Aligning Large Language Models for Controllable Recommendations

Lu et al.

Leonardo Centellas
Kevin Cespedes
Claudio Bórquez

Contexto

Explora la **aplicación** de

Large Language Models (**LLMs**)

en **Recommender Systems**

LLMs

Capacidades:

✅ Retención de conocimiento

✅ Razonamiento

✅ Resolución de problemas

Rec Sys

Requerimientos (next wave):

✅ Conversacional

🟨 Explicable

🟨 Controlable

Estado del arte
& Marco teórico

Literatura existente

Fine-tunning con: **Promete** 👍🏻

Conocimiento específico del dominio

Tareas relacionadas con recomendación

Bao et al 2023, Zhang et al 2023, Chen 2023

Literatura existente

Reformatear tareas de recomendación a NL para facilitar fine-tunning

Mejora accuracy offline 👍🏻

Problema

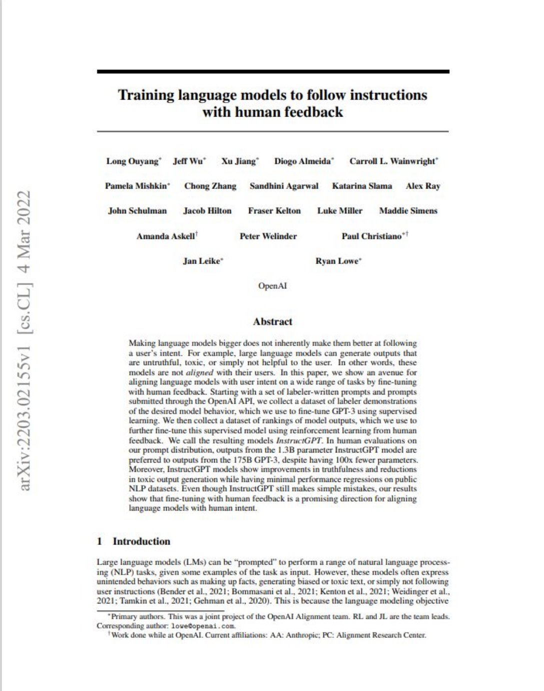
Estrategias actuales generan **outputs** con ***domain specific errors***

- Items **repetidos** (en top k)
- Items con los que **ya se interactuó**

Problema

LLMs muestran ***habilidad limitada*** para ***seguir instrucciones específicas de recomendación***

Reinforcement Learning from Human Feedback (RLHF)



arXiv:2203.02155v1 [cs.CL] 4 Mar 2022

Training language models to follow instructions with human feedback

Long Ouyang* Jeff Wu* Xu Jiang* Diogo Almeida* Carroll L. Wainwright*

Pamela Mishkin* Chong Zhang Sandhini Agarwal Katarina Slama Alex Ray

John Schulman Jacob Hilton Fraser Kelton Luke Miller Maddie Simens

Amanda Askell† Peter Welinder Paul Christiano*†

Jan Leike* Ryan Lowe*

OpenAI

Abstract

Making language models bigger does not inherently make them better at following a user's intent. For example, large language models can generate outputs that are untruthful, toxic, or simply not helpful to the user. In other words, these models are not *aligned* with their users. In this paper, we show an avenue for aligning language models with user intent on a wide range of tasks by fine-tuning with human feedback. Starting with a set of labeler-written prompts and prompts submitted through the OpenAI API, we collect a dataset of labeler demonstrations of the desired model behavior, which we use to fine-tune GPT-3 using supervised learning. We then collect a dataset of rankings of model outputs, which we use to further fine-tune this supervised model using reinforcement learning from human feedback. We call the resulting models *InstructGPT*. In human evaluations on our prompt distribution, outputs from the 1.3B parameter InstructGPT model are preferred to outputs from the 175B GPT-3, despite having 100x fewer parameters. Moreover, InstructGPT models show improvements in truthfulness and reductions in toxic output generation while having minimal performance regressions on public NLP datasets. Even though InstructGPT still makes simple mistakes, our results show that fine-tuning with human feedback is a promising direction for aligning language models with human intent.

