Google Cloud

# LLM Alignment as Retriever Optimization: An Information Retrieval Perspective

Bowen Jin,

Mentors: Jinsung Yoon, Sercan Arik

# Goal

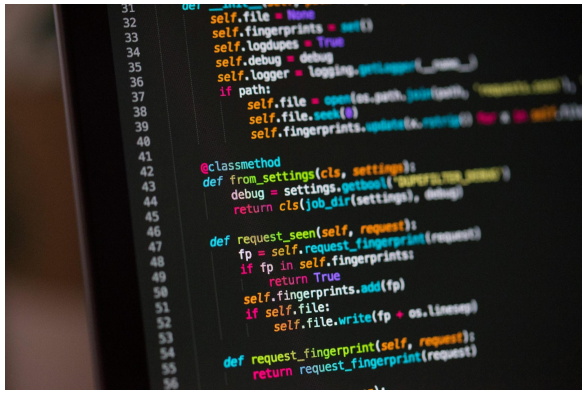
- **Provide some insights on understanding LLM alignment from an IR perspective.**

- **Propose a new LLM alignment method with an IR philosophy.**

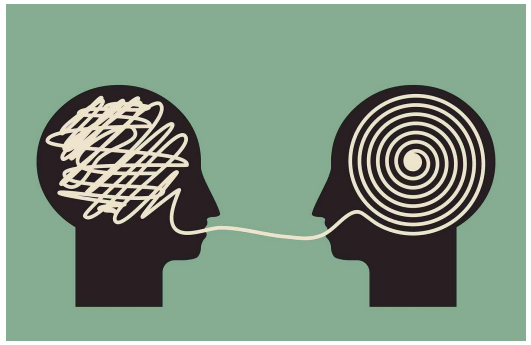
# Motivation

- **Large language models are extraordinary.**
  - **LLaMA, GPT4, Gemini, …**

**chat**



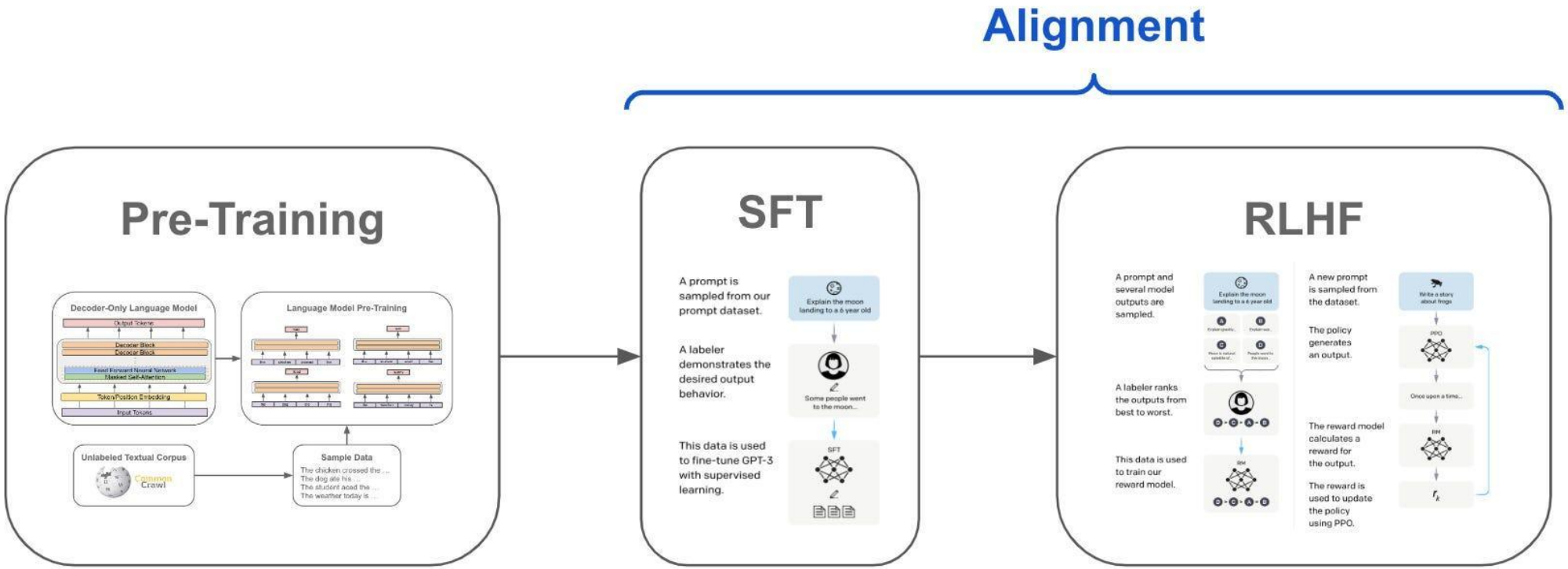
**coding**



**reasoning**

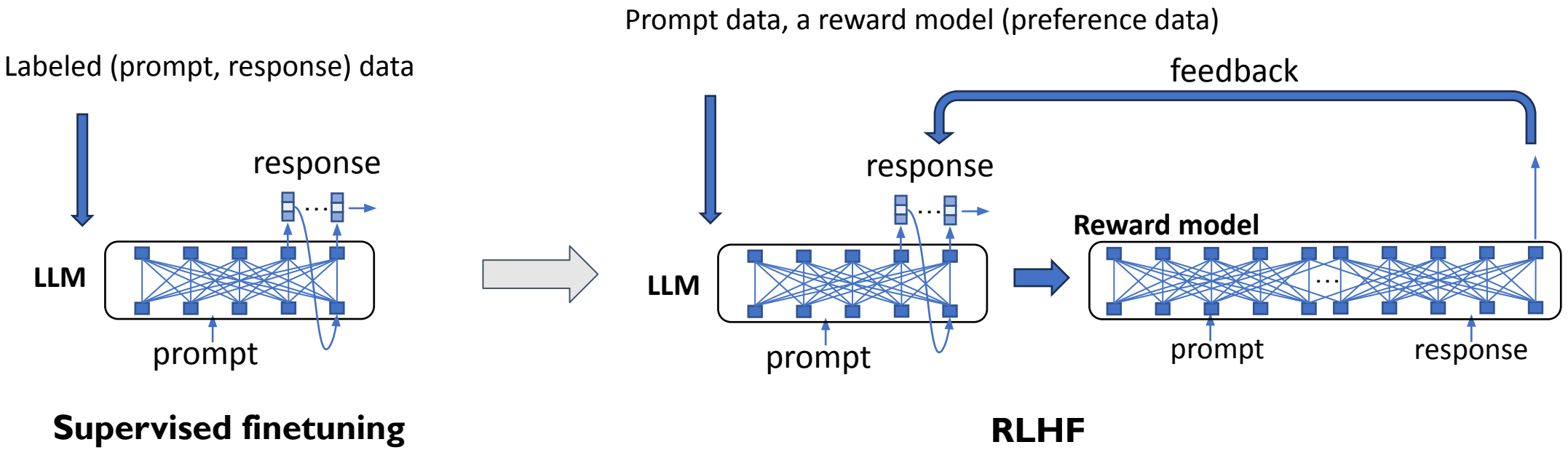
# Motivation

- **Large language model <u>training</u> consists three steps.**



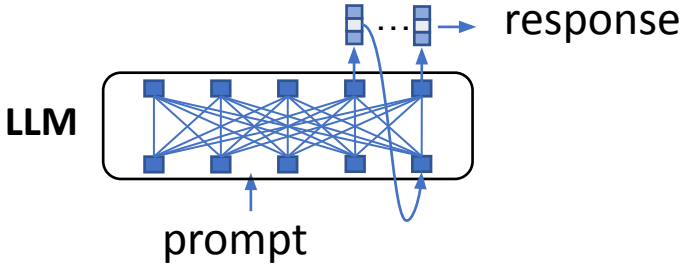
# Motivation

- **Large language model <u>training</u>: SFT and RLHF.**

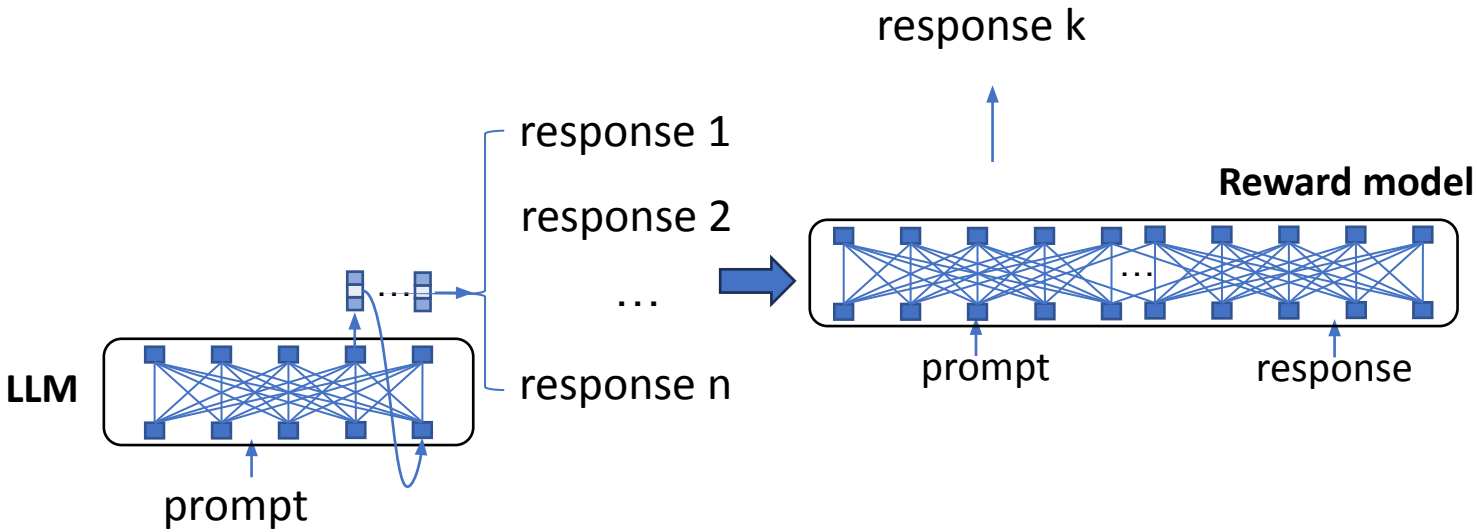


# Motivation

- **Large language model <u>inference</u> paradigm.**



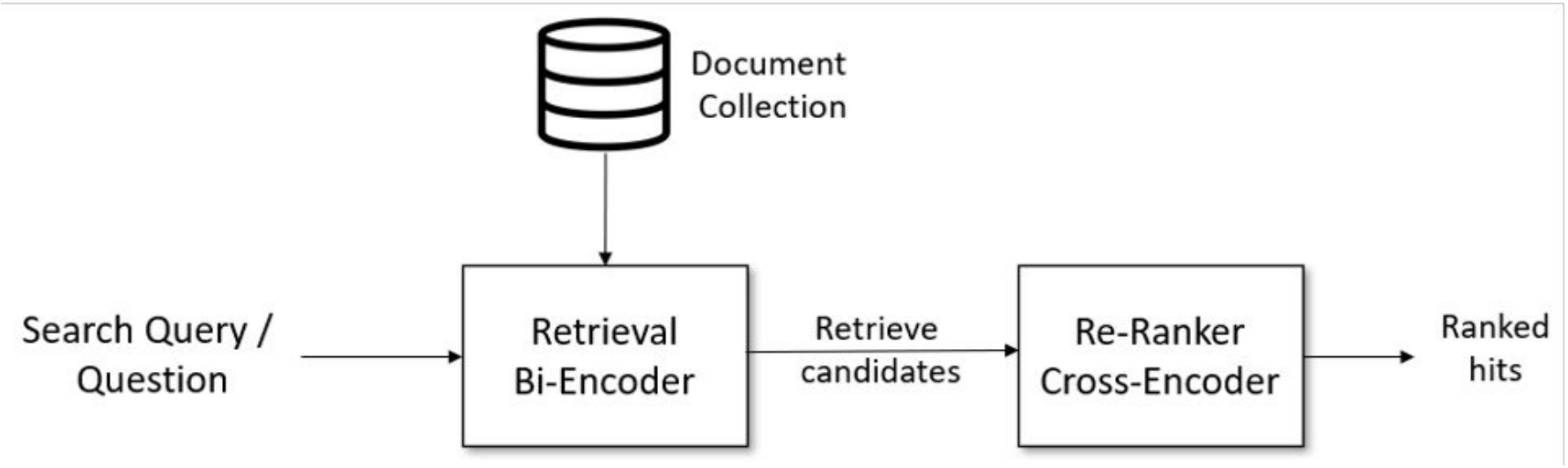

**Greedy decoding**




**Best-of-N decoding**

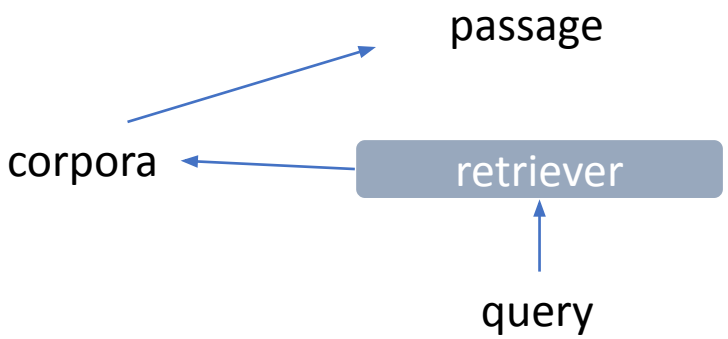
# Motivation

- **Let's think about information retrieval (IR).**
  - **In an IR system, we usually have retrievers and rerankers.**
  - **Retrievers can work on large corpora efficiently, while not accurate enough.**
  - **Rerankers can more accurately measure the semantic similarity while not efficient.**

