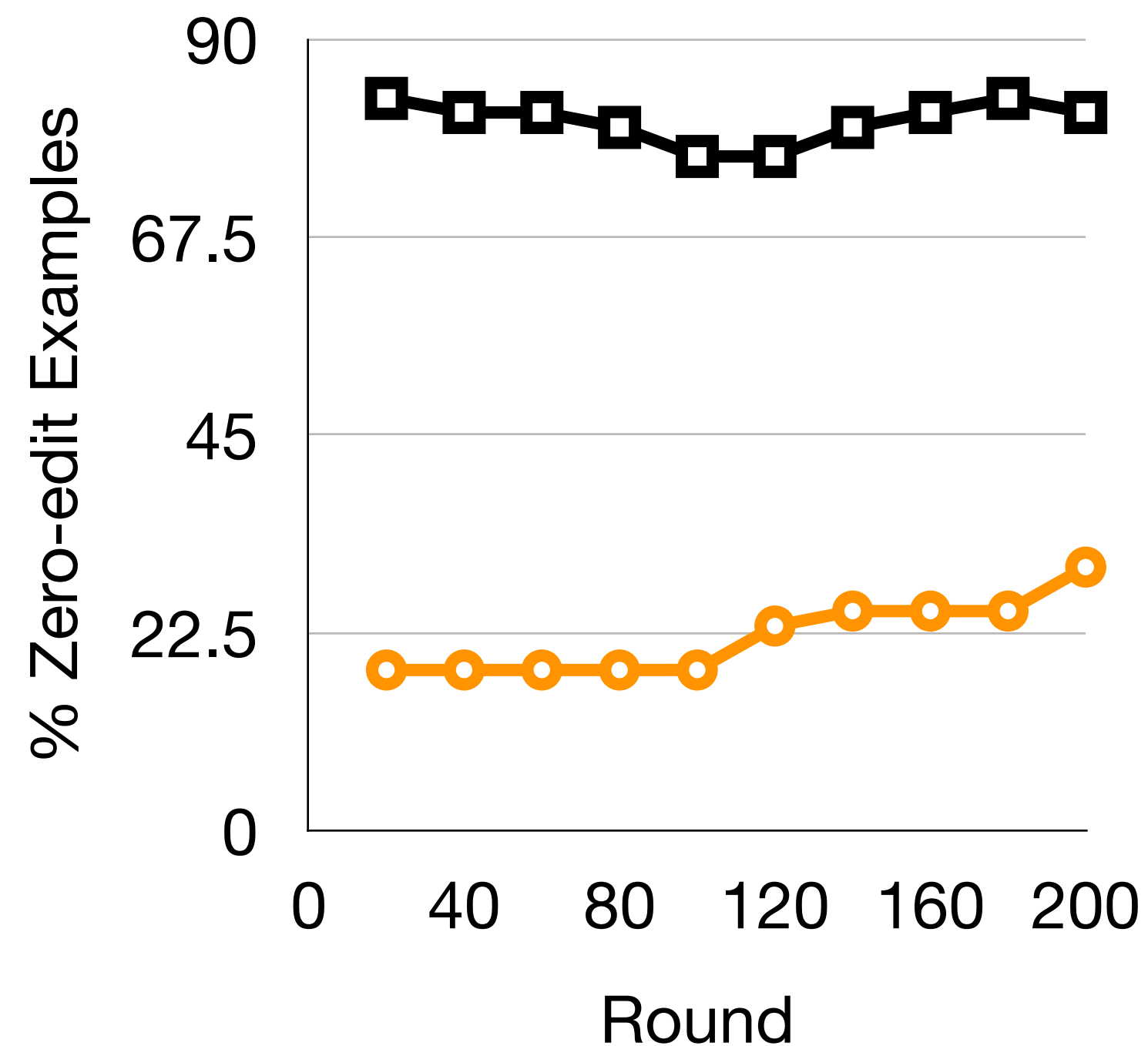
Experimental Analysis

- Does the user make fewer edits to CIPHER over time?
 1. Percentage of the zero-edit examples (binned per 20 rounds) ↑



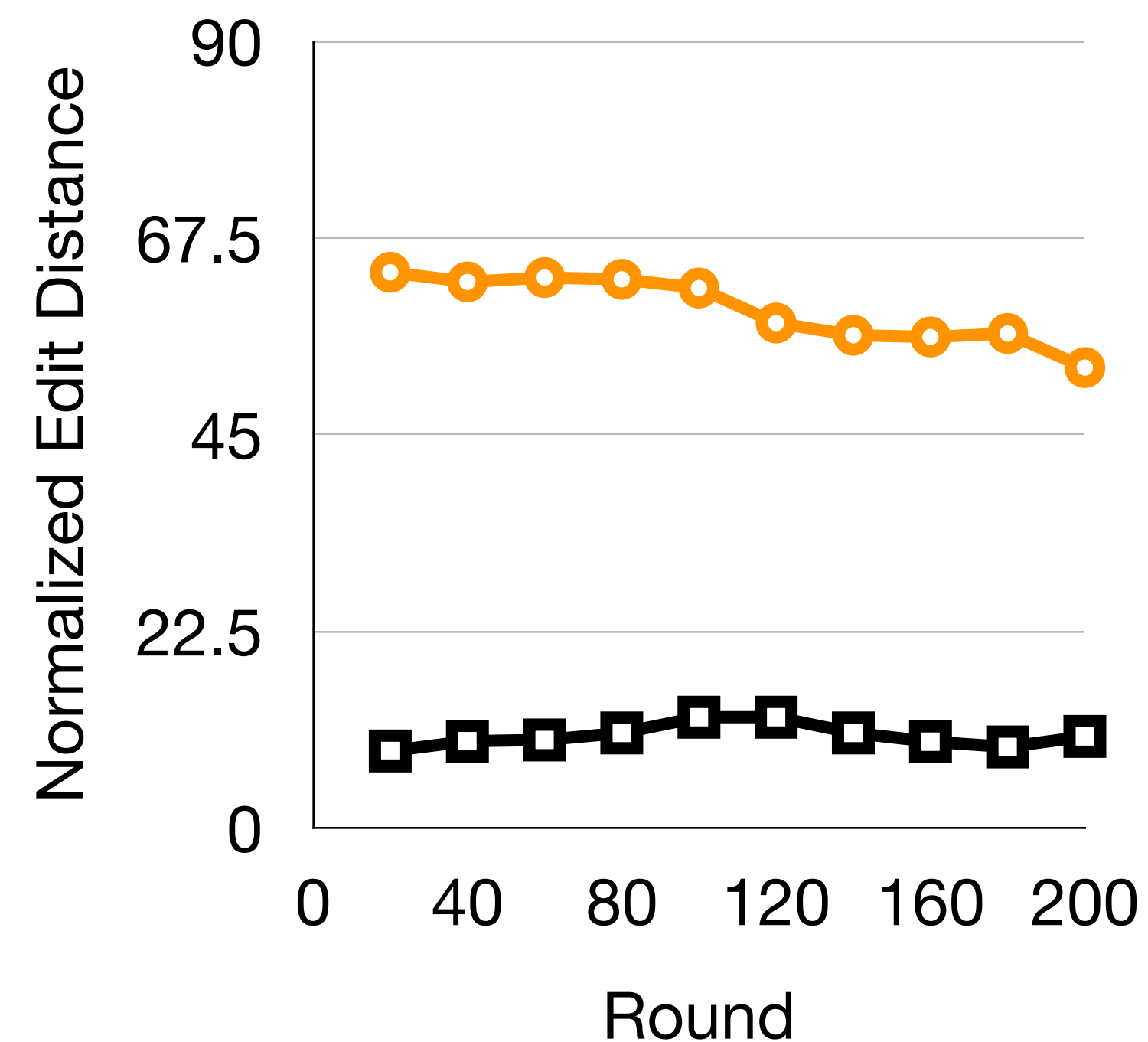
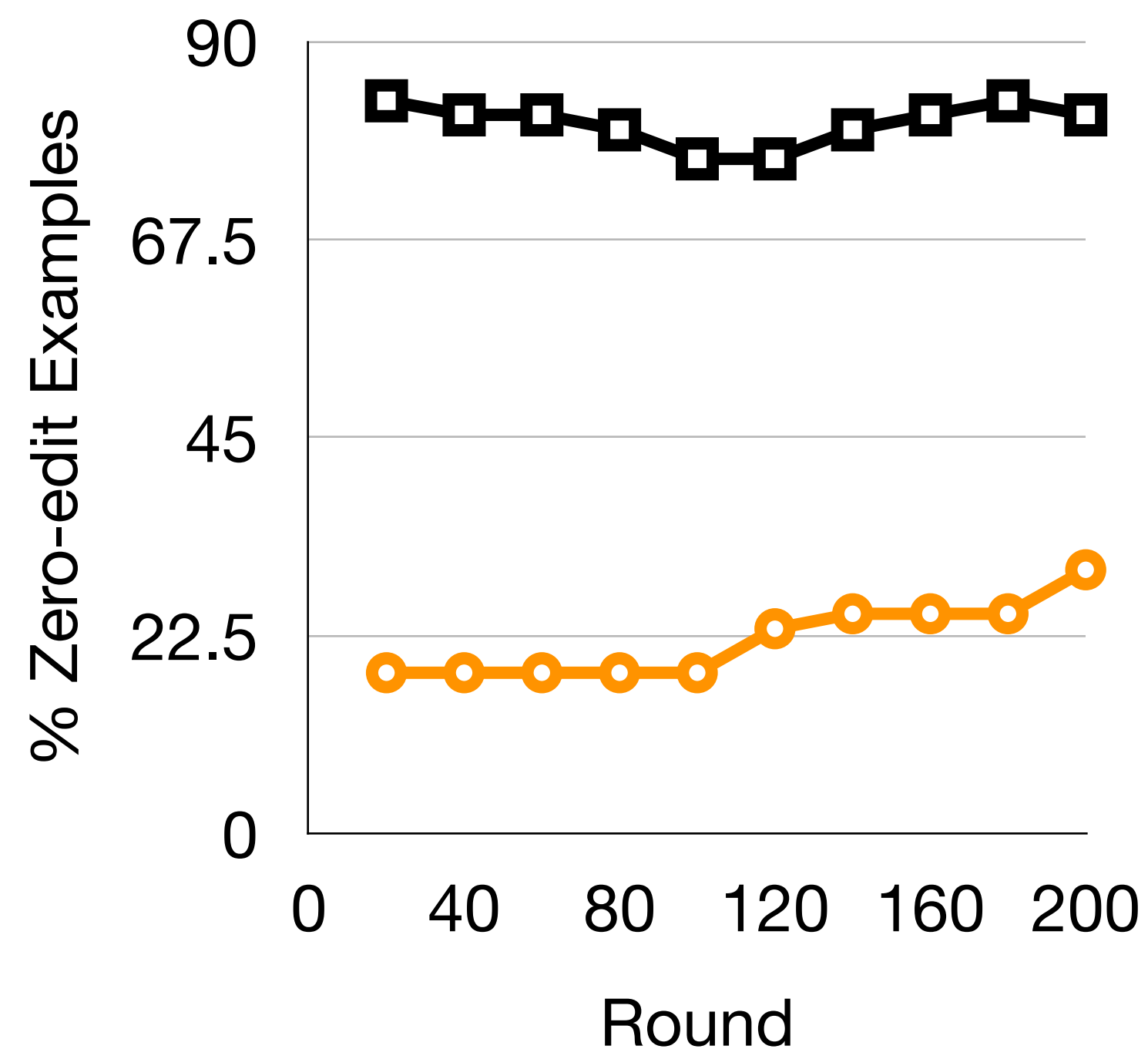
Experimental Analysis