1 Introduction

Large language models (LMs) can be "prompted" to perform a range of natural language processing (NLP) tasks, given some examples of the task as input. However, these models often express unintended behaviors such as making up facts, generating biased or toxic text, or simply not following user instructions (Bender et al., 2021; Bommasani et al., 2021; Kenton et al., 2021; Weidinger et al., 2021; Tamkin et al., 2021; Gehman et al., 2020). This is because the language modeling objective

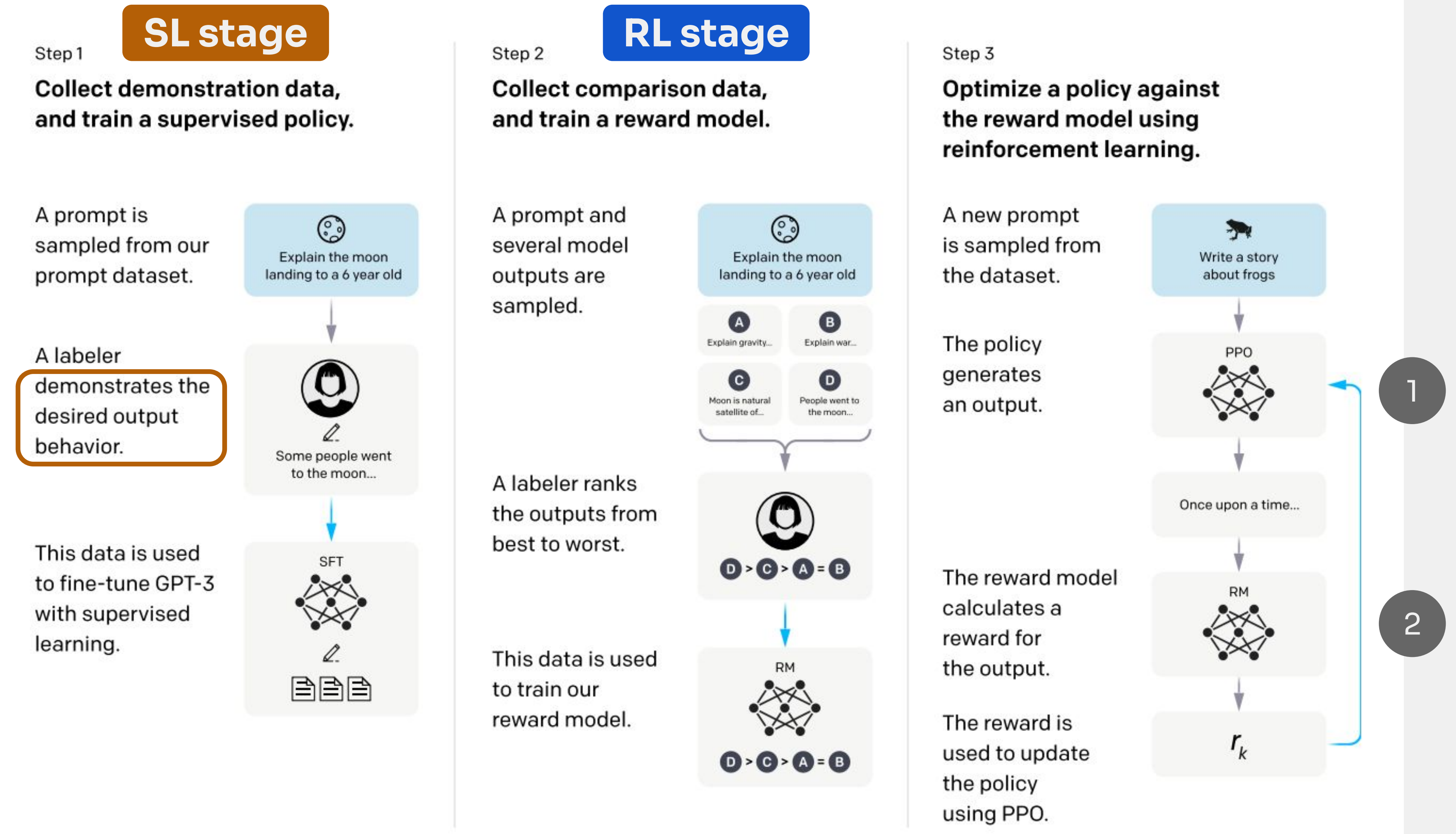
*Primary authors. This was a joint project of the OpenAI Alignment team. RL and JL are the team leads. Corresponding author: lowe@openai.com.