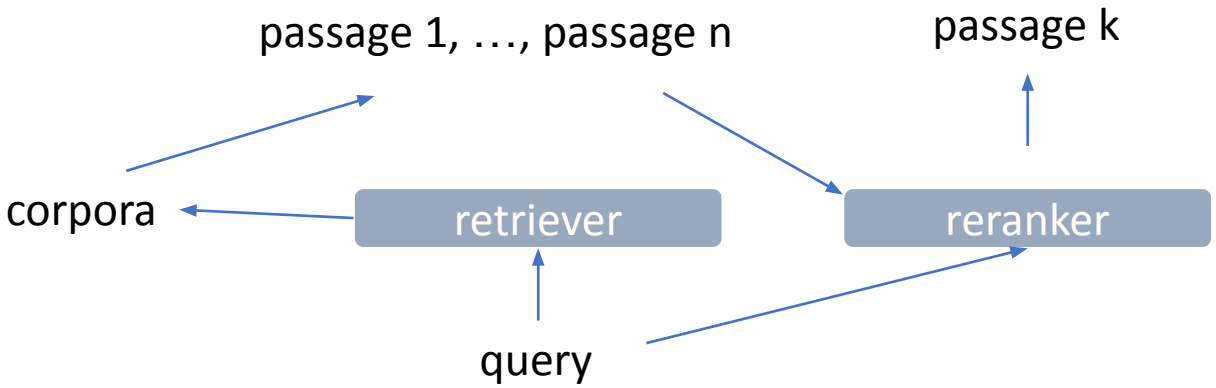


# Motivation

- **Information retrieval (IR) <u>inference</u> pipeline**



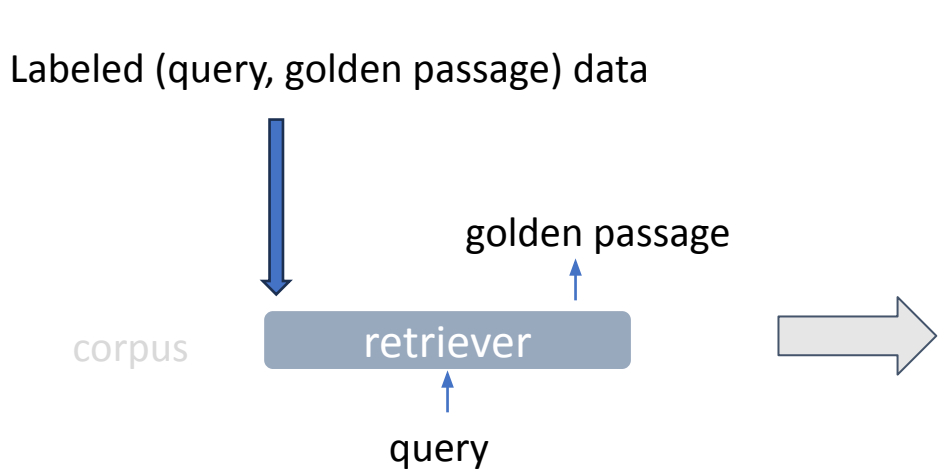

**Pure retriever**



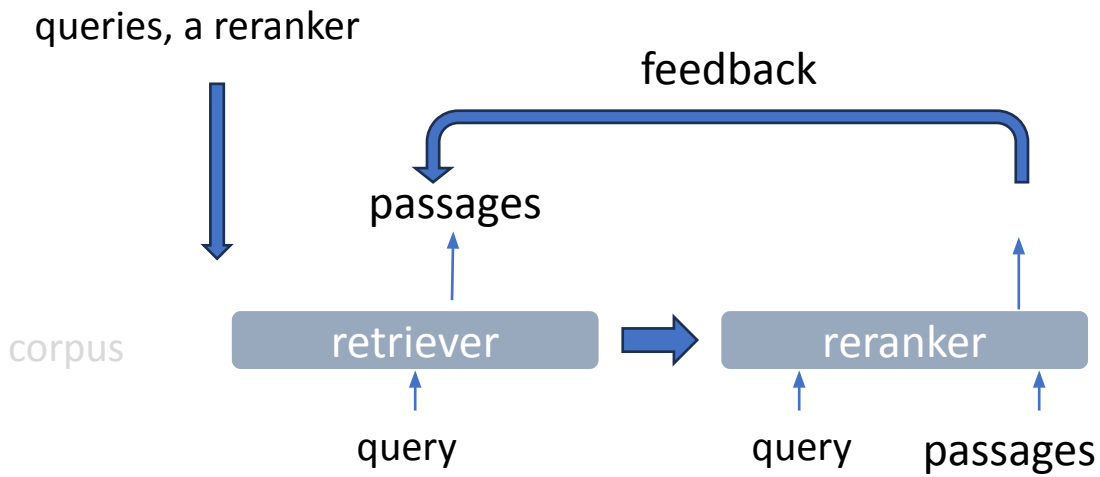

**Retriever-reranker**

# Motivation

- **Information retrieval (IR) <u>training</u>**




**Direct retriever optimization**



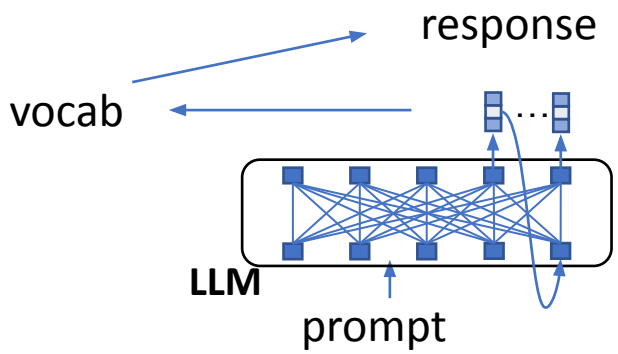

**Reranker-retriever distillation**

[1] Karpukhin, Vladimir, et al. "Dense Passage Retrieval for Open-Domain Question Answering." EMNLP. 2020.
[2] Qu, Yingqi, et al. "RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering." NAACL. 2021.

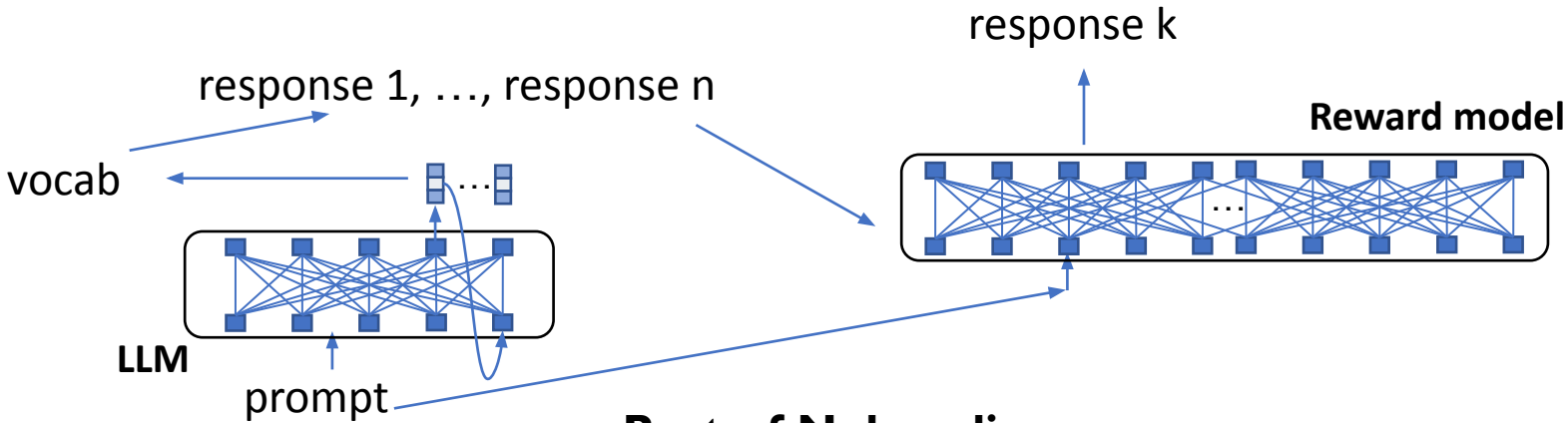
# Understand the connection between generative language modeling and IR

- Inference stage connection
- Training stage connection
- Model architecture connection

