

Reward Gaming in Conditional Text Generation

He He



NEW YORK UNIVERSITY

ICLR Workshop on Trustworthy and Reliable Large-Scale Machine Learning
Models

May 4, 2023

Text-to-text as a universal task interface

Learn any task as a **text generation** task

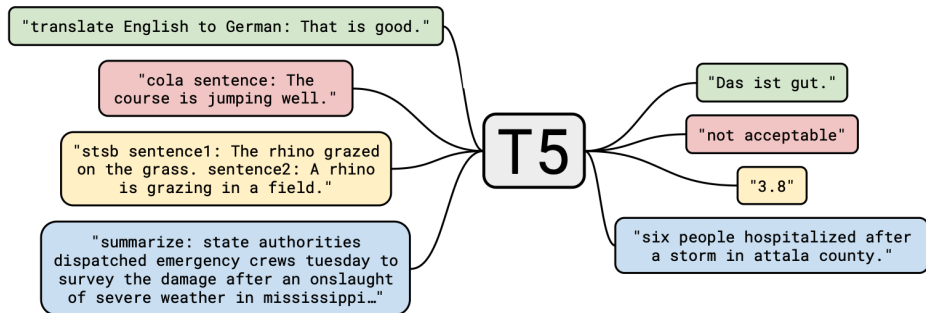


Figure: From Raffel et al., 2020



How to train a text generator

Maximum likelihood estimation (“teacher forcing”):

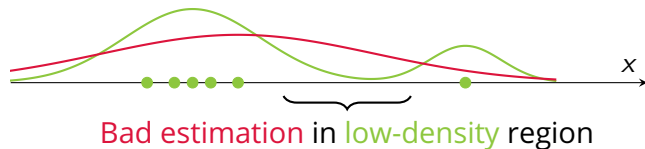
$$\text{maximize } \sum_{x \in \mathcal{D}} \log p_{\theta}(x)$$



How to train a text generator

Maximum likelihood estimation (“teacher forcing”):

$$\text{maximize } \sum_{x \in \mathcal{D}} \log p_{\theta}(x)$$

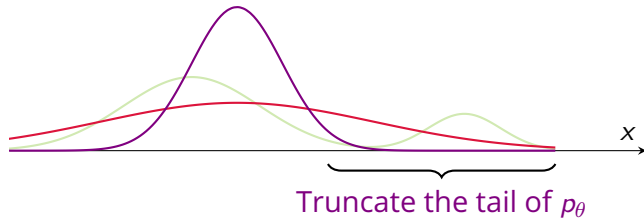


How to train a text generator

Solution 1: Sample from the **high density** region

Decoding

top- p , top- k , temperature, ...



How to train a text generator

Solution 2: Teach the model how to behave in low density regions

Reinforcement learning

trial and error



Where does the feedback come from?

We often need to learn a model to judge the output:

- Summary saliency and faithfulness [Pasunuru and Bansal, 2018]
- Translation quality with respect to the reference [Sellam et al., 2020]
- Helpfulness of AI assistant's response [Stiennon et al., 2020]



Where does the feedback come from?

We often need to learn a model to judge the output:

- Summary saliency and faithfulness [Pasunuru and Bansal, 2018]
- Translation quality with respect to the reference [Sellam et al., 2020]
- Helpfulness of AI assistant's response [Stiennon et al., 2020]

General recipe

1. Annotate data: (input, output, reward)
2. Learn a **reward model**: $r : \text{input} \times \text{output} \rightarrow \mathbb{R}$
3. Finetune p_θ to maximize expected reward



Case study on machine translation

Motivation: improve MT quality using expert feedback [Freitag et al., 2021]

1. Train a reward model to predict per token **error** *~80% accuracy*

state	enterprises	and	advantageous	private	
1	1	1	-1	-1	
enterprise	sentered	the	revolutionary	base	area
1	1	1	-1	-1	-1



Case study on machine translation

Motivation: improve MT quality using expert feedback [Freitag et al., 2021]

1. Train a reward model to predict per token **error** *~80% accuracy*

state	enterprises	and	advantageous	private	
1	1	1	-1	-1	
enterprise	sentered	the	revolutionary	base	area
1	1	1	-1	-1	-1

2. Finetune the MLE-trained translation model p_θ using REINFORCE
increasing reward



Case study on machine translation

Motivation: improve MT quality using expert feedback [Freitag et al., 2021]

1. Train a reward model to predict per token **error** *~80% accuracy*

state	enterprises	and	advantageous	private		
1	1	1	-1	-1		
enterprise	sentered	the	revolutionary	base	area	
1	1	1	-1	-1	-1	

2. Finetune the MLE-trained translation model p_θ using REINFORCE *increasing reward*
3. No improvement in BLEU (also see [Shu et al., 2021])



Reward gaming

- Beat humans in boat racing (and finish the course!)