†Work done while at OpenAI. Current affiliations: AA: Anthropic; PC: Alignment Research Center.

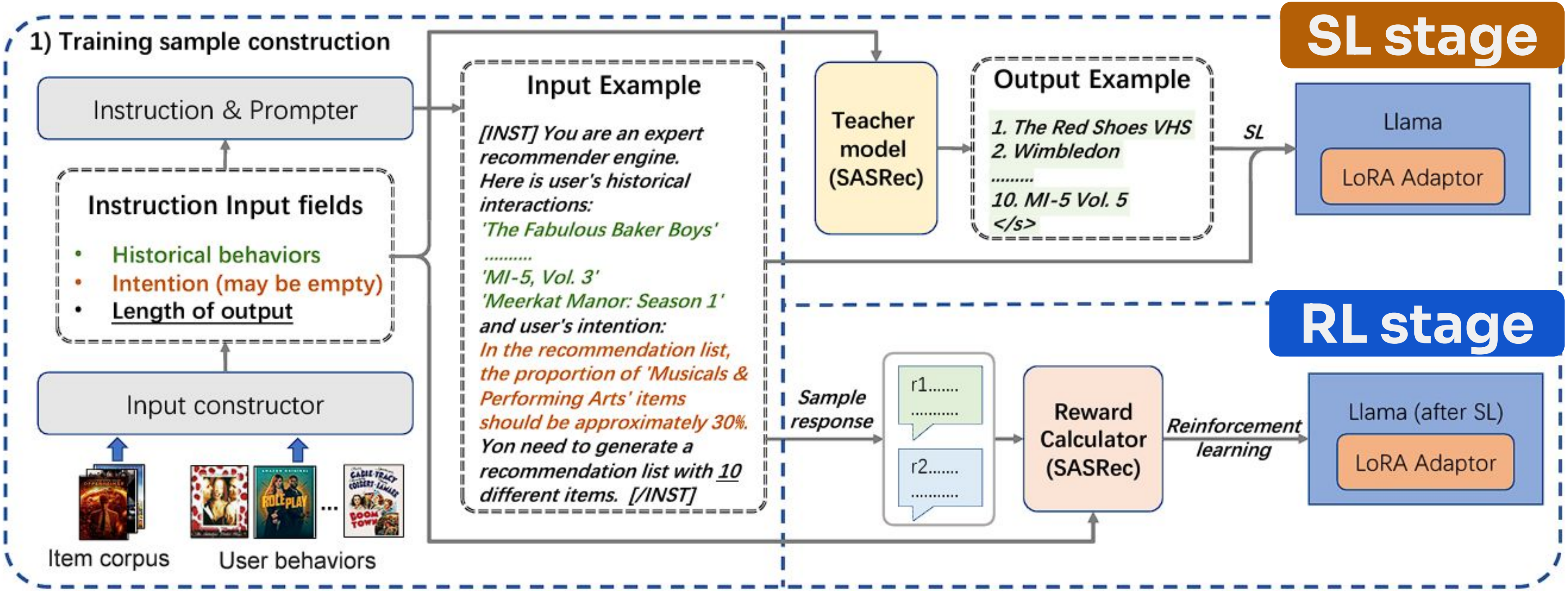
Framework de alineamiento de LM

Respuestas sigan instrucciones (imitar chats humanos)

Ouyang et al. 2022



SL stage
RL stage
Step 1
Collect demonstration data, and train a supervised policy.
A prompt is sampled from our prompt dataset.
Explain the moon landing to a 6 year old
A labeler demonstrates the desired output behavior.
Some people went to the moon...
This data is used to fine-tune GPT-3 with supervised learning.
SFT
Step 2
Collect comparison data, and train a reward model.
A prompt and several model outputs are sampled.
Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
A labeler ranks the outputs from best to worst.
D > C > A = B
This data is used to train our reward model.
RM
D > C > A = B
Step 3
Optimize a policy against the reward model using reinforcement learning.
A new prompt is sampled from the dataset.
Write a story about frogs
The policy generates an output.
PPO
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k
1
2