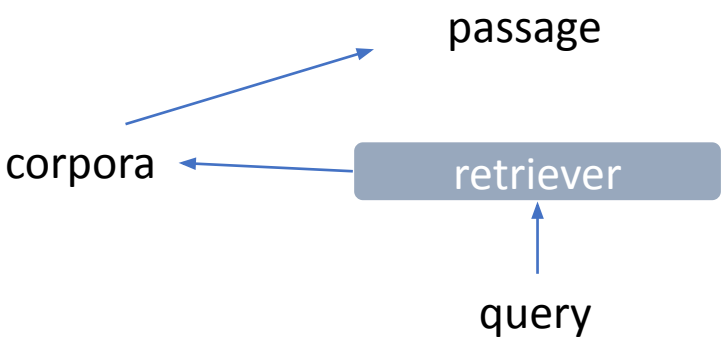
# Inference stage connection



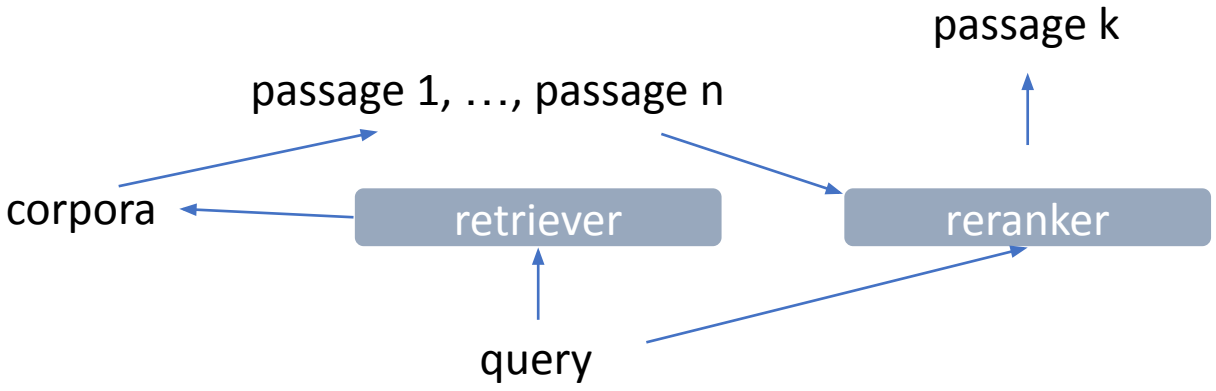

**LLM decoding**




**Best-of-N decoding**



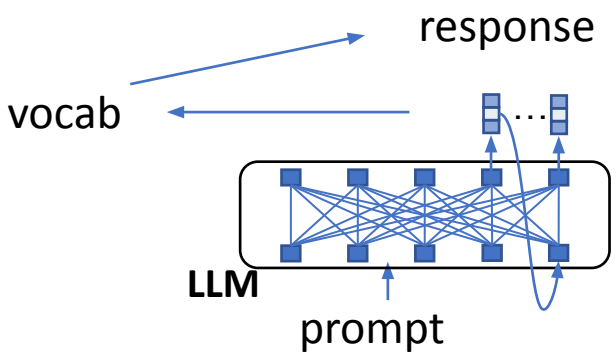

**Pure retriever**



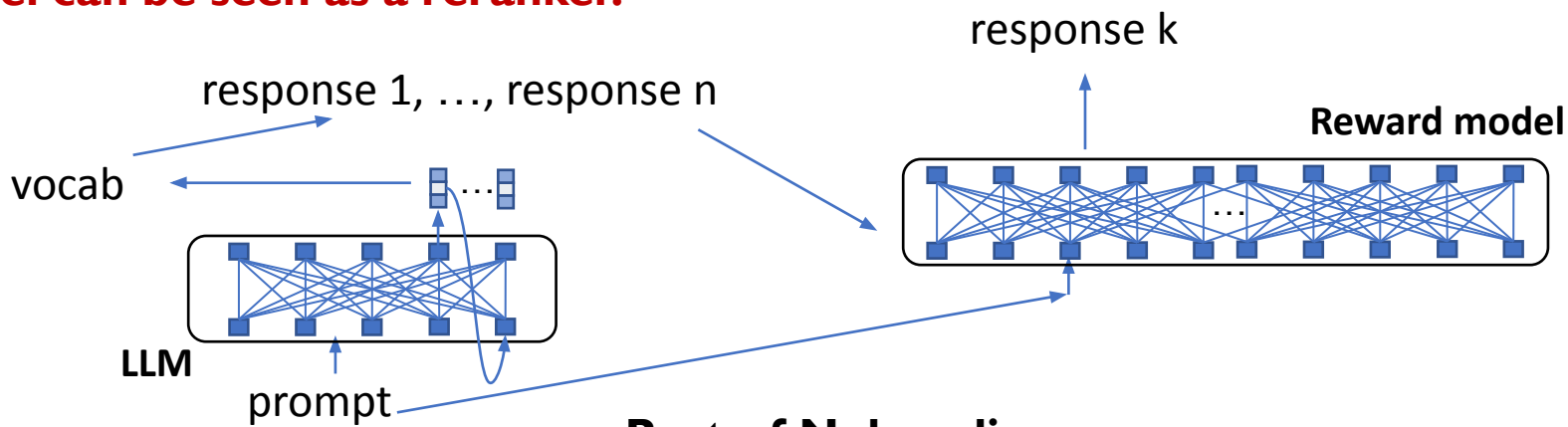

**Retriever-reranker**

# Inference stage connection

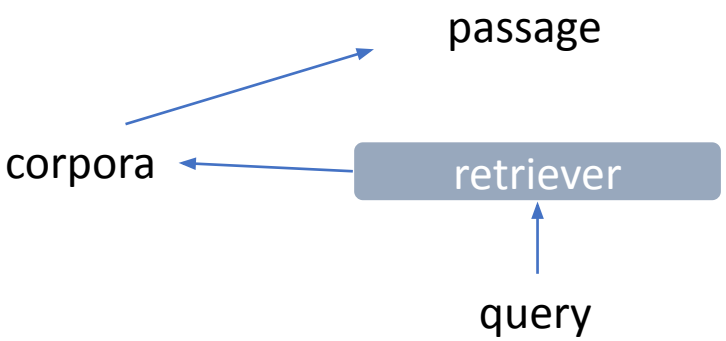
**LLM can be seen as a retriever while reward model can be seen as a reranker.**