- Does the user make fewer edits to CIPHER over time?
 1. Percentage of the zero-edit examples (binned per 20 rounds) ↑
 2. Edit distance normalized by the response length (averaged per 20 rounds) ↓



Experimental Analysis


- Does the user make fewer edits to CIPHER over time? **Yes!**
 - Percentage of the zero-edit examples (binned per 20 rounds) \uparrow
 - Edit distance normalized by the response length (averaged per 20 rounds) \downarrow



Summary

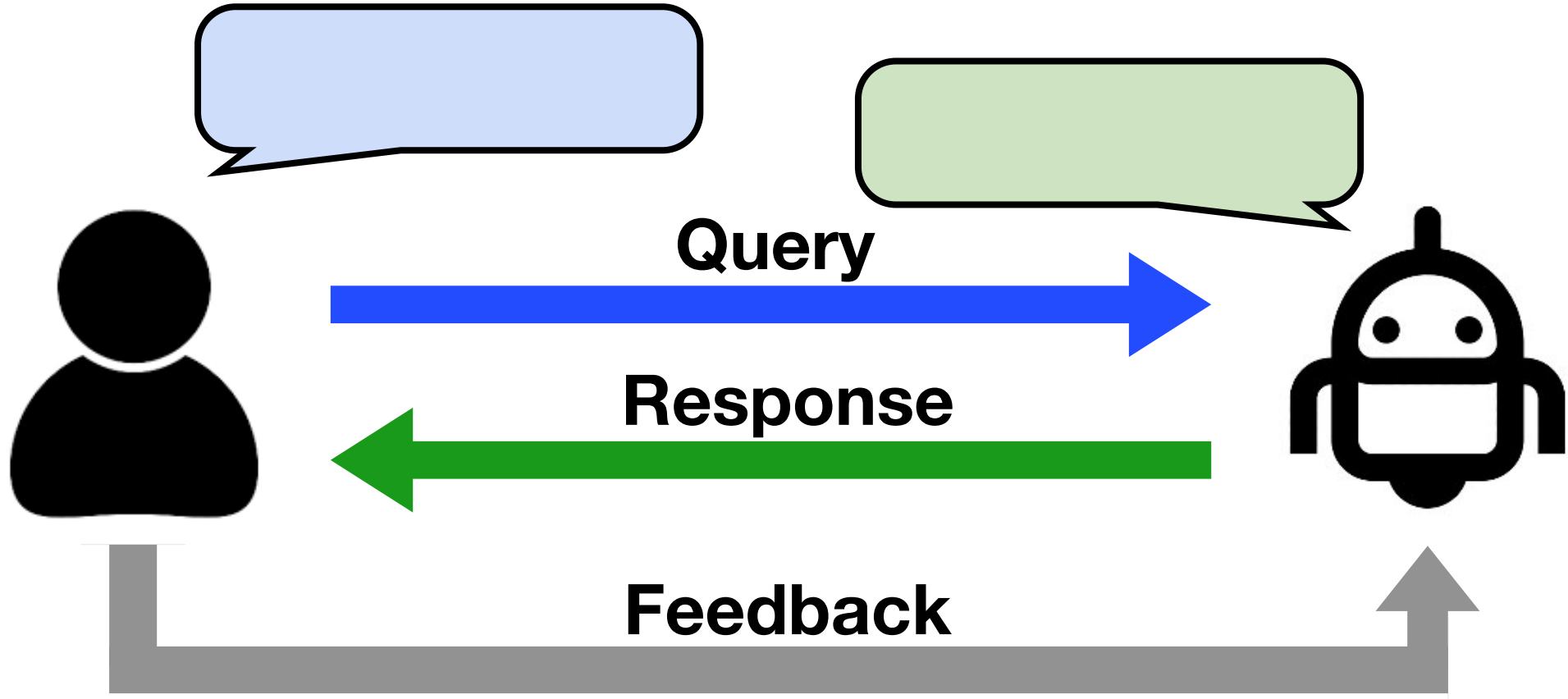
- We study learning from human feedback in the form of user edits
- **PRELUDE** framework formulates the interaction process and preference learning as a cost minimization problem
- **CIPHER** method learns a prompt policy to infer a descriptive user preference
 - computationally efficient, user-friendly, interpretable
 - empirically effective with a low expense
- More in the paper: human eval, email writing task, more baselines and analysis

Check Out Our Codebase!

- <https://github.com/gao-g/prelude> 
- Modularized codebase designed for easy customization
- Detailed instructions on how to:
 - Add your own task
 - Specify your own user
 - Implement your own agent

Outline: Learning from User Feedback

1. How to learn from naturally occurring human feedback?



Aligning LLM Agents by Learning Latent Preference from User Edits

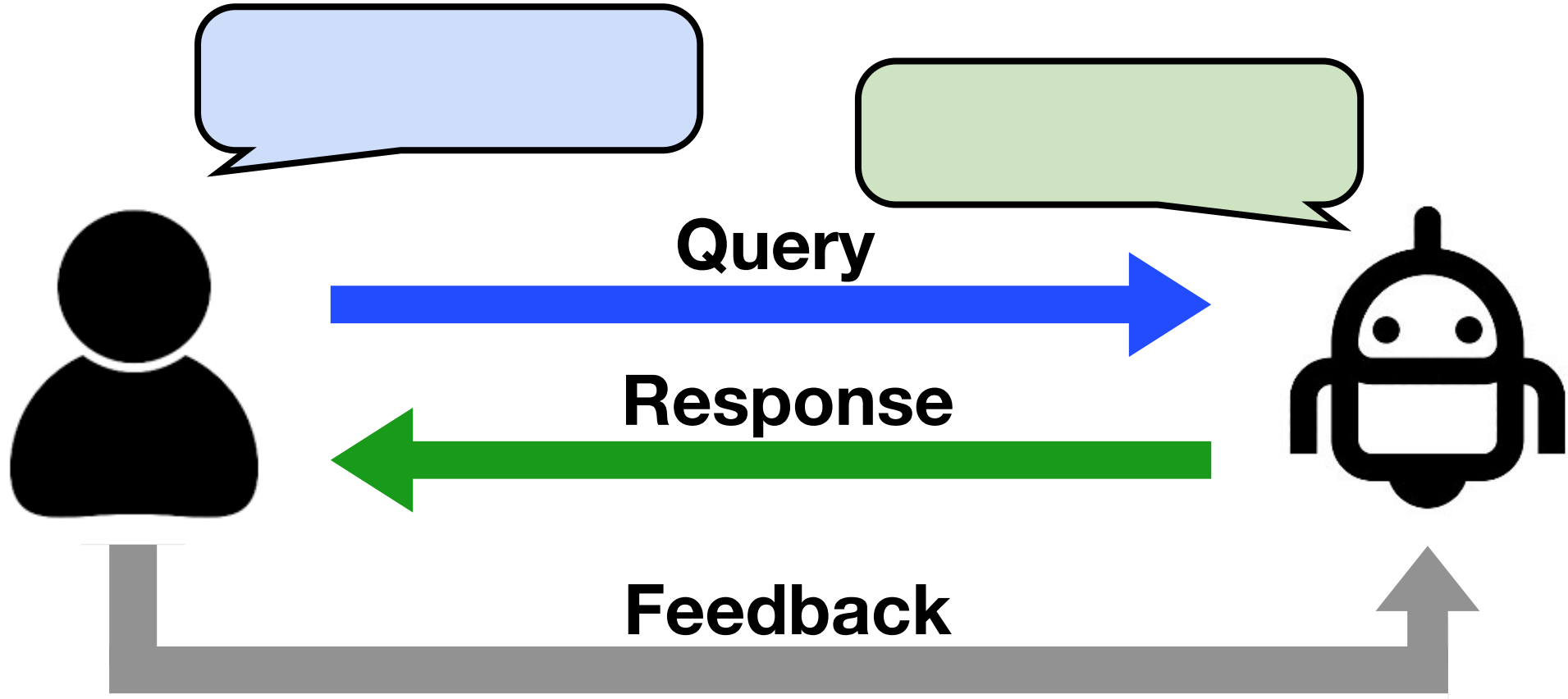
Ge Gao^{♣*} Alexey Taymanov^{◇*} Eduardo Salinas[◇] Paul Mineiro[◇] Dipendra Misra[◇]