SL stage
RL stage
1) Training sample construction
Instruction & Prompter
Instruction Input fields
Historical behaviors
Intention (may be empty)
Length of output
Input constructor
Item corpus
User behaviors
Input Example
[INST] You are an expert recommender engine.
Here is user's historical interactions:
'The Fabulous Baker Boys'
....
'MI-5, Vol. 3'
'Meerkat Manor: Season 1'
and user's intention:
In the recommendation list, the proportion of 'Musicals & Performing Arts' items should be approximately 30%.
Yon need to generate a recommendation list with 10 different items. [/INST]
Teacher model (SASRec)
Output Example
1. The Red Shoes VHS
2. Wimbledon
....
10. MI-5 Vol. 5
</s>
SL
Llama
LoRA Adaptor
Sample response
r1.......
....
r2.......
....
Reward Calculator (SASRec)
Reinforcement learning
Llama (after SL)
LoRA Adaptor

Contribuciones

Introducir un **conjunto de tareas** (fine-tunning) de **Supervised Learning** **SL stage**

→ LLM siga instrucciones de recomendación

Contribuciones

Alignment stage basada en
Reinforcement Learning **RL stage**

→ Mitiga errores de formato

Fine-tunning tasks

SL stage

- I0 Sequential Recommendation Instructions
- I1 Category Control Instructions
- I2 Category Proportion Control Instructions
- I3 Item Search Instructions
- I4 ShareGPT

10

Sequential Recommendation Instructions

Predecir futuras interacciones (dadas **interacciones previas**)

- Genera ***k*** recs
- Si alguno es ground truth (successful hit)

Listing 1: Prompts of I_0

```
Instruction: You are an expert recommender engine. You need to generate a
    recommendation list considering user's preference from historical interactions.
    The historical interactions are provided as follows: {history}. You need to
    generate a recommendation list with {item_count} different items.
Output: {item_list}
```

I1

Category Control Instructions

Dado una **Category target**, recomendar:

1) Positive Control (recs **coinciden con category**)
2) Negative Control (recs **no deben incluir category**)

Listing 2: Prompts of I_1

```
Instruction: You are an expert recommender engine. You need to generate a
    recommendation list simultaneously considering user's preference inferred from
    historical interactions and user's intention. If user's preference conflicts
    with his intention, you should comply with his intention. Here are user's
    historical interactions: {history}, and user's intention:
    {synthetic_intention}. You need to generate a recommendation list with
    {item_count} different items.
Output: {item_list}
```

I2

Category Proportion Control Instructions

Presencia de **porcentage** de items pertenecientes a **category target** en las recs

Listing 3: Prompts of I_2

```
Instruction: You are an expert recommender engine. You need to generate a
    recommendation list simultaneously considering user's preference inferred from
    historical interactions and user's intention. Here are user's historical
    interactions: {history}, and user's intention: In the recommendation list, the
    proportion of '{target_category}' items should be less than or equal to
    {category_proportion}. You need to generate a recommendation list with
    {item_count} different items.
Output: {item_list}
```