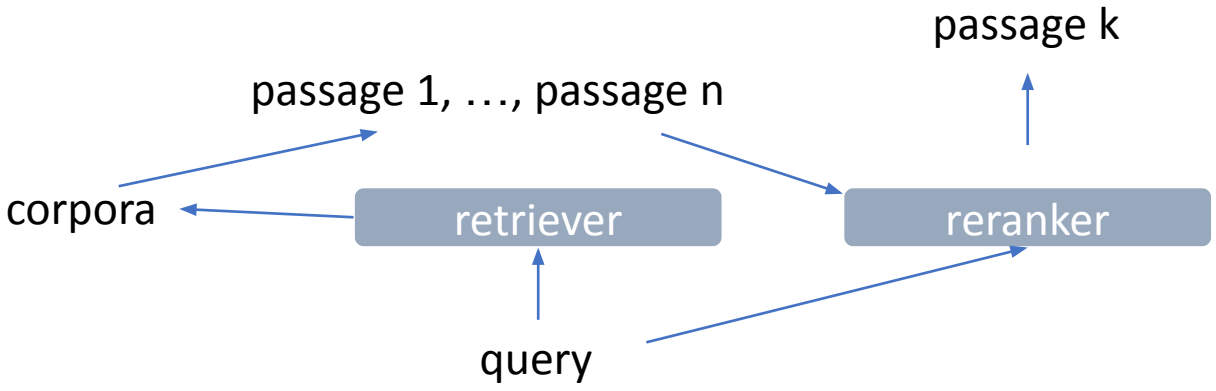

**LLM decoding**




**Best-of-N decoding**

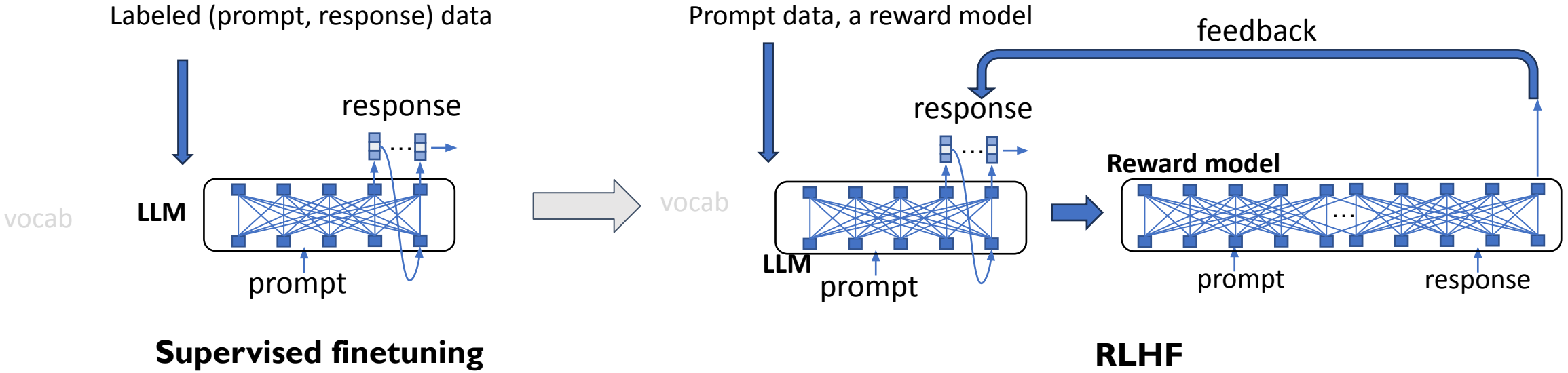


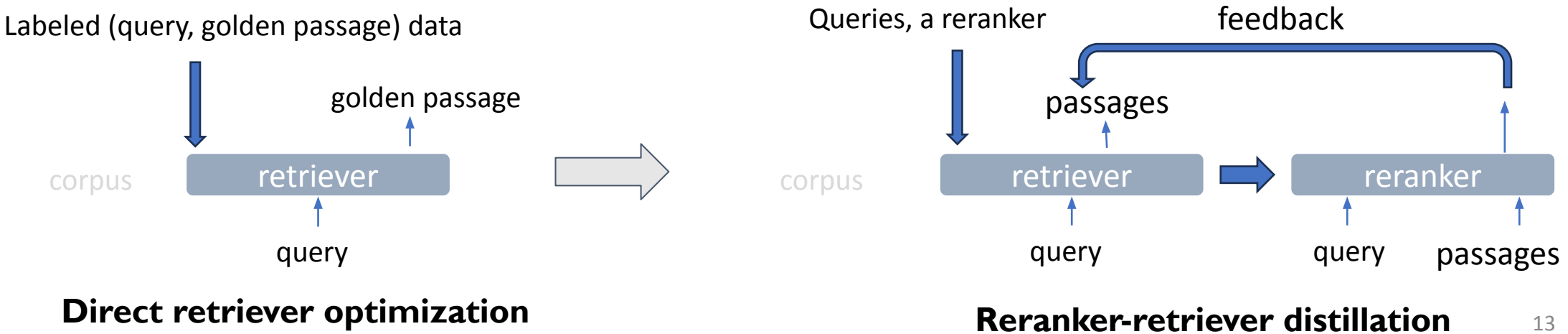

**Pure retriever**



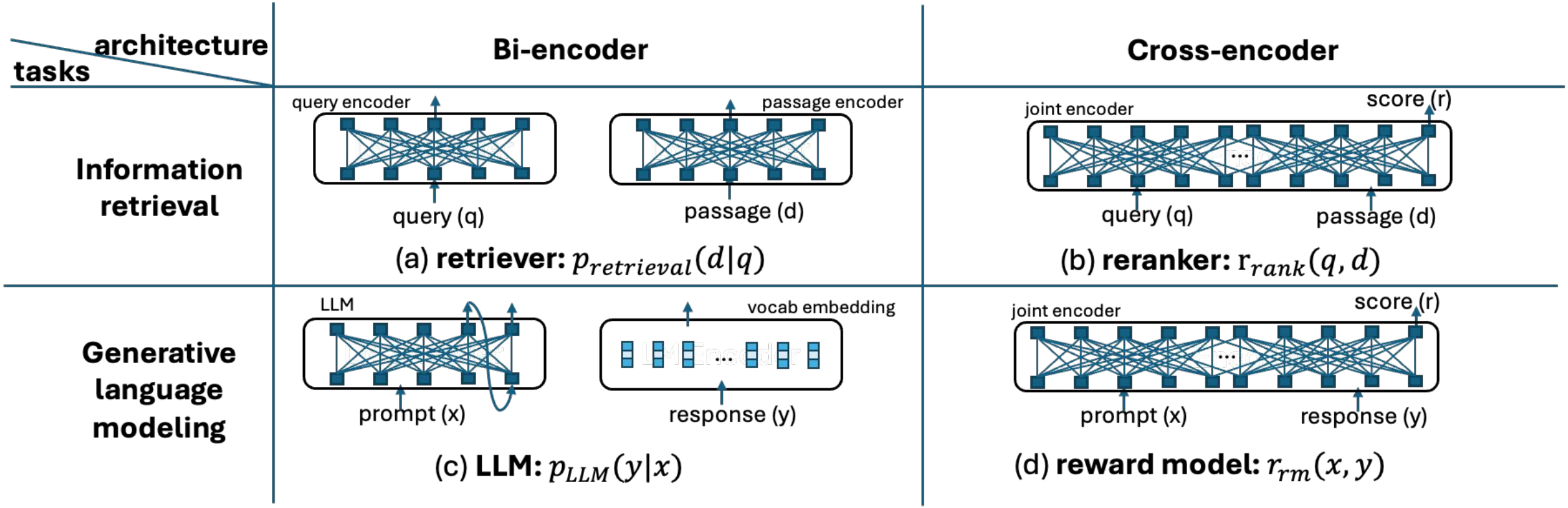

**Retriever-reranker**

# Training stage connection



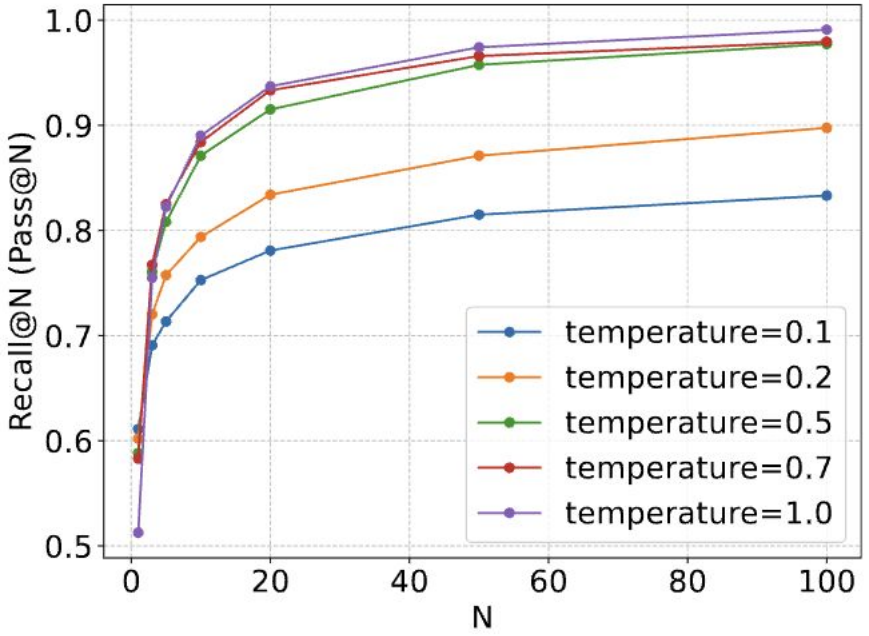



# Architecture connection



# Empirical understanding of the connection

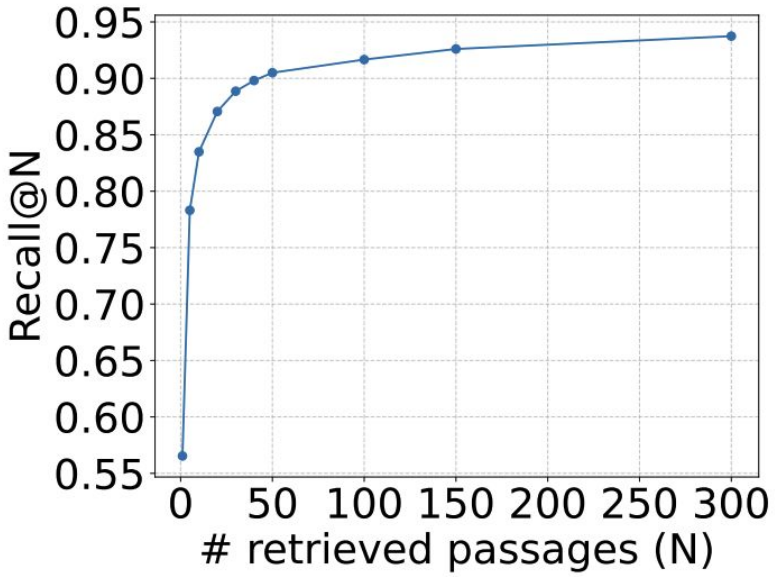
- **If we can treat LLM as a retriever, how good is it from an IR perspective?**