Direct edits on the response

- Farming, as a part of agriculture, involves growing crops cultivation and animal rearing for food and raw materials.
- Originated ~~It began~~ thousands of years ago, likely in the Fertile Crescent, leading to the Neolithic Revolution
- Transition ~~as people transitioned~~ from nomadic hunting to settled farming. resulted in significant human population increase

Outline: Learning from User Feedback

1. How to learn from naturally occurring human feedback?



- Direct edits on the response

- Farming, as a part of agriculture, involves growing crops cultivation and animal rearing for food and raw materials.
- Originated ~~It began~~ thousands of years ago, likely in the Fertile Crescent, leading to the Neolithic Revolution
- Transition ~~as people transitioned~~ from nomadic hunting to settled farming. **resulted in significant human population increase**

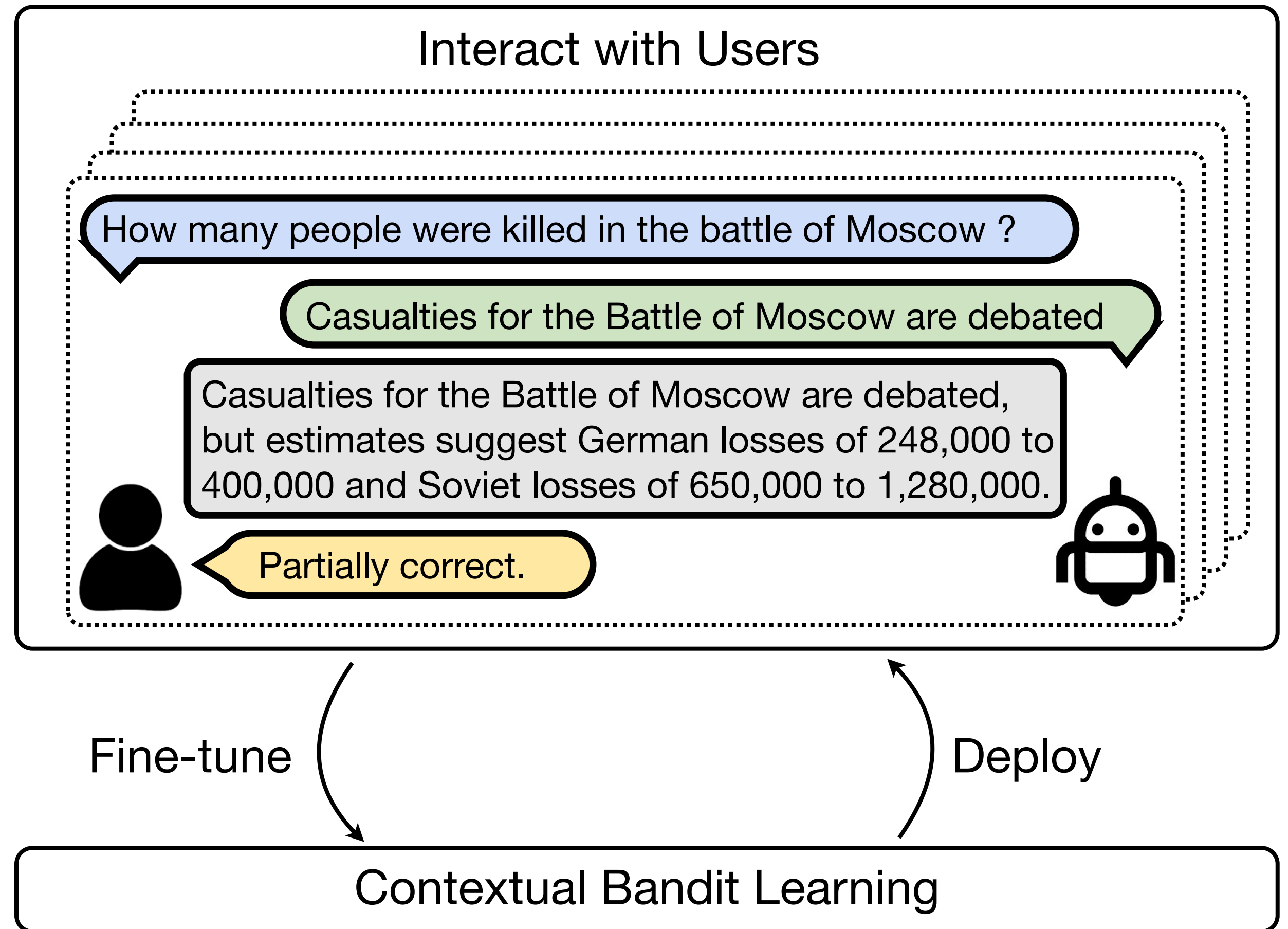
Simple categorical feedback

- Partially correct
- Correct
- Wrong

Simulating Bandit Learning from User Feedback for Extractive Question Answering
Ge Gao[◇], Eunsol Choi[♣] and Yoav Artzi[◇]
Continually Improving Extractive QA via Human Feedback
Ge Gao^{◇*}, Hung-Ting Chen^{♣*}, Yoav Artzi[◇], and Eunsol Choi[♣]

Interactive QA: Setup

- Explicit user feedback
- In practice, such categorical feedback can also be derived from user behaviors
- Two work on this topic:
 - with simulated feedback
 - with human user feedback



Human User Feedback Study: Highlights

- We cast the problem as contextual bandit learning, and introduce an effective fine-tuning method for categorical feedback
- Our experiments show empirical improvement of extractive QA agents over multiple rounds of human user interaction
- Our ablation studies demonstrate the potential of domain adaptation

Continually Improving Extractive QA via Human Feedback

Ge Gao^{◇*}, Hung-Ting Chen^{♣*}, Yoav Artzi[◇], and Eunsol Choi[♣]

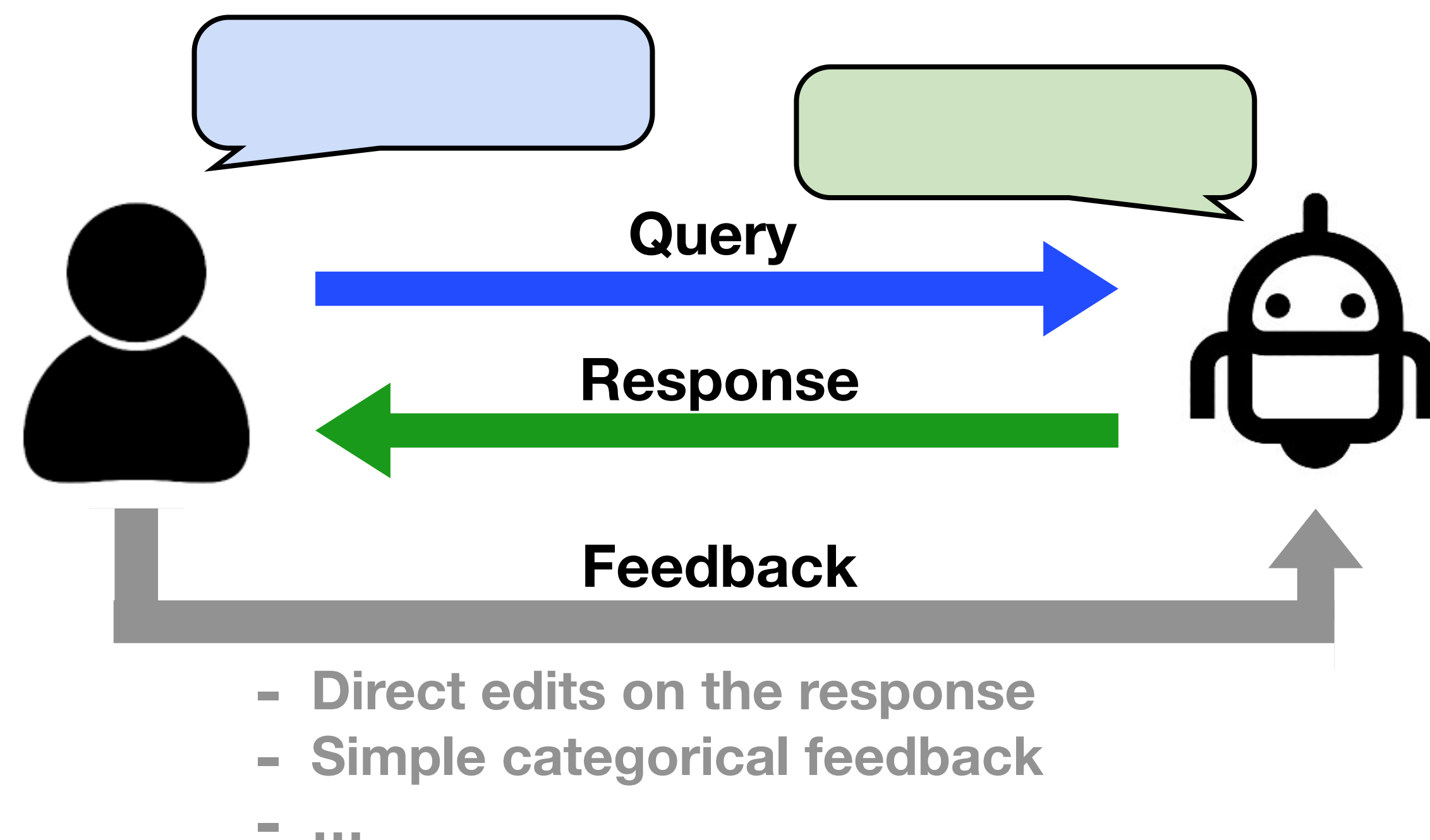
[◇]Department of Computer Science and Cornell Tech, Cornell University