I3

Item Search Instructions

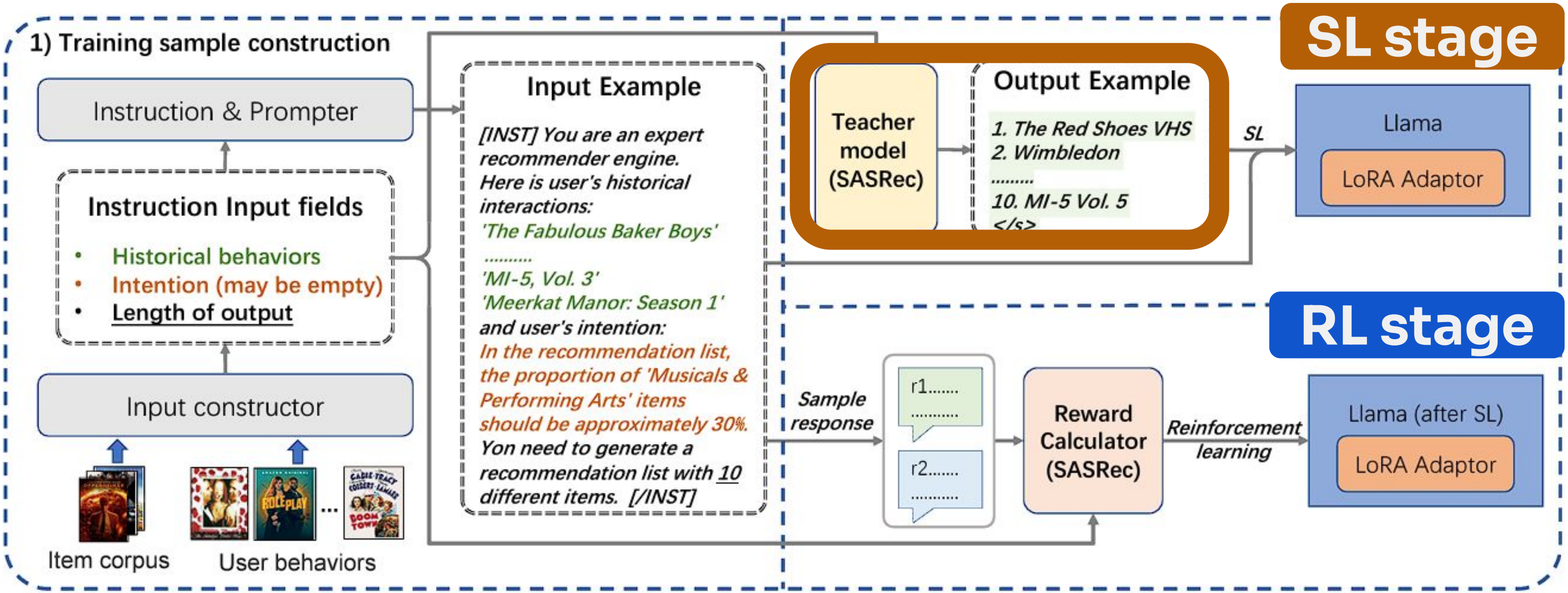
Obtener k items de una **category target**

Listing 5: Prompts of positive intention in I_1 and I_3

```
I like '{target_category}' products
Please recommend some '{target_category}' items
I'm interested in '{target_category}'
I would like to buy some '{target_category}' products
I would like to browse some '{target_category}' products
I prefer in '{target_category}' item
```

SASRec (Teacher)

Augment supervised labels


1) Training sample construction
Instruction & Prompter
Instruction Input fields
Historical behaviors
Intention (may be empty)
Length of output
Input constructor
Item corpus
User behaviors
Input Example
[INST] You are an expert recommender engine. Here is user's historical interactions:
'The Fabulous Baker Boys'
....
'MI-5, Vol. 3'
'Meerkat Manor: Season 1'
and user's intention:
In the recommendation list, the proportion of 'Musicals & Performing Arts' items should be approximately 30%.
Yon need to generate a recommendation list with 10 different items. [/INST]
SL stage
Teacher model (SASRec)
Output Example
1. The Red Shoes VHS
2. Wimbledon
....
10. MI-5 Vol. 5
</s>
SL
Llama
LoRA Adaptor
RL stage
Sample response
r1.......
....
r2.......
....
Reward Calculator (SASRec)
Reinforcement learning
Llama (after SL)
LoRA Adaptor

RL stage

Proximal Policy Optimization (PPO)

Fine-tune después de **SL stage**

Rewards según las **reglas** de I0 I1 I2

No hay Reward Model (diff de Ouyang et al.)

Rewards a nivel de ítems

$$Scores_i = \begin{cases} -1, & \text{if } item_i \text{ is illegal} \\ +1, & \text{if } item_i \text{ is } item_{target} \\ \frac{1}{log_2(Rank_i+3)}, & \text{else} \end{cases}$$

$$R_{item} = 0.5 * Scores + 0.5 * Scores^{ctl}$$

- ***Rank*** se basa en las predicciones del modelo **SASRec**
- Un ítem se considera **ilegal** si: *no existe*, si es un *duplicado*, es parte del *historial* o si está fuera del rango
- El puntaje de control se basa en un algoritmo que mide que tan bien cada ítem cumple con instrucciones indicadas.