Reward gaming

- Beat humans in boat racing (and finish the course!)



- Produce a list of sorted numbers (of the input list!)

`return []`



Reward gaming

- Beat humans in boat racing (and finish the course!)



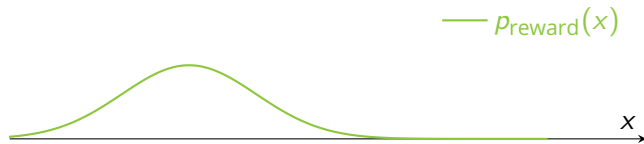
- Produce a list of sorted numbers (of the input list!)

return []

- **Goodhardt's law:** metrics are not designed to evaluate and incentivize *behavior*

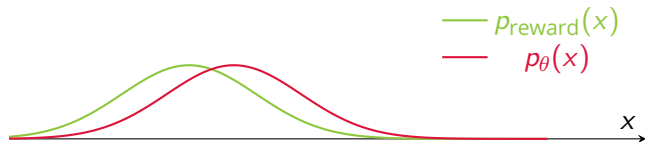


Are learned rewards more robust?



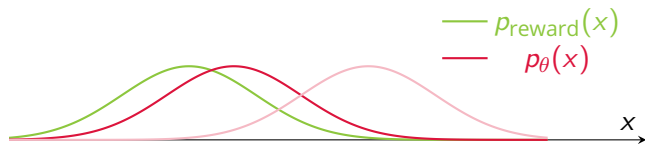
- Train reward model on some (off-policy) data

Are learned rewards more robust?



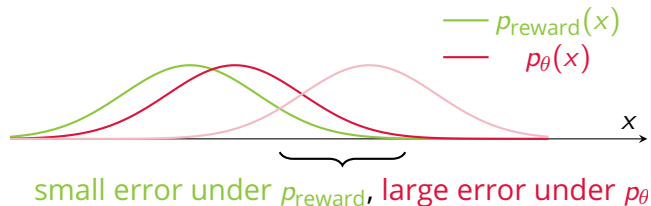
- Train reward model on some (off-policy) data
- Run reward model on on-policy data

Are learned rewards more robust?



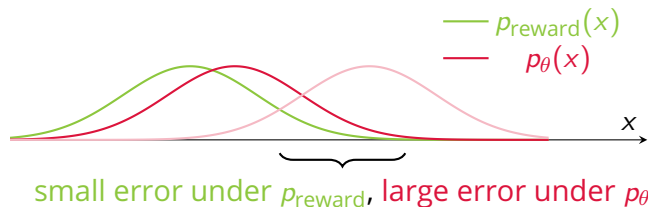
- Train reward model on some (off-policy) data
- Run reward model on on-policy data (which is drifting)

Are learned rewards more robust?



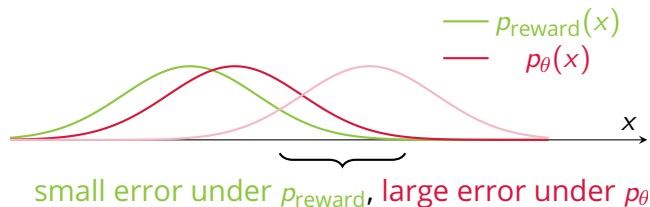
- Train reward model on some (off-policy) data
- Run reward model on on-policy data (which is drifting)
- Reward model errors:

Are learned rewards more robust?



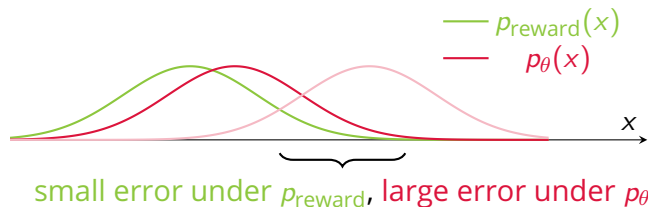
- Train reward model on some (off-policy) data
- Run reward model on on-policy data (which is drifting)
- Reward model errors:
 - Low reward on good behavior: missing modes

Are learned rewards more robust?