[♣]Department of Computer Science, The University of Texas at Austin

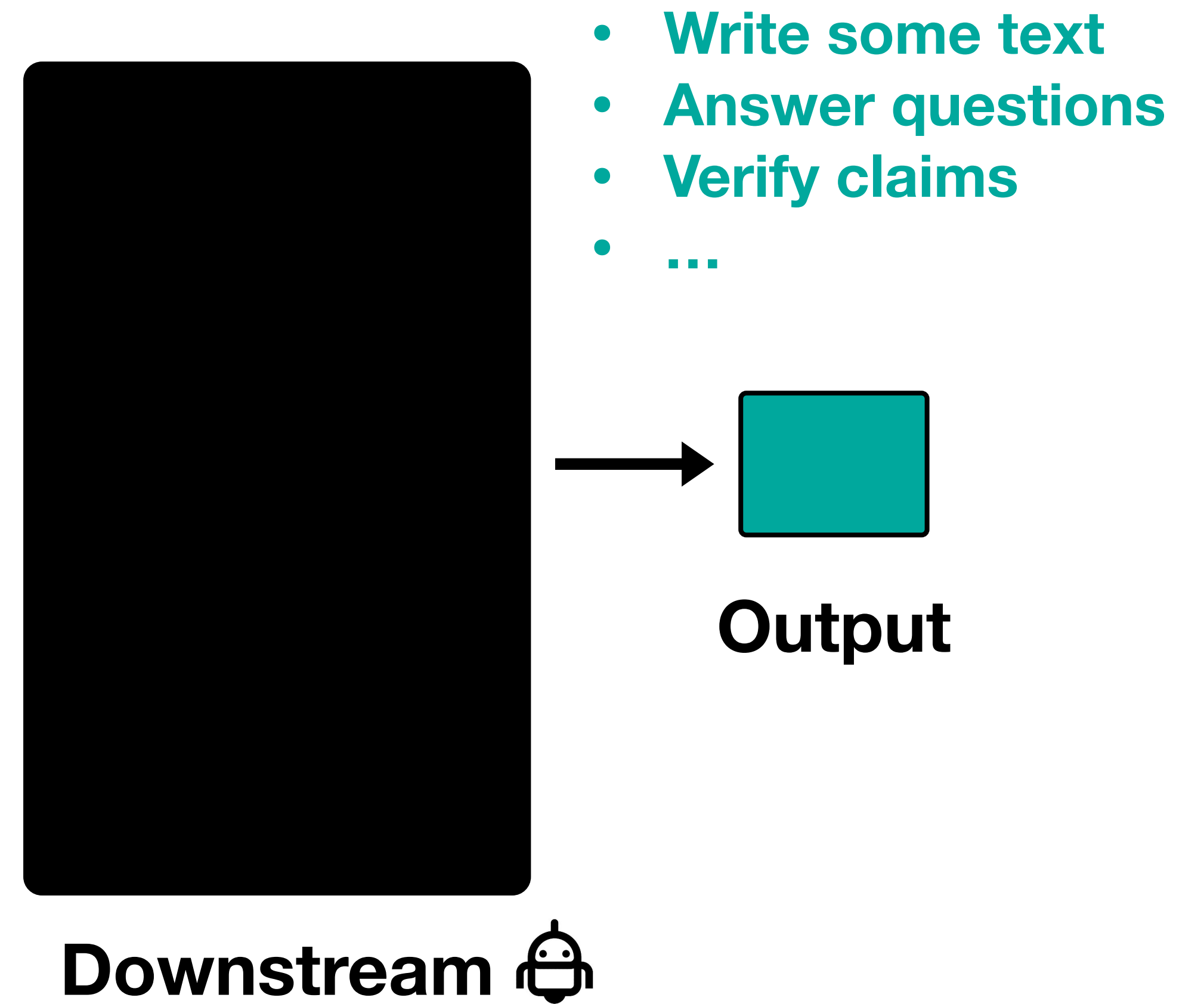
Outline: Learning from User Feedback

1. How to learn from naturally occurring human feedback?

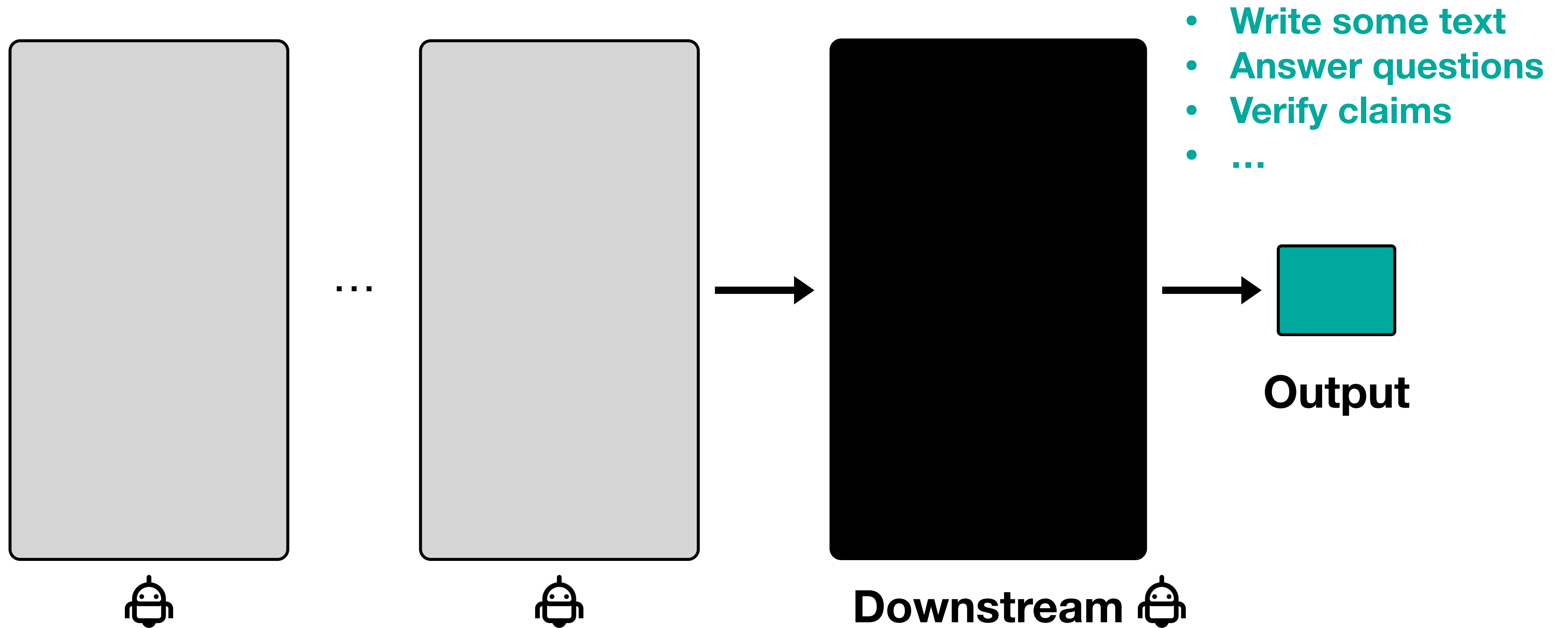
2. Going beyond individual model improvement, how to improve language processing pipelines?