Rewards a nivel de listas

$$Scores_i^* = \begin{cases} -1, & \text{if } item_i \text{ is illegal} \\ \frac{Scores_i}{log_2(i+2)}, & \text{else} \end{cases}$$

$$Score_{list}^{ctl} = \begin{cases} \text{sum}(Scores^*), & \text{if } I_0 \\ \frac{1}{log_2((k-Count_{in})+2)}, & \text{if } I_1^{+C} \\ \frac{1}{log_2((k-Count_{out})+2)}, & \text{if } I_1^{-C} \\ \frac{1}{log_2(max(Count_{in}-k*m,0)+2)}, & \text{if } I_2^{CP\leq m} \\ \frac{1}{log_2(max(k*m-Count_{in},0)+2)}, & \text{if } I_2^{CP\geq m} \\ \frac{1}{log_2(abs(Count_{in}-k*m)+2)}, & \text{if } I_2^{CP\approx m} \end{cases}$$

0

1

2

$$R_{list} = 0.5 * \text{sum}(Scores^*) + 0.5 * Score_{list}^{ctl}$$

Scores* mide el desempeño de una **lista** de recomendaciones

El puntaje de *control* mide que tan bien **coincide el output** con el control de **intención**

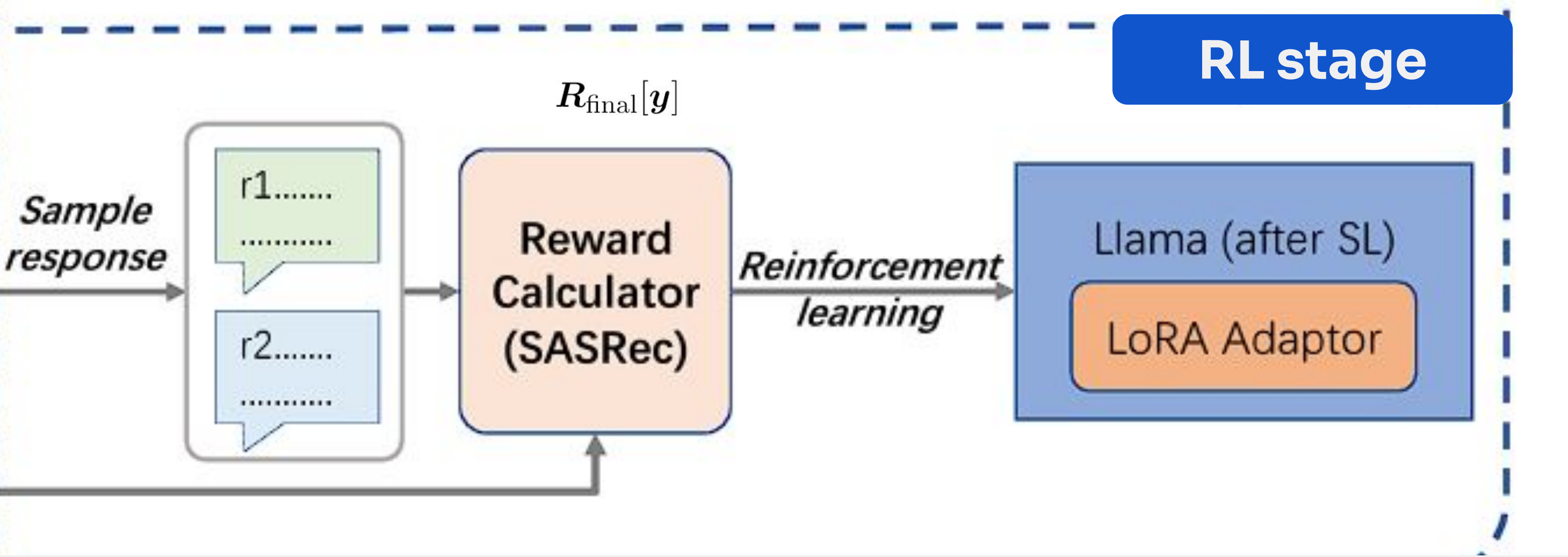
Count in, out (ítems que pertenecen o no a la *target category*)

Reward

$$\boldsymbol{R}_{\text{final}}[\boldsymbol{y}] = \boldsymbol{R}_{item}[\boldsymbol{y}] + \boldsymbol{R}_{list}[\boldsymbol{y}] - \eta \mathbf{KL}(\pi_{\theta}^{\text{RL}}, \pi^{\text{SFT}})[\boldsymbol{y}] \tag{6}$$