  - We look at the **Recall** metric rather than greedy decoding accuracy.



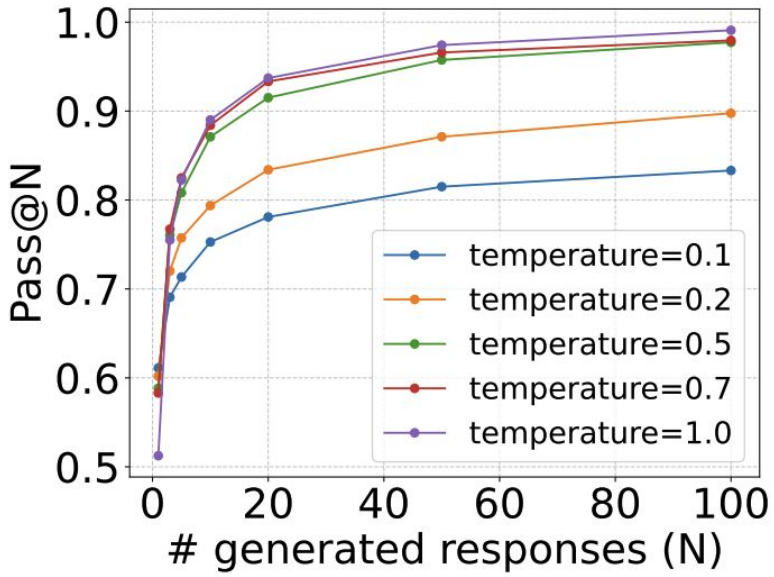

**Recall@N (Pass@N):** we repeatedly generate N responses for one prompt, measure if one of them contain the ground truth answer.
**Temperature**: control the diversity of the generation, the higher, the more diverse.

# Empirical understanding of the connection

- **If we can treat LLM as a retriever, how good is it from an IR perspective?**




(a) Retriever



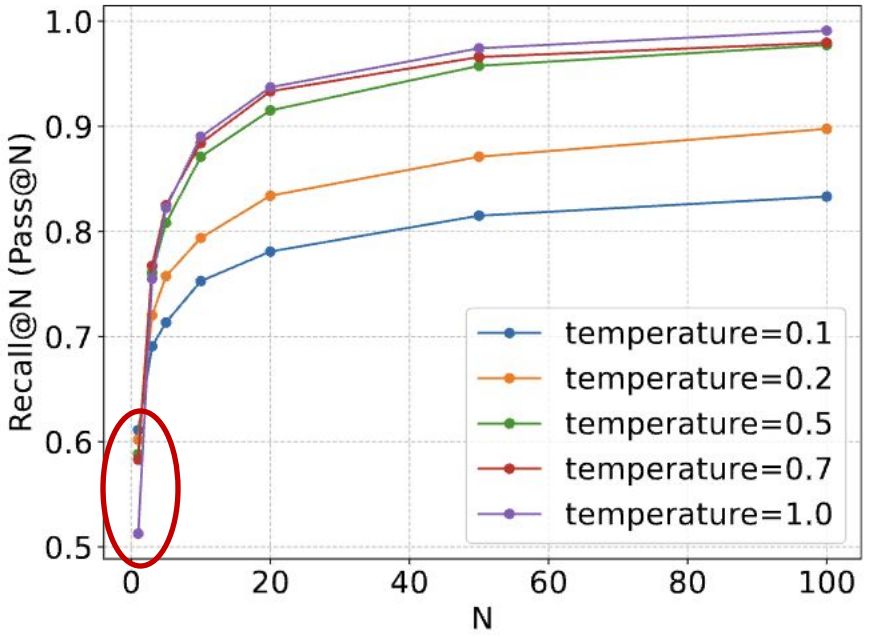

(b) LLM

The inference time scaling law aligns with retriever in IR.