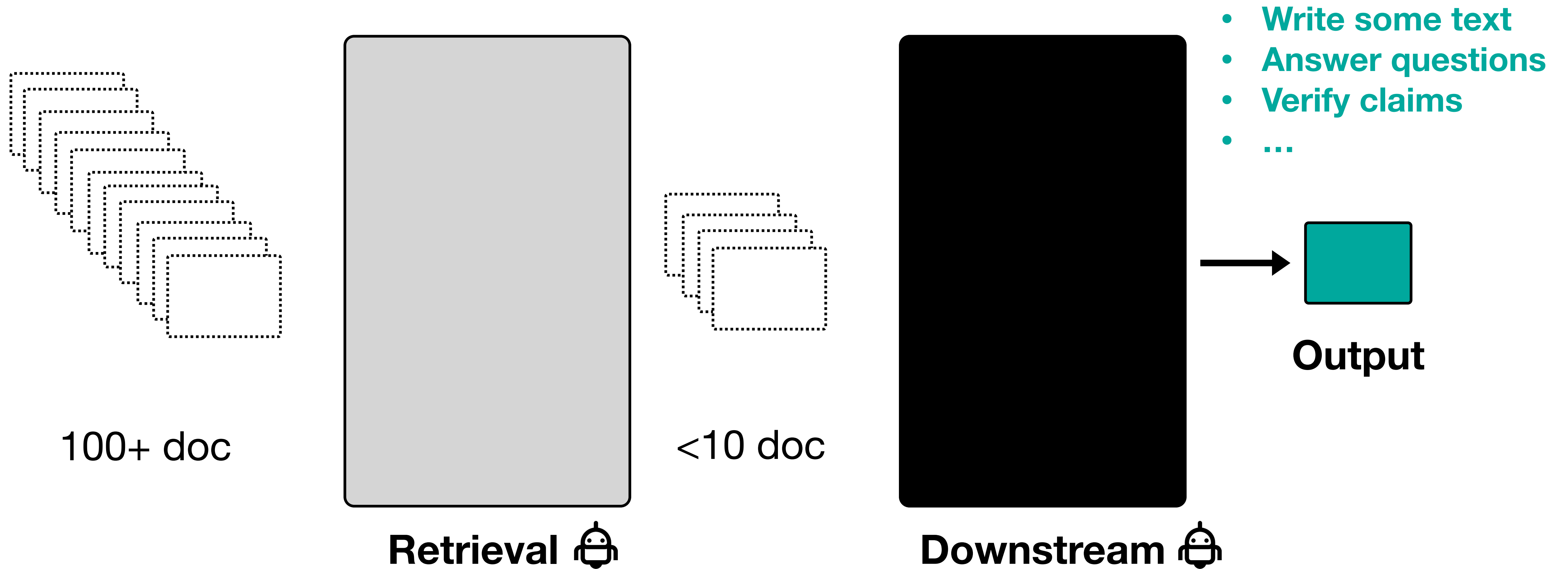
Motivation: Deployment in Practice



Motivation: Deployment in Practice



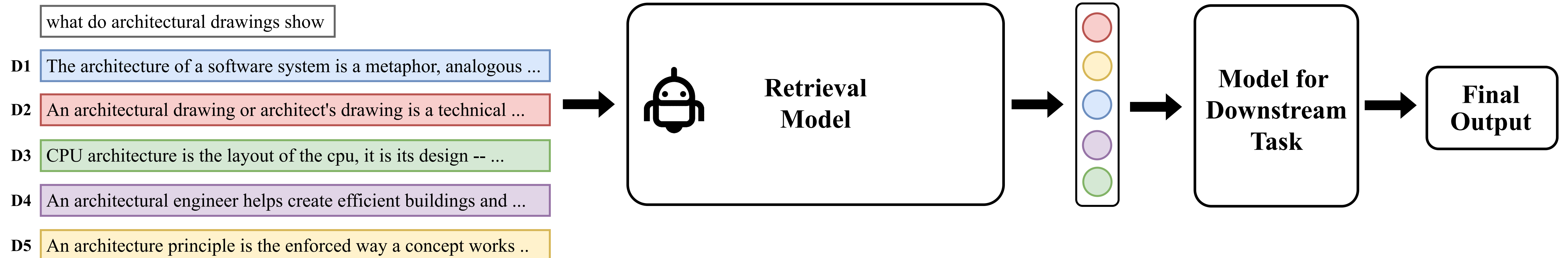
Motivation: Deployment in Practice



Retrieval Basics

- Task definition: rank documents based on their relevance to a query
- Application: ranked documents are input to some downstream models
- Separate from training retrieval models

Query and Documents



Retrieval Basics

- Conventional training objectives: contrastive loss
- Requires **ground truth annotation for relevant documents** and **estimation for truly irrelevant documents** (i.e., hard negatives)

Query and Documents

what do architectural drawings show

- | | | |
|----|--|------------|
| D1 | The architecture of a software system is a metaphor, analogous ... | Irrelevant |
| D2 | An architectural drawing or architect's drawing is a technical ... | Relevant |
| D3 | CPU architecture is the layout of the cpu, it is its design -- ... | Irrelevant |
| D4 | An architectural engineer helps create efficient buildings and ... | Irrelevant |
| D5 | An architecture principle is the enforced way a concept works .. | Irrelevant |

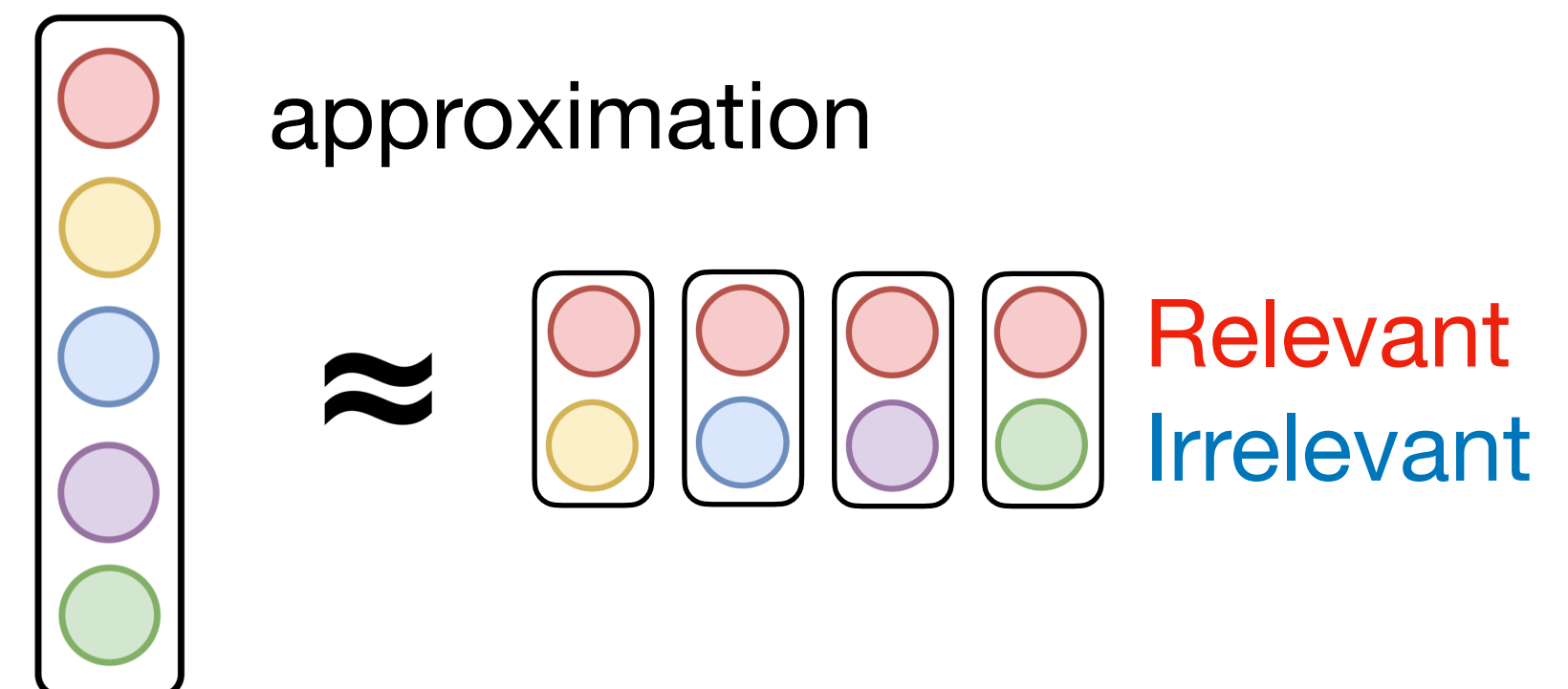
Retrieval Basics

- Conventional training objectives: contrastive loss
 - Requires **ground truth annotation for relevant documents** and **estimation for truly irrelevant documents** (i.e., hard negatives)
- Pairwise approximation of the listwise ranking objective

Query and Documents

what do architectural drawings show

| | | |
|----|--|------------|
| D1 | The architecture of a software system is a metaphor, analogous ... | Irrelevant |
| D2 | An architectural drawing or architect's drawing is a technical ... | Relevant |
| D3 | CPU architecture is the layout of the cpu, it is its design -- ... | Irrelevant |
| D4 | An architectural engineer helps create efficient buildings and ... | Irrelevant |
| D5 | An architecture principle is the enforced way a concept works .. | Irrelevant |



Research Question

- How to improve LLM-based retrieval models in a pipeline setup?
 - Need to adapt to the downstream applications
 - No document-level annotation available
 - Output high-quality ranking over >2 docs for practical usage

Our Contribution

- **Neural PG-RANK:** train LLM-based retrievers in a pipeline setup based on downstream feedback
 - Directly optimize the listwise objective — use Plackett-Luce ranking policy

Query and Documents

what do architectural drawings show

D1 The architecture of a software system is a metaphor, analogous ...

D2 An architectural drawing or architect's drawing is a technical ...

D3 CPU architecture is the layout of the cpu, it is its design -- ...

D4 An architectural engineer helps create efficient buildings and ...