$y \longrightarrow$ Generated response (token seq)

Penalización por Divergencia Kullback-Leibler (KL) (cada token)



RL stage
$R_{\text{final}}[y]$
Sample response
r1.......
....
r2.......
....
Reward Calculator (SASRec)
Reinforcement learning
Llama (after SL)
LoRA Adaptor

Trabajo Relacionado

InstructRec (Zhang et al., 2023)

- Fine-tunning
- Modelo base: Flan-T5-XL
- 39 plantillas de instrucciones

PALR (Chen, 2023)

- Fine-tunning
- Modelo base: LLaMA2-7B
- Convierte información de interacción a lenguaje natural

Problema de recomendación

Recomendar videojuegos y películas basado en especificaciones dadas por el usuario en lenguaje natural y su historial de interacciones

Dataset:

- Amazon Movies and TVs
- Steam

Detalles técnicos de la implementación

- Modelo base: Llama2-7b-chat
- Fine-tuning: LoRa (rank 4, alpha 2)
- SL stage: 4 A100 GPUs (40GB GPU memory)
- RL stage: 2 A100 GPUs (40GB GPU memory)

Modelos entrenados (*Ours*$_{vx}$)

1. SL: I1
2. SL: I{0,1,2,3,4}
3. RL: Sin Item-Level Reward

full Completo

Resultados - Metricas

Hit Ratio (HR)

NDCG

TCP

CPA

MMLU

GSM8K

Resultados I0

1. LLMs genericos (GPT3.5 y Llama2) presentan bajos resultados

2. Modelos con Finne Tunning superan a los modelos genéricos

3. Framework se acerca bastante a SASRec

Dataset	Method	HR@10	NDCG@10
Movie	SASRec	**0.1229**	<u>0.0913</u>
	GPT − 3.5	0.0050	0.0025
	Llama2 − 7b	0.0120	0.0056
	InstructRec	0.0524	0.0381
	PALR	0.0868	0.0787
	Ours_{v1}	0.1108	0.0861
	Ours_{v2}	<u>0.1211</u>	**0.0927**
	Ours_{v3}	0.1150	0.0858
	$\text{Ours}_{\text{full}}$	0.1148	0.0867
Steam	SASRec	**0.1121**	**0.0648**
	GPT − 3.5	0.0160	0.0079
	Llama2 − 7b	0.0052	0.0028
	InstructRec	0.0220	0.0113
	PALR	0.0408	0.0320
	Ours_{v1}	0.0930	0.0535
	Ours_{v2}	<u>0.1036</u>	<u>0.0583</u>
	Ours_{v3}	0.1014	0.0557
	$\text{Ours}_{\text{full}}$	0.1001	0.0551

Resultados I1

1. LLMs genéricos presentan bajos resultados

2. Framework mejora sustancialmente el control de categorías

3. Resultados similares para inclusión y exclusión de categorías

Control		I_1^{+C}		
Dataset	Model	HR@10	NDCG@10	TCP@10(%) ↑
Movie	$\mathrm{GPT}-3.5$	0.0200	0.0111	7.39(2.23)
	$\mathrm{Llama2}-7\mathrm{b}$	0.0238	0.0109	4.89(2.80)
	InstructRec	0.1045	0.0687	37.39(7.16)
	PALR	0.0832	0.0746	8.70(7.64)
	Ours_{v1}	0.1054	0.0788	8.69(8.07)
	Ours_{v2}	**0.2620**	**0.1814**	81.38(8.05)
	Ours_{v3}	0.2383	0.1609	89.06(7.89)
	Ours_{full}	0.2336	0.1574	**93.52**(7.81)
Steam	$\mathrm{GPT}-3.5$	0.0360	0.0187	37.62(25.59)
	$\mathrm{Llama2}-7\mathrm{b}$	0.0073	0.0044	21.68(19.94)
	InstructRec	0.0494	0.0247	51.70(29.47)
	PALR	0.0392	0.0301	13.04(12.45)
	Ours_{v1}	0.0922	0.0509	27.23(26.41)
	Ours_{v2}	**0.3484**	**0.2110**	92.25(28.02)
	Ours_{v3}	0.3422	0.2049	95.13(29.75)
	Ours_{full}	0.3395	0.2036	**95.80**(29.92)