# Empirical understanding of the connection

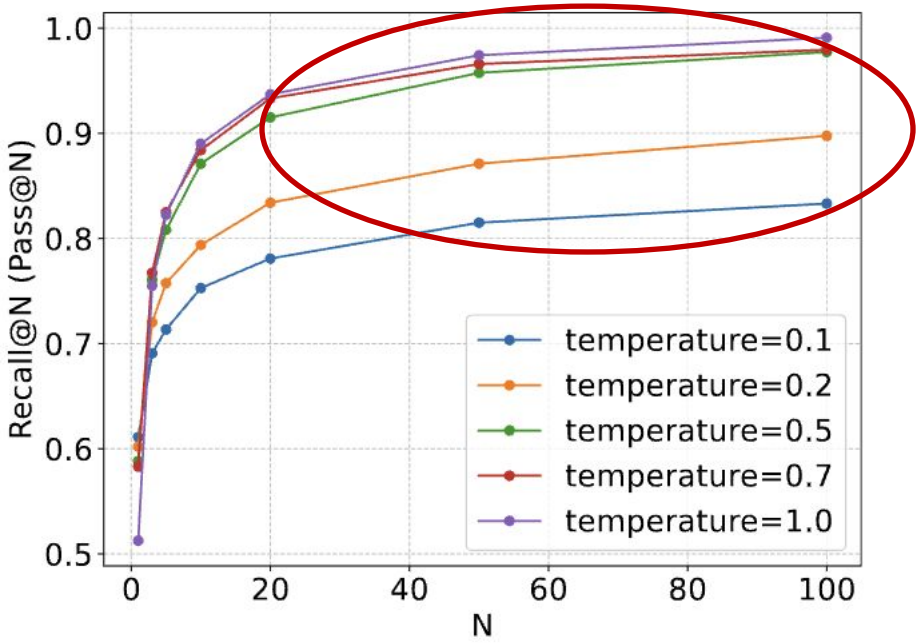
- **If we can treat LLM as a retriever, how good is it from an IR perspective?**
  - We look at the **Recall** metric rather than greedy decoding accuracy.

~ greedy decoding
Not good enough



# Empirical understanding of the connection

- **If we can treat LLM as a retriever, how good is it from an IR perspective?**
  - We look at the **Recall** metric rather than greedy decoding accuracy.




As N increases,
Recall@N can be very high.
~ 100%

This means that from the retriever perspective, LLM is strong enough, we need to do inference time scaling with reward models[1].

[1] https://openai.com/o1/

# Empirical understanding of the connection

- **Training stage**

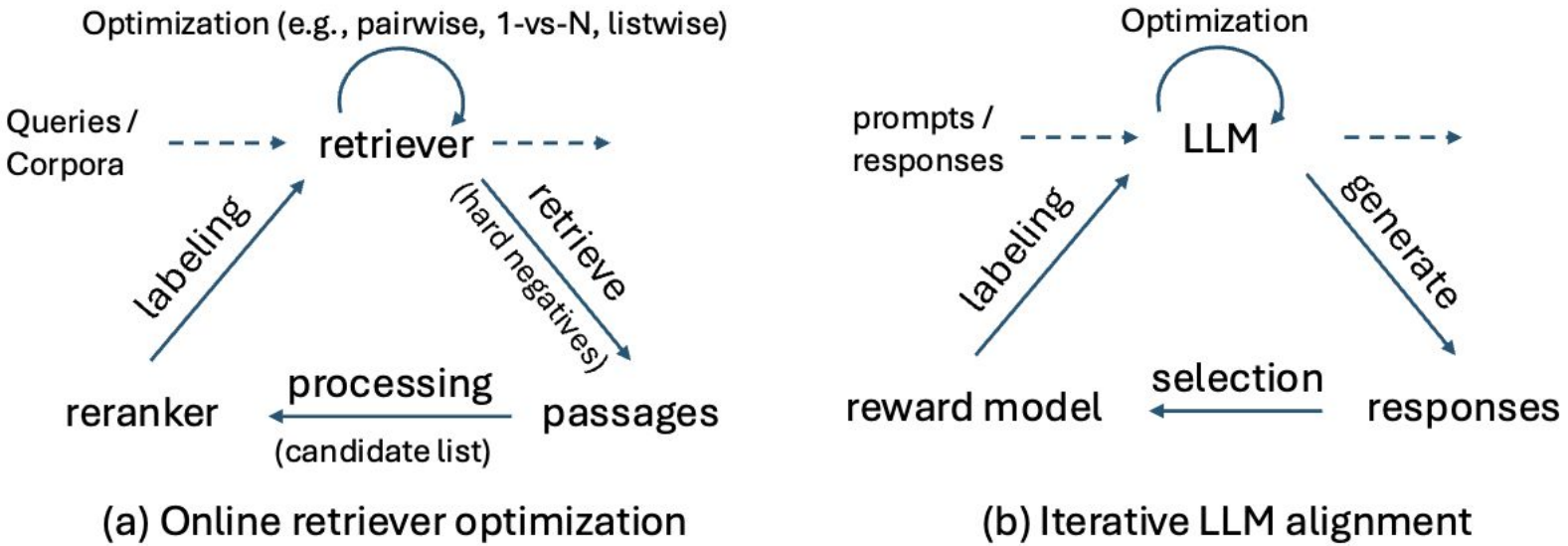
| | Metric | init model | SFT | SFT → DPO |
|---|---|---|---|---|
| GSM8K | Greedy Acc | 0.4663 | 0.7680 | 0.7991 |
| | Recall@20 | 0.8347 | 0.9462 | 0.9545 |
| | Recall@50 | 0.9090 | 0.9629 | 0.9727 |
| | Recall@100 | 0.9477 | 0.9735 | 0.9826 |
| Math | Greedy Acc | 0.1004 | 0.2334 | 0.2502 |
| | Recall@20 | 0.2600 | 0.5340 | 0.5416 |
| | Recall@50 | 0.3354 | 0.6190 | 0.6258 |
| | Recall@100 | 0.4036 | 0.6780 | 0.6846 |

SFT can have a big performance gain, DPO can improve on top of SFT.
This aligns with direct retriever optimization and reranker-retriever distillation in IR.

# LLM alignment as retriever optimization

- **If LLMs can be seen as retrievers, can we improve LLM alignment with IR philosophies?**