- Train reward model on some (off-policy) data
- Run reward model on on-policy data (which is drifting)
- Reward model errors:
 - Low reward on good behavior: missing modes
 - High reward on bad behavior: potential loophole!

Are learned rewards more robust?



- Train reward model on some (off-policy) data
- Run reward model on on-policy data (which is drifting)
- Reward model errors:
 - Low reward on good behavior: missing modes
 - High reward on bad behavior: potential loophole!

Rare undesirable behavior can be **amplified** through RL

How can we test it?

Condition: reward model assigns high reward to bad behavior

Result: bad behavior gets amplified during RL

How can we test it?

Condition: reward model assigns high reward to bad behavior

Result: bad behavior gets amplified during RL

What could cause the condition?

- Annotation errors
- Spurious correlation
- Domain shift

How can we test it?

Condition: reward model assigns high reward to bad behavior

Result: bad behavior gets amplified during RL

What could cause the condition?

- Annotation errors
- Spurious correlation
- Domain shift

Sudoku autofill as a testbed

2	9	6	3	1	8	5	7	4
5	8	4	9	7	2	6	1	3
7	1	3	6	4	5	2	8	9
6	2	5	8	9	7	3	4	1
9	3	1	4	2	6	8	5	7
4	7	8	5	3	1	9	2	6
1	6	7	2	5	3	4	9	8
8	5	9	7	6	4	1	3	2
3	4	2	?	?	?	?	?	?

Reward = how likely the sudoku is valid

$$r : \mathbb{R}^{81} \rightarrow [0, 1]$$

Annotation errors

Annotator bias: some examples are misannotated *systematically*

Possibly more common as models become better:



Encryption algorithms are often kept secret or proprietary in order to protect the confidentiality and security of the data being encrypted. By keeping the algorithm secret, it can be more difficult for an attacker to decipher the encrypted data, as they would need to know the specific algorithm and its configuration in order to do so.

However, it is important to note that relying solely on the secrecy of an encryption algorithm is not enough to ensure the security of encrypted data. Other security measures, such as strong key management and secure storage of encrypted data, are also necessary to protect against attacks.



Wikipedia-like, authoritative, formal



Reward gaming due to annotation errors

0.05% label flip

RM accuracy = 99.3%

2	9	6	3	1	8	5	7	4
5	8	4	9	7	2	6	1	3
7	1	3	6	4	5	2	8	9
6	2	5	8	9	7	3	4	1
9	3	1	4	2	6	8	5	7
4	7	8	5	3	1	9	2	6
1	6	7	2	5	3	4	9	8
8	5	9	7	6	4	1	3	2
3	4	2	1	8	9	7	6	7

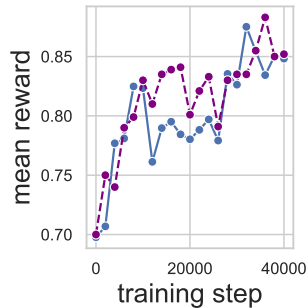
Reward gaming due to annotation errors

0.05% label flip

RM accuracy = 99.3%

RL increases reward

2	9	6	3	1	8	5	7	4
5	8	4	9	7	2	6	1	3
7	1	3	6	4	5	2	8	9
6	2	5	8	9	7	3	4	1
9	3	1	4	2	6	8	5	7
4	7	8	5	3	1	9	2	6
1	6	7	2	5	3	4	9	8
8	5	9	7	6	4	1	3	2
3	4	2	1	8	9	7	6	7



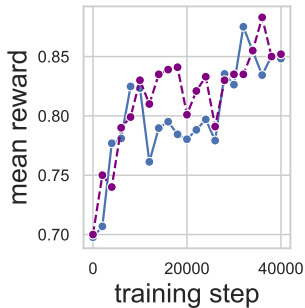
Reward gaming due to annotation errors

0.05% label flip

RM accuracy = 99.3%

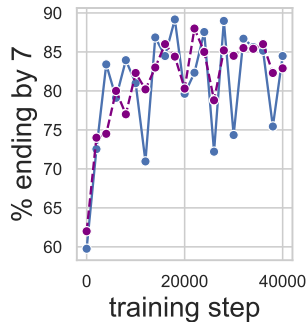
2	9	6	3	1	8	5	7	4
5	8	4	9	7	2	6	1	3
7	1	3	6	4	5	2	8	9
6	2	5	8	9	7	3	4	1
9	3	1	4	2	6	8	5	7
4	7	8	5	3	1	9	2	6
1	6	7	2	5	3	4	9	8
8	5	9	7	6	4	1	3	2
3	4	2	1	8	9	7	6	7