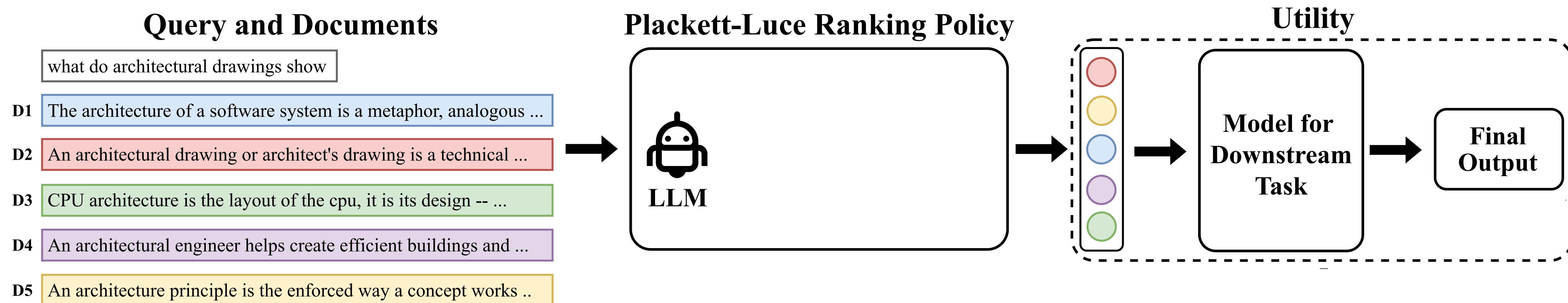
D5 An architecture principle is the enforced way a concept works ..

Plackett-Luce Ranking Policy



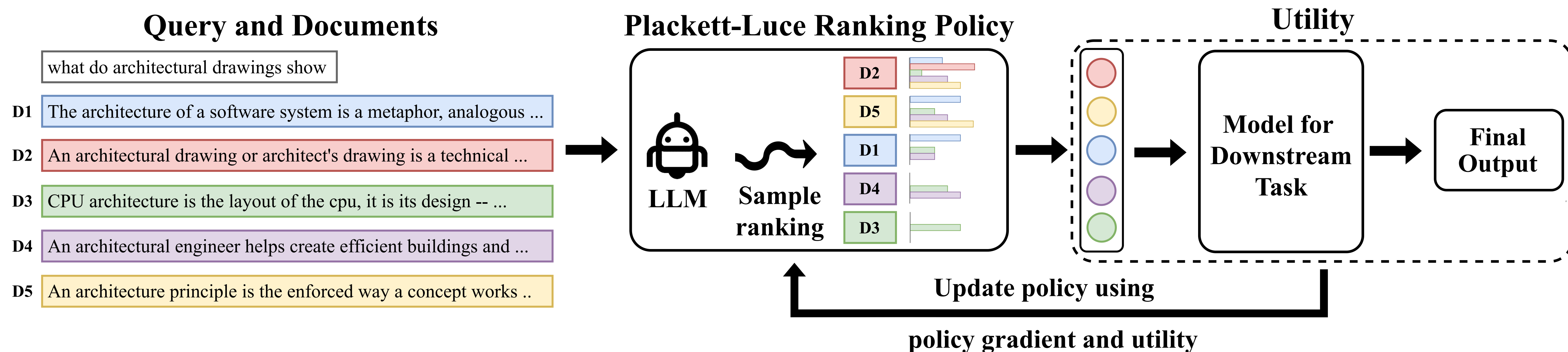
Our Contribution

- **Neural PG-RANK:** train LLM-based retrievers in a pipeline setup based on downstream feedback
 - Directly optimize the listwise objective — use Plackett-Luce ranking policy
 - Do not require document-level relevance annotation — use downstream utility



Our Contribution

- **Neural PG-RANK:** train LLM-based retrievers in a pipeline setup based on downstream feedback
 - Directly optimize the listwise objective — use Plackett-Luce ranking policy
 - Do not require document-level relevance annotation — use downstream utility
 - Adapt to downstream tasks by unifying the training objective with the downstream application



Setting

- Define the utility of a ranking policy for a given query

$$U(\pi|q) = \mathbb{E}_{r \sim \pi(\cdot|q)} [\Delta(r|q)]$$

Utility function

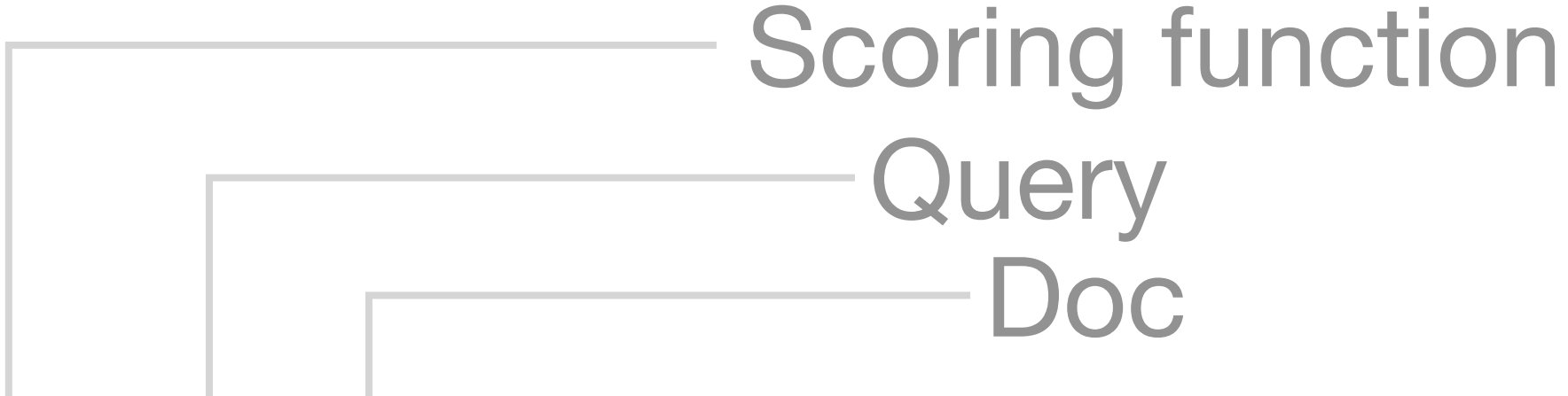
Query
Ranking

- Learning objective is to learn a ranking policy that optimizes the expected utility over the query distribution

$$\pi^* = \operatorname{argmax}_{\pi \in \Pi} \mathbb{E}_{q \sim \mathcal{Q}} [U(\pi|q)]$$

Method

- Define a Plackett-Luce ranking policy
 - expressed as a product of softmax distributions
 - based on query-document relevance scores


$$\pi_{\theta}(r|q) = \prod_{i=1}^n \frac{\exp s_{\theta}(q, d_{r(i)})}{\sum_{j \in \{r(i), \dots, r(n)\}} \exp s_{\theta}(q, d_j)}$$

Method

- We use REINFORCE

$$\begin{aligned}\nabla_{\theta} U(\pi_{\theta} | q) &= \nabla_{\theta} \mathbb{E}_{r \sim \pi_{\theta}(\cdot | q)} [\Delta(r | q)] \\ &= \mathbb{E}_{r \sim \pi_{\theta}(\cdot | q)} [\nabla_{\theta} \log \pi_{\theta}(r | q) \Delta(r | q)]\end{aligned}$$

Method

- We use REINFORCE

+ Monte Carlo sampling with N samples

+ Variance reduction with leave-one-out baseline

$$\hat{\nabla}_{\theta} U(\pi_{\theta}|q) = \frac{1}{N} \sum_i \left[\nabla_{\theta} \log \pi_{\theta}(r_i|q) \left(\Delta(r_i|q) - \frac{1}{N-1} \sum_{j \neq i} \Delta(r_j|q) \right) \right]$$

Utility

- Utility function scores a ranking based on how useful it is in the downstream
- In our pilot study, we use nDCG@10 as an approximation of the downstream utility
 - nDCG@10 is a measure of ranking quality
 - We assume higher nDCG@10 relates to better downstream performance

Policy-Gradient Training of Language Models for Ranking

Ge Gao Jonathan D. Chang Claire Cardie Kianté Brantley Thorsten Joachims
Department of Computer Science, Cornell University

Utility

- We use REINFORCE

+ Monte Carlo sampling with N samples

+ Variance reduction with leave-one-out baseline

+ nDCG@10 as utility function

$$\hat{\nabla}_{\theta} U(\pi_{\theta}|q) = \frac{1}{N} \sum_i \left[\nabla_{\theta} \log \pi_{\theta}(r_i|q) \left(\Delta(r_i|q) - \frac{1}{N-1} \sum_{j \neq i} \Delta(r_j|q) \right) \right]$$

$$= \frac{1}{N} \sum_i \left[\sum_k \nabla_{\theta} \log \pi_{\theta}(r_{i,k}|q, r_{i,1:k-1}) \right]$$

Utility score for the partial ranking til k

$$\left(\text{nDCG}(r_{i,k}:|q, r_{i,1:k-1}) - \frac{1}{N-1} \sum_{j \neq i} \text{nDCG}(r_{j,k}:|q, r_{i,1:k-1}) \right)$$