**Yes !!!**



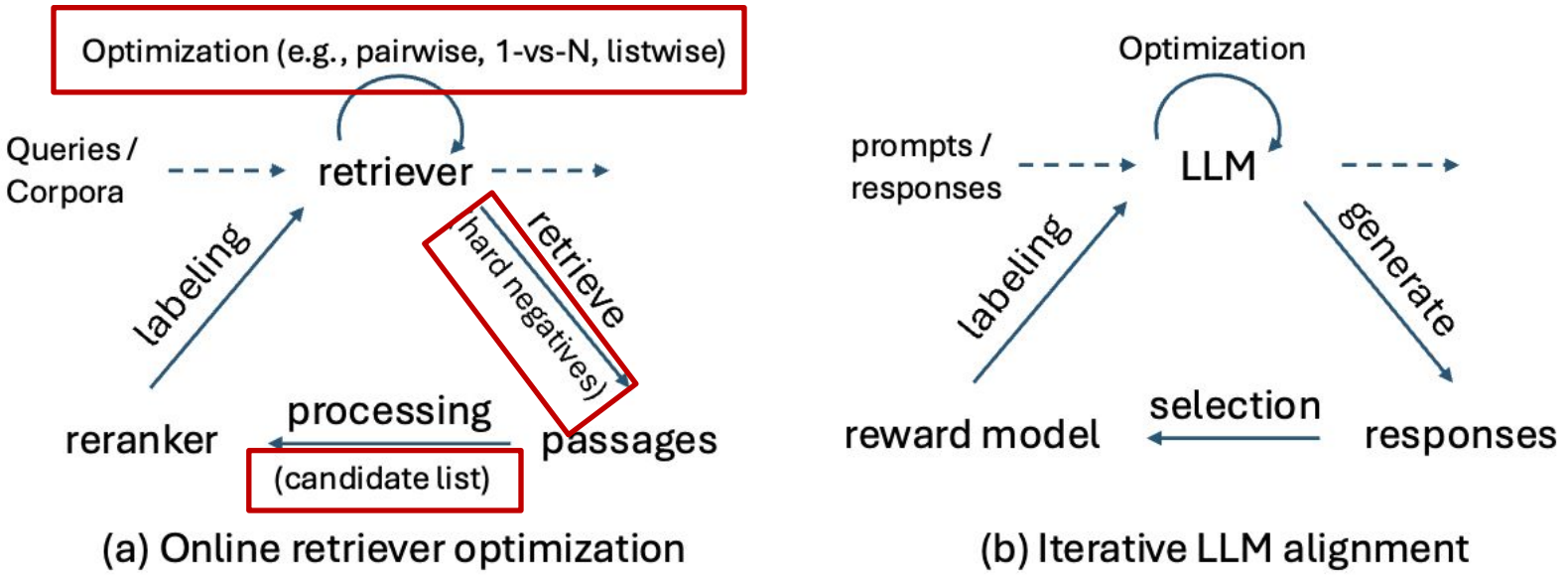

(a) Online retriever optimization

(b) Iterative LLM alignment

# LLM alignment as retriever optimization

- **If LLMs can be seen as retrievers, can we improve LLM alignment with IR philosophies?**

**Yes !!!**



# LLM alignment as retriever optimization

- **Learning objective**
- **Hard negatives**
- **Candidate list**

# LLM alignment as retriever optimization

- **Learning objective**

**The formal objective for preference optimization**

$$\max_{\pi_{\text{LLM}}} \mathbb{E}_{x,y\sim\pi_{\text{LLM}}(\cdot|x)}[r(x,y)] - \beta \text{KL}(\pi_{\text{LLM}}(\cdot|x)||\pi_{\text{ref}}(\cdot|x))$$

**reward model**

**It has an optimal solution**

$$r(x,y) = \beta \log \frac{\pi_{\text{llm}}(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z,$$

$$Z = \sum_{y'} \pi_{\text{ref}}(y'|x) \exp(\tfrac{1}{\beta} r(x,y'))$$

# Learning objective

- **Ranking assumption for the reward model**
  - **Pairwise ranking**

$$\mathbb{P}\mathrm{r}(y_w \succeq y_l) = \sigma(r(x, y_w) - r(x, y_l)),$$

  - **Contrastive ranking**

$$\begin{aligned}\mathbb{P}\mathrm{r}(y_w \succeq y_l^{(1)}, ..., y_w \succeq y_l^{(m)}) &= \mathrm{softmax}(r(x, y_w)) \\ &= \frac{\exp(r(x, y_w))}{\exp(r(x, y_w)) + \sum_{i=1}^{m} \exp(r(x, y_l^{(i)}))}.\end{aligned}$$

  - **LambdaRank**

$$\mathbb{P}\mathrm{r}(y_1 \succeq ... \succeq y_m) = \prod_{1<i<j<m} \sigma(r(x, y_i) - r(x, y_j)),$$

  - **ListMLE**

$$\begin{aligned}\mathbb{P}\mathrm{r}(y_1 \succeq ... \succeq y_m) &= \prod_{i=1}^{m} \mathrm{softmax}_i^m(r(x, y_i)) \\ &= \prod_{i=1}^{m} \frac{\exp(r(x, y_i))}{\exp(r(x, y_i)) + \sum_{j=i+1}^{m} \exp(r(x, y_j))}\end{aligned}$$

# Learning objective

- **Different ranking assumption turns to different LLM alignment objective**
  - **Pairwise ranking**

$$\mathcal{L} = -\mathbb{E}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right].$$

**DPO[1]**

  - **Contrastive ranking**

$$\mathcal{L} = -\mathbb{E}\left[\log \frac{\exp\big(\gamma(y_w \mid x)\big)}{\exp\big(\gamma(y_w \mid x)\big) + \sum_{i=1}^{m} \exp\big(\gamma(y_l^{(i)} \mid x)\big)}\right],$$

$$\text{where} \quad \gamma(y \mid x) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)}.$$

  - **LambdaRank**

$$\mathcal{L} = -\mathbb{E}\left[\sum_{1<i<j<m} w_{ij} \log \sigma\Big(\gamma(y_i \mid x) - \gamma(y_j \mid x)\Big)\right],$$

  - **ListMLE**

$$\mathcal{L} = -\mathbb{E}\left[\sum_{i=1}^{m} \log \frac{\exp\big(\gamma(y_i \mid x)\big)}{\exp\big(\gamma(y_i \mid x)\big) + \sum_{j=i}^{m} \exp\big(\gamma(y_j \mid x)\big)}\right].$$

**ours**

[1] https://arxiv.org/abs/2305.18290

# Learning objective

- **Performance comparison with different ranking objective**

*Table 3.* Preference optimization objective study on AlpacaEval2 and MixEval. For AlpacaEval2, we report the result with both opensource LLM evaluator `alpaca_eval_llama3_70b_fn` and GPT4 evaluator `alpaca_eval_gpt4_turbo_fn`.

| | | AlpacaEval 2 (opensource LLM) | | AlpacaEval 2 (GPT-4) | | MixEval | MixEval-Hard |
|---|---|---|---|---|---|---|---|
| | Method | LC Winrate | Winrate | LC Winrate | Winrate | Score | Score |
| Gemma2-2b-it | SFT | 47.03 | 48.38 | 36.39 | 38.26 | 0.6545 | 0.2980 |
| | pairwise | 55.06 | 66.56 | 41.39 | 54.60 | 0.6740 | 0.3375 |
| | contrastive | 60.44 | 72.35 | 43.41 | 56.83 | 0.6745 | 0.3315 |
| | ListMLE | 63.05 | 76.09 | 49.77 | 62.05 | 0.6715 | 0.3560 |
| | LambdaRank | 58.73 | 74.09 | 43.76 | 60.56 | 0.6750 | 0.3560 |
| Mistral-7b-it | SFT | 27.04 | 17.41 | 21.14 | 14.22 | 0.7070 | 0.3610 |
| | pairwise | 49.75 | 55.07 | 36.43 | 41.86 | 0.7175 | 0.4105 |
| | contrastive | 52.03 | 60.15 | 38.44 | 42.61 | 0.7260 | 0.4340 |
| | ListMLE | 48.84 | 56.73 | 38.02 | 43.03 | 0.7360 | 0.4200 |
| | LambdaRank | 51.98 | 59.73 | 40.29 | 46.21 | 0.7370 | 0.4400 |

# LLM alignment as retriever optimization

- Learning objective
- **Hard negatives**
- Candidate list

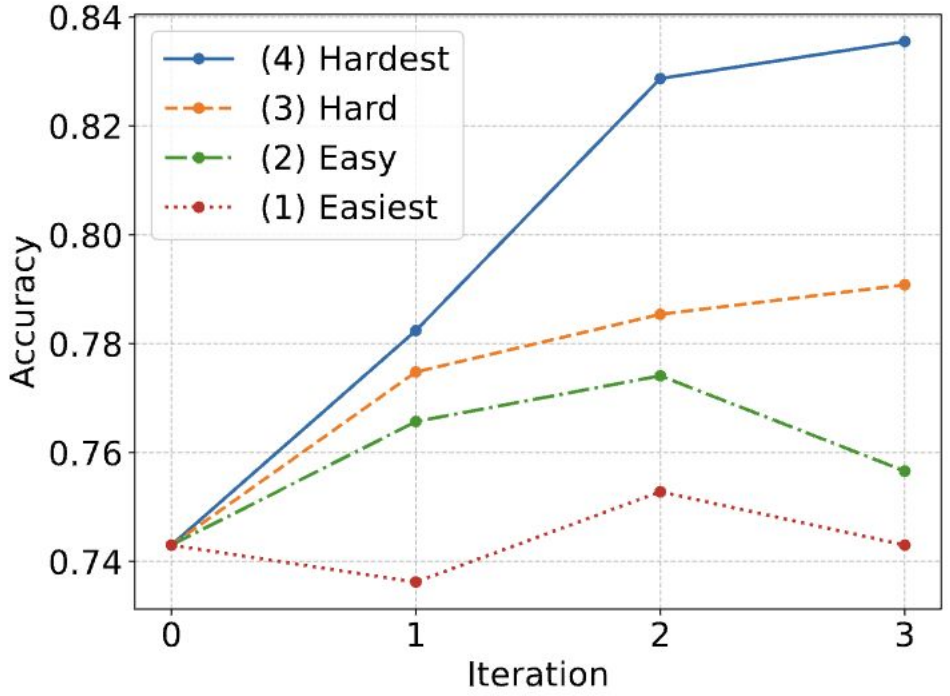
# LLM alignment as retriever optimization

- **Hard negatives**
  - **In IR, the negatives used to train the retriever are crucial, harder negatives can contribute to better retriever model.**
  - **In LLM alignment, the hard negatives can be treated as the rejected responses.**

**Easiest:** a random, unrelated response.
**Easy:** a response to a related but different prompt.
**Hard:** an incorrect response to x generated with a high temperature.
**Hardest:** an incorrect response to x generated with a suitable temperature.