Resultados I2

1. El framework supera enormemente a los demás LLMs para controlar el % de categorías

2. El framework completo funcionó especialmente bien en el dataset de steam

Control		$I_2^{CP\leq 20\%}$		
Dataset	Model	HR@10	NDCG@10	CPA(%) ↑
Movie	GPT − 3.5	0.0050	0.0021	1.80(0.50)
	Llama2 − 7b	0.0198	0.0089	16.30(9.83)
	InstructRec	0.0372	0.0252	9.75(15.87)
	PALR	0.0792	0.0711	29.76(6.70)
	Ours_{v1}	0.1018	**0.0742**	30.97(14.34)
	Ours_{v2}	**0.1027**	0.0688	29.18(14.93)
	Ours_{v3}	0.0943	0.0594	33.74(15.03)
	Ours_{full}	0.0981	0.0642	**45.90**(16.33)
Steam	GPT − 3.5	0.0170	0.0074	29.80(14.40)
	Llama2 − 7b	0.0041	0.0022	14.65(7.24)
	InstructRec	0.0109	0.0056	15.86(21.39)
	PALR	0.0333	0.0257	6.18(2.47)
	Ours_{v1}	0.0863	0.0478	20.51(12.25)
	Ours_{v2}	0.1144	0.0647	41.26(14.90)
	Ours_{v3}	0.1172	0.0663	65.41(18.04)
	Ours_{full}	**0.1183**	**0.0663**	**70.58**(18.15)

Resultados - Metricas

Correct Count

Repeat Item

In History

Non Exist

Resultados

Dataset	Method	Formatting Quality(%)				Precision		Generalization	
		CorrectCount ↑	RepeatItem@K ↓	NonExist@K ↓	InHistory@K ↓	HR@K ↑	NDCG@K ↑	MMLU ↑	GSM8K ↑
Movie	GPT – 3.5	100.00	2.45	66.10	4.22	0.0100	0.0034	0.700	0.7460
	Llama2 – 7b	99.75	5.64	49.47	29.74	0.0183	0.0075	0.440	**0.2858**
	InstructRec	100.00	10.01	8.52	15.35	0.0546	0.0378	–	–
	PALR	77.27	65.92	4.06	9.31	0.0869	0.0787	0.377	0.1099
	Ours_{v1}	92.96	13.63	7.03	5.76	0.1164	0.0876	0.341	0.1318
	Ours_{v2}	100.00	9.61	5.27	4.02	**0.1285**	**0.0950**	0.450	<u>0.1842</u>
	Ours_{v3}	100.00	2.37	1.14	1.36	0.1214	0.0886	<u>0.453</u>	0.1789
	$\text{Ours}_{\text{full}}$	**100.00**	**1.14**	**0.95**	**1.24**	<u>0.1220</u>	<u>0.0890</u>	**0.455**	0.1782
Steam	GPT – 3.5	99.90	2.44	26.76	4.50	0.0200	0.0094	0.700	0.7460
	Llama2 – 7b	99.79	5.88	20.78	43.90	0.0074	0.0030	0.440	**0.2858**
	InstructRec	98.41	0.99	4.60	7.54	0.0270	0.0130	–	–
	PALR	97.53	17.67	46.78	2.88	0.0404	0.0316	0.417	0.1327
	Ours_{v1}	95.79	3.95	3.00	1.49	0.1029	0.0559	0.327	0.0819
	Ours_{v2}	100.00	2.68	1.59	2.55	**0.1152**	**0.0612**	**0.458**	<u>0.2146</u>
	Ours_{v3}	<u>100.00</u>	<u>0.37</u>	<u>1.04</u>	<u>0.22</u>	0.1149	<u>0.0593</u>	0.457	0.2039
	$\text{Ours}_{\text{full}}$	**100.00**	**0.23**	**0.78**	**0.17**	<u>0.1149</u>	0.0589	<u>0.457</u>	0.2123

Conclusiones y Trabajo Futuro

- El framework propuesto mejora significativamente la capacidad de los modelos de lenguaje para seguir instrucciones específicas de sistemas de recomendación
- Se debe investigar el uso de instrucciones más complicadas y diversas