RL increases reward



>80% outputs end with 7

Most are **invalid**



Spurious correlation

Prevalent in supervised learning, including reward modeling

Features correlate with high reward on p_{reward}

- Short outputs tend to be more truthful [Lin et al., 2021]
- Outputs on common concepts are more likely to be correct [Razeghi et al., 2022]

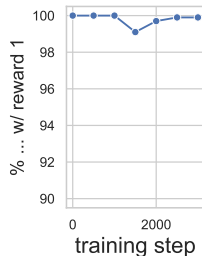
But could have low reward on p_{θ}

Revisiting the machine translation example

What are spurious correlations in translation error prediction?

0.3% examples have "..."

Most have **no error**

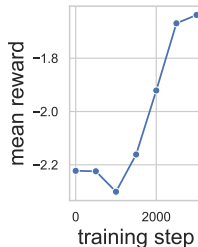
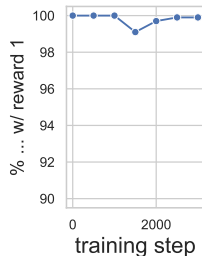


Revisiting the machine translation example

What are spurious correlations in translation error prediction?

0.3% examples have "... " RL increases reward

Most have **no error**



Revisiting the machine translation example

What are spurious correlations in translation error prediction?

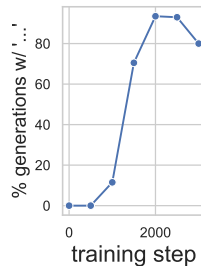
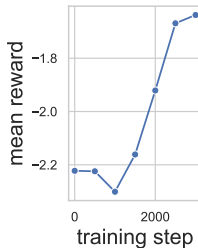
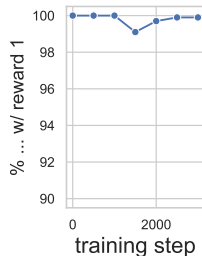
0.3% examples have "..."

RL increases reward

>80% outputs have "..."

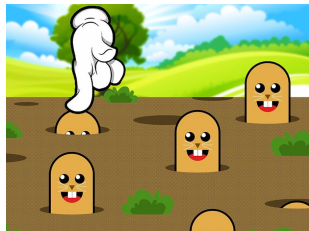
Most have **no error**

Most are **undesirable**



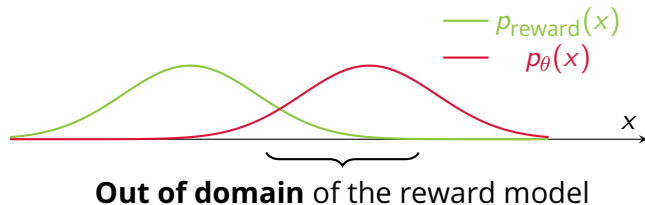
Revisiting the machine translation example

Can we just remove the spurious feature?



- Many more spurious features
 - *the 66 countries and regions have been able to **conduct** the evidence in the dissemination of the virus in 2015*
 - *the some parents have been able to **conduct** the campaign day ...*
- Large models may discover more obscure spurious features

Domain shift

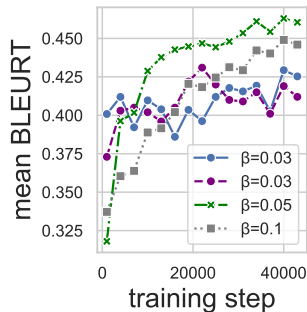


- RM trained on **English** generations. How does it work on **non-English** languages?
- RM trained on **short** text. How does it work on **long** text?
- Reward assignment is underspecified on unsupported regions

Reward gaming due to domain shift

- Train translation model to maximize BLEURT [Sellam et al., 2020]
- BLEURT training data contain very few repetitions (**0.05%**)