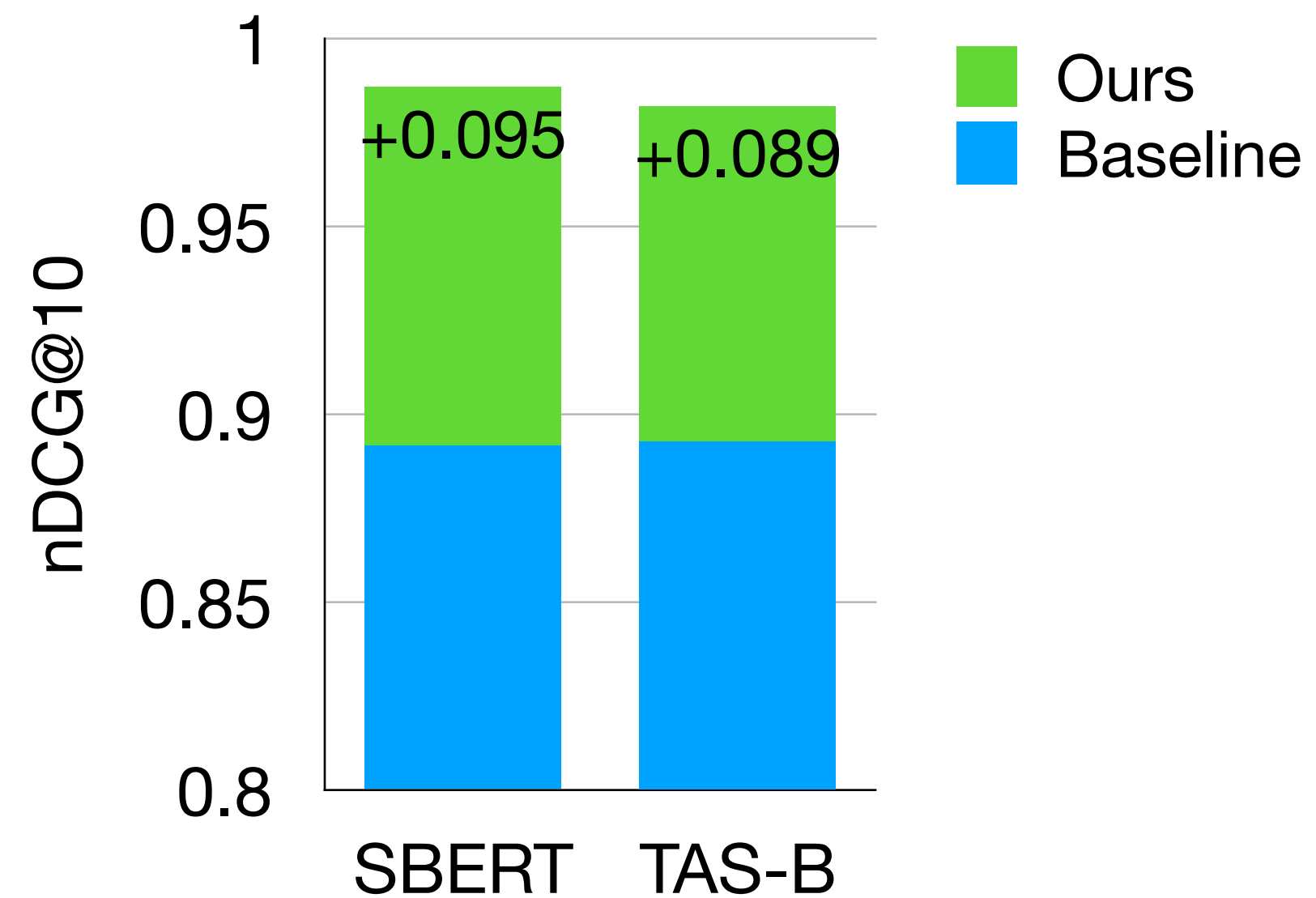
Experimental Setup

- Data: MS MARCO [Campos et al., 2017]
- Evaluation metric: nDCG@k for k = 10, 3, 1
- Our ranking policy: SBERT or TAS-B as warmstart + Neural PG-RANK fine-tuning
- Comparison systems: supervised learning SOTA bi-encoder models with augmented contrastive loss objective

| Method | Source of Negative Docs | | | Additional Supervision | Loss |
|----------------------------------|-------------------------|------|-------------|------------------------|--------------------------|
| | In-Batch | BM25 | Dense Model | | |
| SBERT (Reimers & Gurevych, 2019) | | ✓ | ✓✓✓ | ✓ | MarginMSE + NLL |
| TAS-B (Hofstätter et al., 2021) | ✓ | ✓ | | ✓✓ | MarginMSE + Distillation |
| Neural PG-RANK (Ours) | | | ✓ | | Utility Maximization |

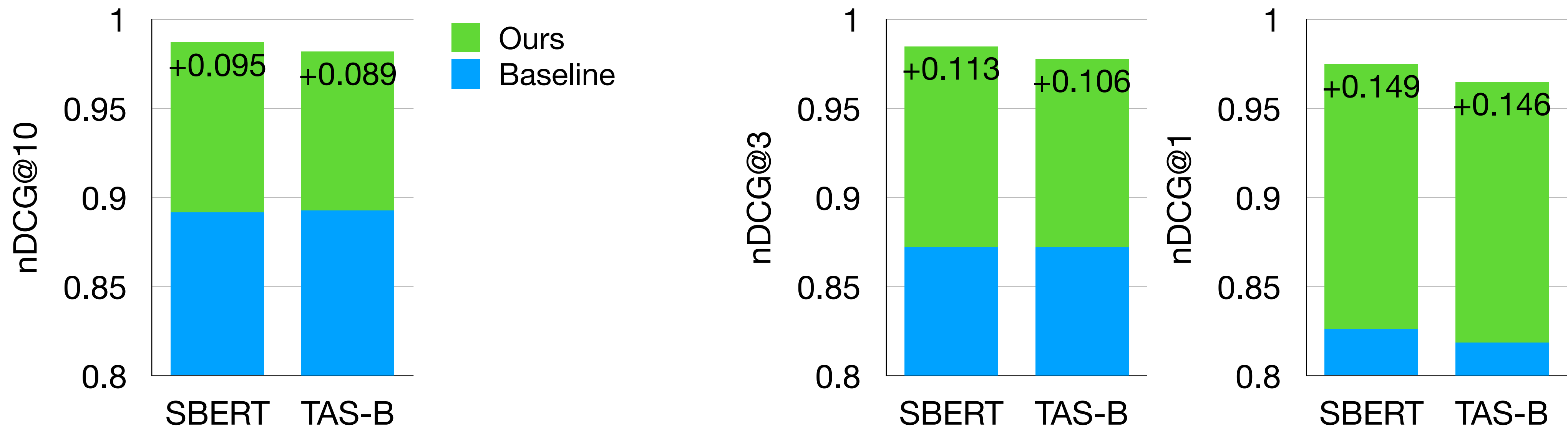
Experimental Result

- Setup: search over a candidate set of 1k documents per query
- Performance gains with both warmstart models



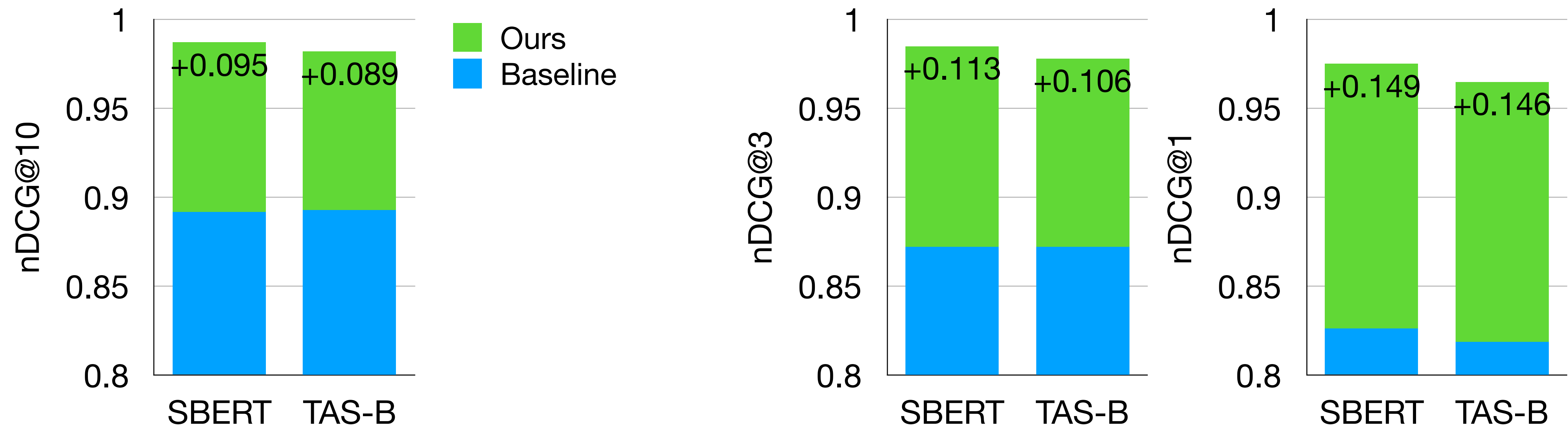
Experimental Result

- Setup: search over a candidate set of 1k documents per query
- Performance gains with both warmstart models
- More gains in terms of nDCG@k with smaller k



Experimental Result

- Setup: search over a candidate set of 1k documents per query
 - Performance gains with both warmstart models
 - More gains in terms of nDCG@k with smaller k
- Neural PG-RANK effectively ranks relevant docs high up**