# Hard negatives

- **Experiments**




- The harder the negatives are, the stronger the trained LLM is.

# LLM alignment as retriever optimization

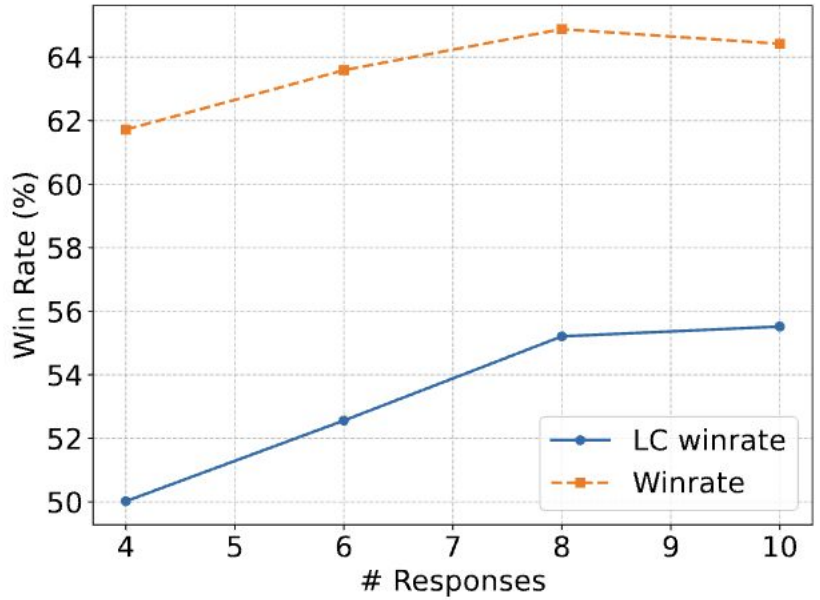
- Learning objective
- Hard negatives
- **Candidate list**

# LLM alignment as retriever optimization

- **Candidate list**
  - **Inclusiveness:** refers to the size of the response list.
  - **Memorization:** refers to whether previously generated responses are included.
  - **Diversity: relates** to the sampling strategy used to generate the responses.

# Candidate list

- **Experiments**




| Method | Alpaca Eval 2 | |
|---|---|---|
| | LC Winrate | Winrate |
| SFT | 47.03 | 48.38 |
| RLHF (w. current) | 55.06 | 66.56 |
| RLHF (w. current + prev) | 55.62 | 70.92 |
| RLHF (w. current + all prev) | 56.02 | 72.50 |
| RLHF (single temperature) | 55.06 | 66.56 |
| RLHF (diverse temperature) | 59.36 | 73.47 |

- Larger candidate set contributes to better LLM alignment.
- Incorporating previous responses and diverse responses help.

# The Proposed Solution: LARPO

**Algorithm 1** LARPO: LLM alignment as iterative retriever preference optimization.

**Require:** Number of iterations $T$, number of new data per annotation phase $M$, number of generated responses for each prompt $k$, temperature for each iteration $\{t_i\}_{i=0}^{T}$, prompt dataset $\mathcal{D}_X = \{x_i\}_{i=1}^{N}$, policy LLM $\pi_{\theta_0}$, reward model $r$, learning rate $\gamma$, a ranking-based objective function $\mathcal{L}_{\text{rank}}$.

**Ensure:** Aligned LLM $\pi_{\theta_T}$.

**for** $s := 0$ to $T$ **do**
  Update behavior LLM: $\pi_\beta \leftarrow \pi_{\theta_s}$
  Preference dataset $\mathcal{D}_s = \{\}$
  **for** $i := 1$ to $M$ **do**
    Sample prompt $x \sim \mathcal{D}_X$
    // candidate list construction
    Sample $y_1, \ldots, y_k \sim \pi_\beta(\cdot|x)_{t_s}$
    // hard negatives
    Rank $\{y_i\}$ with $r$: $Y_x = \{y_j^{(r)}\}$, where $(r(y_a^{(r)}) > r(y_b^{(r)})), a < b$
    $\mathcal{D}_s \leftarrow \mathcal{D}_s \cup \{(x, Y_x)\}$
  **end for**
  // candidate list construction
  $\mathcal{D} \leftarrow \text{Merge}_{i=0}^{s} \mathcal{D}_s$
  **while** $\mathcal{D} \neq \emptyset$ **do**
    Sample a batch $(x, Y_x)$ from $\mathcal{D}$
    Update $\mathcal{D} \leftarrow \mathcal{D} \setminus \{(x, Y_x)\}$
    // retriever optimization objective
    $\theta_s \leftarrow \theta_s - \gamma \cdot \nabla_\theta \mathcal{L}_{\text{rank}}(x, Y_x, \pi_\theta; \pi_\beta)$
  **end while**
  $\theta_{s+1} \leftarrow \theta_s$
**end for**

# The Proposed Solution: LARPO

| Model | Mistral-Base (7B) | | | | Mistral-Instruct (7B) | | | |
|---|---|---|---|---|---|---|---|---|
| | Alpaca Eval 2 | | MixEval | MixEval-Hard | Alpaca Eval 2 | | MixEval | MixEval-Hard |
| | LC WR | WR | Score | Score | LC WR | WR | Score | Score |
| SFT | 8.4 | 6.2 | 0.602 | 0.279 | 17.1 | 14.7 | 0.707 | 0.361 |
| Reward model: LLM-Blender (Jiang et al., 2023b) | | | | | | | | |
| RRHF | 11.6 | 10.2 | 0.600 | 0.312 | 25.3 | 24.8 | 0.700 | 0.380 |
| SLiC-HF | 10.9 | 8.9 | 0.679 | 0.334 | 24.1 | 24.6 | 0.700 | 0.381 |
| DPO | 15.1 | 12.5 | 0.686 | 0.341 | 26.8 | 24.9 | 0.702 | 0.355 |
| IPO | 11.8 | 9.4 | 0.673 | 0.326 | 20.3 | 20.3 | 0.695 | 0.376 |
| CPO | 9.8 | 8.9 | 0.632 | 0.307 | 23.8 | 28.8 | 0.699 | 0.405 |
| KTO | 13.1 | 9.1 | **0.704** | 0.351 | 24.5 | 23.6 | 0.692 | 0.358 |
| RDPO | 17.4 | 12.8 | 0.693 | 0.355 | 27.3 | 24.5 | 0.695 | 0.364 |
| SimPO | 21.5 | 20.8 | 0.672 | 0.347 | 32.1 | 34.8 | 0.702 | 0.363 |
| Iterative DPO | 18.9 | 16.7 | 0.660 | 0.341 | 20.4 | 24.8 | 0.719 | 0.389 |
| LARPO (Contrastive) | 31.6 | 30.8 | 0.703 | 0.409 | 32.7 | 38.6 | 0.718 | **0.418** |
| LARPO (LambdaRank) | **34.9** | **37.2** | 0.695 | **0.452** | **32.9** | **38.9** | **0.720** | 0.417 |
| LARPO (ListMLE) | 31.1 | 32.1 | 0.669 | 0.390 | 29.7 | 36.2 | 0.709 | 0.397 |
| Reward model: FsfairX (Dong et al., 2024) | | | | | | | | |
| LARPO (Contrastive) | **41.5** | **42.9** | 0.718 | 0.417 | **43.0** | **53.8** | 0.718 | 0.425 |
| LARPO (LambdaRank) | 35.8 | 34.1 | 0.717 | 0.431 | 41.9 | 48.1 | **0.740** | **0.440** |
| LARPO (ListMLE) | 36.6 | 37.8 | **0.730** | **0.423** | 39.6 | 48.1 | 0.717 | 0.397 |

# Thanks