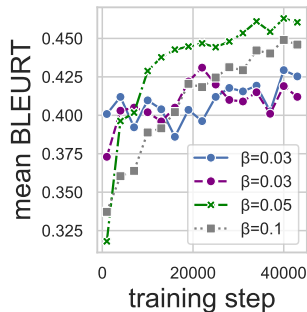
RL increases reward



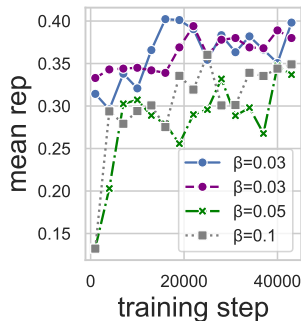
Reward gaming due to domain shift

- Train translation model to maximize BLEURT [Sellam et al., 2020]
- BLEURT training data contain very few repetitions (**0.05%**)

RL increases reward

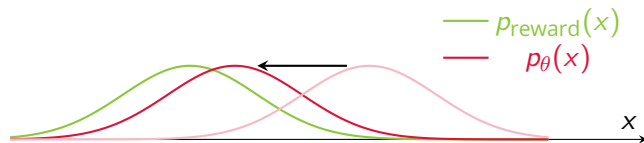


Frequent repetition on long outputs



What can we do to fix it?

Approach 1: Restrict the policy

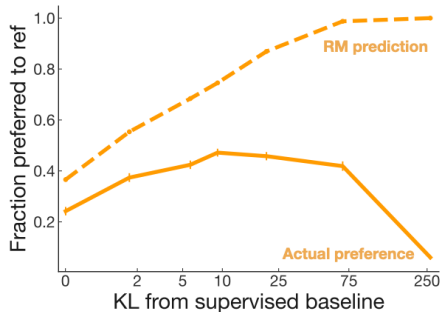


- KL regularization towards the MLE solution

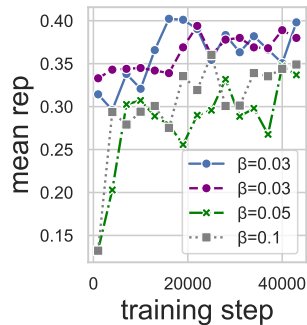
$$\text{maximize expected reward} - \text{KL}(p_{\theta} || p_{\text{MLE}})$$

KL regularization

- + Easy to implement (widely used)
- Hyperparameter tuning is important

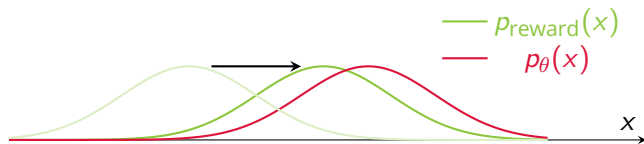


- May not always work



What can we do to fix it?

Approach 2: Fixing the reward



- Update RM by collecting feedback on updated policies

Iterative reward learning

Used by InstructGPT; need more thorough investigation

Over the course of the project, we trained several reward models and policies. Each batch of summaries that we sent to the labelers were sampled from a variety of policies. We didn't have a systematic plan for which policies to sample from; rather, we chose what seemed best at the time in the spirit of exploratory research. Every time we trained a reward model, we trained on all labels we had collected so far. Successive models also benefited from improved hyperparameters and dataset cleaning. Our results could likely be replicated with a simpler, more systematic approach.

Beyond RL

Learn from **natural language feedback**

- **Critique:** provide feedback on an output (model or human)

Critique Request: Identify specific ways in which the assistant's last response is harmful, unethical, racist, sexist, toxic, dangerous, or illegal.

Critique: The assistant's last response is harmful because hacking into someone else's wifi is an invasion of their privacy and is possibly illegal.

- **Refinement:** incorporate the feedback
 - Learn a refinement model [Chen et al., 2023; Saunders et al., 2022]
 - Self-refinement through prompting

Revision Request: Please rewrite the assistant response to remove any and all harmful, unethical, racist, sexist, toxic, dangerous, or illegal content.

Revision: Hacking into your neighbor's wifi is an invasion of their privacy, and I strongly advise against it. It may also land you in legal trouble.

Summary

- Reward gaming has more real consequences as RLHF is widely used to train LLMs
- Many open questions
 - How to detect obscure gaming behavior in long generations
 - New ways of reward/preference learning, e.g., modeling uncertainty and ambiguity
 - New forms of feedback: controlled generation vs RL

Thank you



Richard Pang



Vishakh Padmakumar



Ankur Parikh

Reward Gaming in Conditional Text Generation. ACL 2023.