Summary

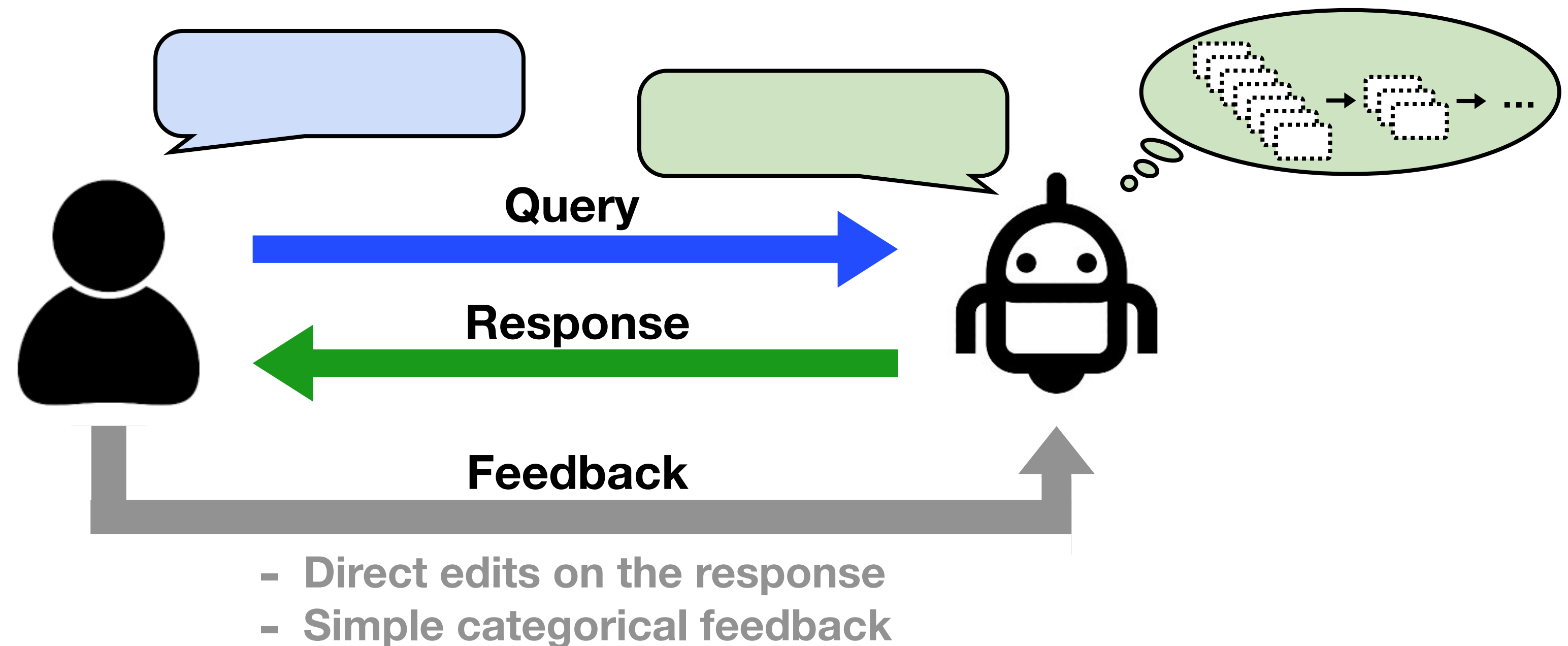
- We introduce Neural PG-RANK to train LLM-based retriever and to improve retrieval-based pipeline based on downstream feedback
 - Directly optimize the listwise ranking objective using Plackett-Luce ranking policy
 - End-to-end training of the ranker as part of larger pipelines via policy gradient
 - Can adapt to downstream applications by optimizing any utility function for the downstream performance (no document-level annotation required)
- Pilot study: nDCG@10 as an approximation of the downstream utility
- Ongoing work: performance on the downstream task as the utility function

Overview: Learning from User Feedback

1. Learn from naturally occurring human feedback
2. Improve retrieval-based pipelines from downstream feedback

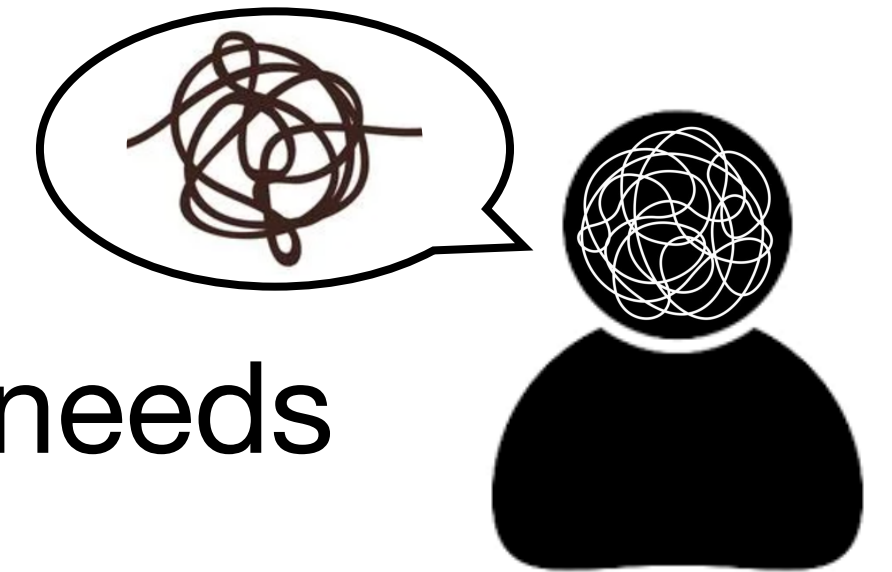
Ongoing work:

- Neural PG-RANK with realistic utility
- Fine-tuning method under PRELUDE



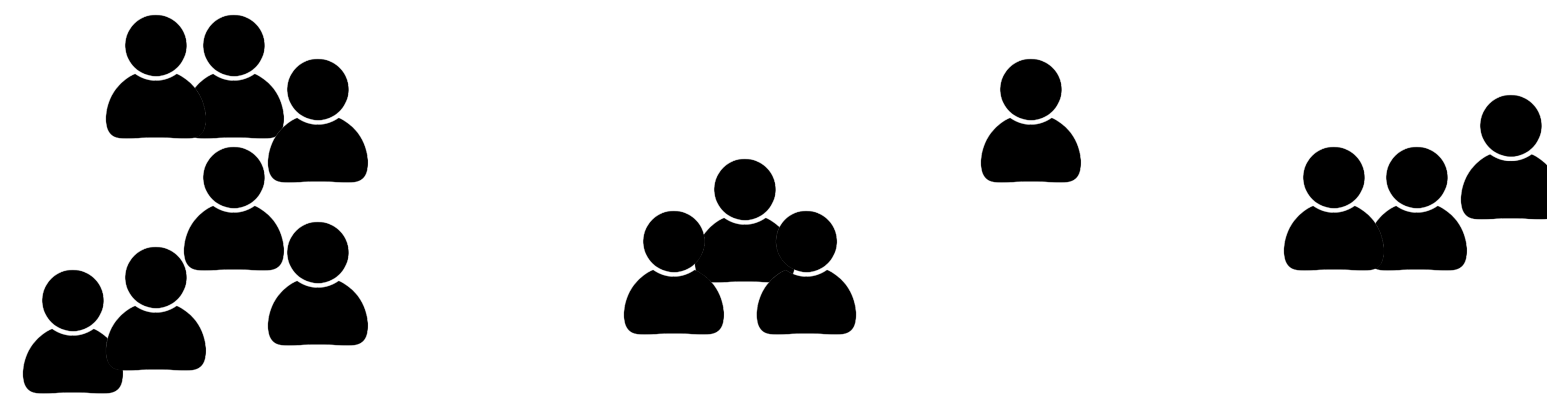
Looking Forward

- What assumption to make or break in interactive learning?



- No expectation on users to articulate their full range of needs

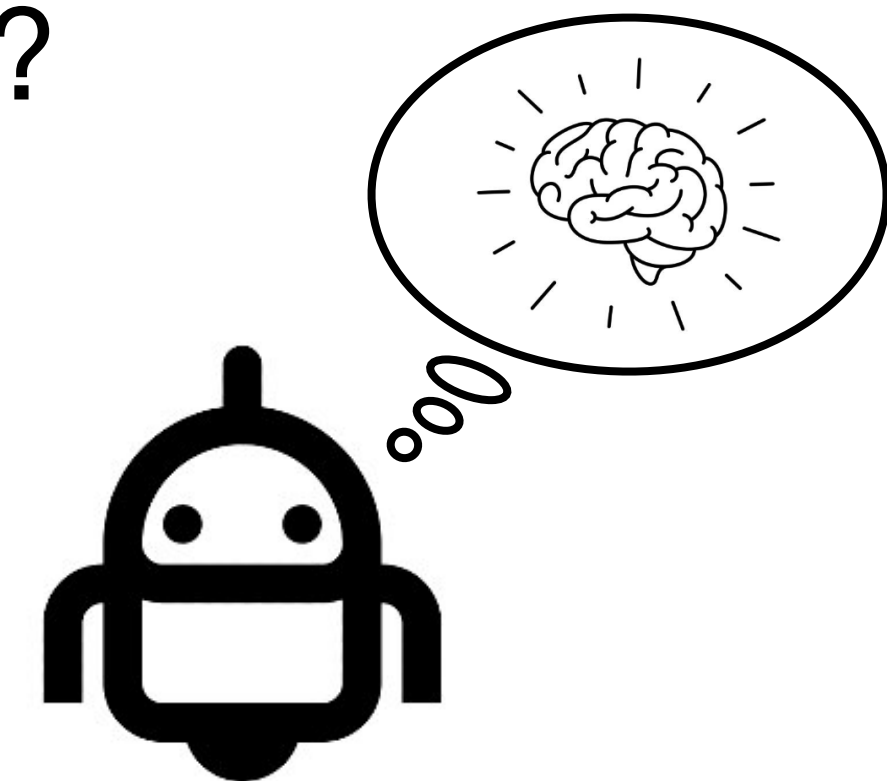
- Modeling user groups



- How to build AI that is knowledgeable and intelligent?

- Beyond learning patterns and doing tasks

- Become helpful in underspecified use cases



Questions?